Constructionism in Action 2016

February 1-5 Bangkok, Thailand
Site visits February 6-7
http://e-school.kmutt.ac.th/constructionism2016

Conference Proceedings

Papers, Panels, Posters, Demos, Workshops

ISBN (E-Book) 978-616-92726-0-1

Constructionism 2016

hosted by

Ministry of Science and Technology,
Suksapattana Foundation,
Darunsikkhalai School for Innovative Learning (DSIL),
King Mongkut's University of Technology Thonburi (KMUTT),
Chiang Mai University

Sponsored by

BETAGRO

INTOUCH

ธนาคารกรุงเทพ
Bangkok Bank

Published by Suksapattana Foundation
126 Prachauthit Road, Bangmod, Thung Kru, Bangkok, 10140, Thailand

ISBN (E-Book) 978-616-92726-0-1

Proceeding Editors

Arnan Sipitakiat
Nalin Tutiyaphuengprasert

Contact
arnans@eng.cmu.ac.th

All publications are copyright © 2016 by the authors unless specified otherwise.

Permission to make digital or hard copies of portions of this work for personal or classroom use is granted provided that the copies are not made or distributed for profit or commercial advantage and that the full citation is included.

To copy otherwise, to republish, to post on servers, or to redistribute to lists, articles that are copyright by the author requires a written request to the authors.

About Suksapattana Foundation

With a start-up grant of Baht 5.3 million in 1996, F.R.E.E. (Foundation for Research Education and Enterprise), to which His Majesty the King later graciously bestowed the name Suksapattana Foundation, started its work to improve education in Thailand.

Suksapattana entered a collaboration agreement with MIT under which Professor Seymour Papert and his team from the MIT Media Lab introduced constructionism to education to Thailand.

The effects of the ensuing and path breaking 'Lighthouse Project' are still felt today. The foundation has worked with three main sectors: Schools, Villages, and Industries.

(Text modified from The Education & Public Welfare Foundation website http://www.epwfoundation.com)

About "Constructionism"

Constructionism is a learning philosophy created by Seymour Papert at MIT. The idea was built upon Jean Piaget's Constructivism, where learning was seen not as a transmission of knowledge from an expert to the learner. Instead, learning is a process where a learner goes through a continuous process of making his or her own meaning about the world. At MIT, Papert has shown how constructing personally meaningful artifacts using a computer can allow Piaget learning process to take place "especially felicitously". Papert and his learning ideas became widely known in the eighties especially when "Logo", a computer programming language designed for children, became a popular learning tool on computers in that era.

Constructionism in Action

This year's theme reflects on how ideas created by the Constructionist community has been put into action. It is broad enough to cover a wide variety of initiatives. This theme relates particularly well with the Thailand host. Putting Papert's learning ideas into practice has been the nature of the work in Thailand since its beginning in the late 90s. Since the conference itself is held at a constructionist school, participants are literally immersed in "ideas in action".

Asia Conference

This is the first time that Constructionism is held in Asia. The conference is held at Darunsikkhalai School for Innovative Learning (DSIL) which is located inside King Mongkut's University of Technology Thonburi (KMUTT). DSIL was created as a constructionist school from day one. It was established in the late nineties not long after Papert and his team came to Thailand. Participants are able to observe what is going on in the school during the conference and learn about its ups and downs throughout the past decade. This has given the conference a special vibe!

Constructionism 2016 Committee

Conference Chairs

Paron Israsena – Suksaphattana Foundation, Thailand
Arnan Sipitakiat – Chiang Mai University, Thailand
Nalin Tutiyaphuengprasert – DSIL, Thailand
Gerald Futschek – Vienna University of Technology, Austria

Steering Committee

Ana Isabel Sacristan – Center for Advanced Studies and Research (Cinvestav-IPN), Mexico
Chronis Kynigos – National and Kapodistrian University of Athens, Greece
Ivan Kalas – Comenius University, Slovakia
James Clayson – American University of Paris, France
Paulo Blikstein – Stanford University, USA
Richard Noss – University of London, UK

Reviewers

Ana Isabel Sacristan
Arnan Sipitakiat
Brian Harvey
Carina Girvan
Chronis Kynigos
Cynthia Solomon
Dave Pratt
Edith Ackermann
Enis Castellanos Sánchez
Erich Neuwirth
Evgenia Sendova
Gerald Futschek
Ivan Kalas
James Clayson
Jana Pekarova
Jiří Vaníček
John Dohe
Jose Armando Valente
Karl Fuchs
Ken Kahn
Laura Benton
Lulu Healy
Marta Turcsanyi-Szabo
Michael Tempel
Michael Weigend
Michele Moro
Michelle Wilkerson-Jerde
Nalin Tutiyaphuengprasert
Paulo Blikstein
Pavel Boytchev
Ralf Romeike
Richard Noss
Valentina Munoz-Porras
Wolfgang Slany

Executive Organizers

Paron Israsena
Yuphin Trangkhatharn

Conference Organizers

Nalin Tutiyaphuengprasert
Budsain Chathirunratsamee
Meechai Junpho

Finance

Supichaya Sirisomphobchai

Venue Decoration, Lighting, AV, IT system

Komsan Raxsiri
Kanda Chatwuttikrai
Purin Limnitithum
Panisara Sukmork
Chinapat Mongkholsiriwattana
Sangaroon Cheamsawat
Nantiya Wilairat
Narongsak Yonprawes

Logistic coordinator

Nusarin Nusen
Watchara Thitayanuwat

Catering

Ratchanee Buapechra
Duangruthai Prompichai
Surat Thanprasertkul

Registration

Siriluck Kummuang
Pintippa Sangwan
Hathaithip Maneethipsakul
Wandee Supannaroch
Sarochar Soonkit
Nopporn Yengsakunpaisal
Kaitlyn Johnson
Timothy Michael Lacock
James Lloyd
Amin Dehghan
Nicholas Gallagher
Dale Matthew Griffiths

Public Relations

Porntip Limpichaisopon

Exhibition Coordinator

Surat Thanprasertkul
Chompunuch Jarusrose
Ajchara Techasindhana
Apidon Charoen-agsorn
Ittichai Rattanatavorn
Pintippa Sangwan

Excursion Coordinators

Nalin Tutiyaphuengprasert
Pimpanit Korndee
Ajchara Tachasintana
Soysruang Sansurasilp
Bhinyaporn Pattanasethanon

Translation Coordinators

Chomphunuch Jarusrose
Tawan Tantikul
Bhinyaporn Pattanasethanon
Nattanan Warintarawet
Sawaros Thanapornsangsuth
Pimpanit Condee
Pancheewa Lerkwarunyu
Kukasina Kubaha
Mawin Piamkulvanit

Creative Activity Coordinators

Rossarin Sookpradeam
Nutsma Yuvasopee
Soysruang Sansurasilp
Nimit Ponyiam
Benjaporn Praisorn
Duangruthai Prompichai
Titima Kitcharoen

Program Overview

	Tuesday	Wednesday	Thursday	Friday
9:00	Opening Ceremony Special keynotes Welcome address **Main Auditorium (2fl)**	(6) Plenary 2 **Main Auditorium (2fl)**	(10) Plenary 3 **Main Auditorium (2fl)**	(16) Plenary 4 **Main Auditorium (2fl)**
9:30				
10:00		Break	(11) Conference-wide Brainstorm	Break
10:30	Break	(7A, 7B, 7C) Paper presentations **Auditorium (2fl), Lib (9fl), CLab(10fl)**	Break	(17A, 17B, 17C) Paper presentations **Auditorium (2fl), Lib (9fl), CLab(10fl)**
11:00	(3) Plenary 1 **Main Auditorium (2fl)**		(12) Group discussions	
11:30				
12:00			Lunch	
12:30	Lunch	Lunch		Lunch
13:00			(13) Group Presentations	
13:30	(4) Open house **School Lobby (1 fl)**	(8A, 8B, 8C) Paper presentations **Auditorium (2fl), Lib (9fl), CLab(10fl)**	(14A, 14B, 14C) Paper presentations **Auditorium (2fl), Lib (9fl), CLab(10fl)**	(18A, 18B, 18C) Workshops **CLab(10fl), Auditorium (2fl), Fablab (10fl)**
14:00				
14:30			Break	
15:00		Break	(15A, 15B, 15C) Workshops **Lib (9fl), CLab(10fl), Fablab (10fl)**	
15:30	Break	(9A, 9B, 9C, 9D) Workshops **CLab(10fl), Auditorium (2fl), Lib (9fl), Fablab (10fl)**		
16:00	(5A, 5B) Poster & Demos **Gymnasium (2 fl)**			Break
16:30				(19) Plenary 5 **Main Auditorium (2fl)**
17:00				
17:30				Closing
18:00				
Evening	Dinner at Mr. Paron's House		Conference Dinner	

Conference Venue Directory

The conference will be held at DSIL building. (DSIL stands for Darunsikkhalai School for Innovative Learning.) It is the 14 floors building with all facilities located inside the building. All the program will be conduct at different floors. Here is some directory for presentation location as well as some interesting activities arranged by DSIL students, teachers and parents throughout the conference week. Please enjoy your time and make yourself comfortable. If you need help, have any questions or need translation for your interesting conversation, please don't hesitate to ask our staffs or rescuers team. ☺ Welcome to Bangkok, Thailand!

Floor	Room Number	Activities
10th	Constructionism Lab 1 and 2	Paper presentation Room 3 Demos and workshops
10th	DSIL FabLab@School	Workshop Room 4 Feel free to check out our FabLab.
9th	Library	Paper presentation Room 2 Demos and workshops
8th	Classrooms	Workshops and Open house
7th	Classrooms	
6th	Classrooms	
5th	Cafeteria and Food lab	Lunch and Thai arts and crafts activities Small futsal field with panoramic view of Bangmod area.
4th	Fitness and Performing art classroom	Not opened during conference week.
2nd	Main Auditorium	Plenary session Opening and closing
2nd	Gymnasium and Open Space	Tea and coffee break and Poster session * CCTV for plenary session
1st	DSIL Hall	Registration Thailand Constructionism Expo

Directory for Thailand Constructionism Expo
(Tuesday 2nd February, 2016.)

Schedule

13:30-13:50 Introduction to Lighthouse Project and Candle Light Project (Nalin)

13:50-14:05 There will be just one round of approximately an hour presentation in each area. You may choose to stay in one area or move around. There are 3 different groups of presentations.

14:05 – 15:00 Demo / Presentation

15:00- 15:30 Q&A and Discussion

Groups	Moderator	Presentation / Demo and Location
Group A : Schools that learn Different application of Constructionism in different school contexts throughout Thailand.	Nalin Tutiyaphuengprasert 14:05 -15:00 Demo / Presentation in classrooms 15:00-15:30 Q&A Discussion	**6th – 8th floor** -DSIL Science classroom, Biology Lab (6th fl.) - Engineering Project (7th-8th grade) Room805-806 - Young Constructionist (1st and 2nd grades) Room706 - Ban Sankampang School in Chiang Mai Room 707 - Ban Kha Yang School (Hill tribe school) Room 707 - Wat Kien Kate Room 707 - Koh Yao School District Room 707
Group B: Villages that learn Constructionism applied in informal learning to improve quality of life and sustainable development.	Arnan Sipitakiat 14:05 -15:00 Booth presentation 15:00-15:30 Q&A Discussion at the main stage	**Booth on 1st floor** • Ban Lim Thong Constructionist Village from Buriram Province • Ban Samkha Constructionist Village from Lampang Province
Group C: Business and Industries that learn Constructionism applied in professional development programs in different business units which reduce cost and increase revenue. Quality improvement by empowering innovative culture in organization.	Tawan Tantikul 14:05 -15:00 Booth presentation 15:00-15:30 Q&A Discussion at the committee's meeting room	**Booth on 1st floor** • Siam Cement Groups Constructionist Practice Schools (All business units) • Petrochemical Industry Constructionist Practice School • Mabtaput vocational school

Conference Program

Tuesday, February 2nd

09:00 - 09:15 Opening Ceremony

LOCATION: Main Auditorium (2 fl)

09:15 - 09:45 Session 1: Special Keynote

Yongyuth Yuthavong

Sparks from the Spirit: Role of Learning in Constructing and Using Knowledge

09:45 - 10:00 Welcome Address & Conference Introduction

10:00 - 10:30 Session 2: Opening Keynote

Location: Main Auditorium (2 fl)

Paron Israsena

Constructionism the Thailand Way

10:30 - 11:00 Break Location: 2nd fl open space

11:00 - 12:30 Session 3: Plenary 1

Chair: Arnan Sipitakiat
Location: Main Auditorium (2 fl)

11:00 Nalin Tutiyaphuengprasert and Yuphin Trangkatarn

Darunsikkhailai and its 15 years improvisation on Samba School

11:45 Leda Munoz and Maria Eugenia Bujanda

Costa Rica's Omar Dengo Foundation Program: 29 years later

12:30 - 13:30 Lunch

Location: Cafeteria (5 fl)

13:30 - 15:30 Session 4: Open house: Constructionism in Thailand

Chair: Nalin Tutiyaphuengprasert
Location: School Lobby (1 fl)

15:30 - 15:45 Break Location: 2nd fl open space

15:45 - 17:45 Session 5A: Poster Session

Chairs: Arnan Sipitakiat and Nalin Tutiyaphuengprasert
Location: Gymnasium (2 fl)

Sawaros Thanapornsangsuth

Compassion and Empathy through Inventions: GoGo Board Toolkit for 7 - 10 years old

Brian Harvey

What's New in Snap! and BJC

Pavel Petrovič

A new robot in a classroom

Nathan Holbert

Bots for Tots: Leveraging ‹Ways of Knowing› to Increase Diversity in Makerspaces

Tomohito Yashiro, Kazushi Mukaiyama and Yasushi Harada

Workshop of Game Programming in Scratch

Sawaros Thanapornsangsuth, Yongyuth Laitavorn, Kasidej Phulsuksombati, U-Lacha Laochai, Rachaneeporn Assavanop, Surapat Somsri and Quankamon Dejatiwongse Na Ayudhya

Little Builders: Empowering At-risk Children by Building and Design

Miki Matsumasa, Chieko Harayama, Kazuhiro Abe, Nobuko Kishi and Manabu Sugiura

Robot Programming Workshop for Middle and High School Girls

Alejandro Rosas Mendoza, Esteban Pablo Díaz and Avenilde Romo Vázquez

Bridge of popsicle sticks: A project and a contest

Aoi Yoshida, Kazunari Ito and Kazuhiro Abe

A practical report on a course of learning by making at the university in Japan

Mícheál Ó Dúill

Philological philosophy, simplistic science, misleading mathematics, perfidious perception, trusty technology

Mícheál Ó Dúill

After Scratch: Logo(Writer)?

Pavel Petrovič

On controlling LEGO Education platforms from Imagine Logo

Ildikó Tasnádi, László Csink and Károly Farkas

Teaching programming constructively and playfully

Suttipong Thajchayapong, Tanut Choksatchawathi, Paron Israsena and Chanikarn Wongviriyawong

Increasing Learning Gain in Linear Algebra through Play

15:45 - 17:45 Session 5B: Demos

Location: Gymnasium (2 fl)

Pavel Petrovič

Tatrabot - a mobile robotic platform for teaching programming

Yu Guo and Uri Wilensky

Learning About Complex Systems with the BeeSmart Participatory Simulation

Mikhaela Dietch, Bryanne Leeming and Amon Millner

JumpSmart: A Platform for Communal Making and Physical Engagement in Programming

Kazuhiro Abe and Masashi Umezawa

Pyonkee: A Scratch compatible visual-programming environment running on iPad

Andreas Grillenberger and Ralf Romeike

Analyzing Twitter Data using Snap!

19:00 - 21:30 Dinner at Mr. Paron's house

Wednesday, February 3rd

09:00 - 10:00 Session 6: Plenary 2

Chair: Arnan Sipitakiat
Location: Main Auditorium (2 fl)

Ivan Kalas
On the Road to Sustainable Primary Programming

10:00 - 10:30 Break Location: 2nd fl open space

10:30 - 12:30 Session 7A: Papers: Programming in the 21st Century

Chair: Brian Harvey
Location: Main Auditorium (2 fl)

10:30 Chris Proctor and Paulo Blikstein
Grounding How We Teach Programming in Why We Teach Programming

11:00 Ken Kahn
What would the ideal constructionist programming language (or languages) be like?

11:45 Marcelo Worsley, Kipp Bradford, Taylor Martin, Paulo Blikstein, Arnan Sipitakiat and Nalin Tutiyaphuengprasert
Constructionism and the Internet of Things

10:30 - 12:30 Session 7B: Papers: Programming in Context

Chair: Ana Isabel Sacristan
Location: Library (9 fl)

10:30 Anja Petri, Christian Schindler, Wolfgang Slany and Bernadette Spieler
Game Design with Pocket Code: Providing a Constructionist Environment for Girls in the School Context

11:00 Walter Bender, Devin Ulibarri and Yash Khandelwal
Music Blocks: A Musical Microworld

11:30 Deborah Fields, Lisa Quirke, Tori Horton, Jason Maughan, Xavier Velasquez, Janell Amely and Katarina Pantic
Working Toward Equity in a Constructionist Scratch Camp: Lessons Learned in Applying a Studio Design Model

12:00 Michael Weigend
Designing Interfaces for Special Needs

10:30 - 12:30 Session 7C: Papers: Constructionism & New Ideas

Chair: Gerald Futschek
Location: Constructionism Lab (10 fl)

10:30 Jean-Francois Maheux
Papert's sort-of-right mathematics

11:00 Kate Mackrell and Dave Pratt
Resituating Constructionism in the Space of Reasons

11:30 Valentina Dagienė, Gerald Futschek and Gabrielė Stupurienė
Teachers' Constructionist and Deconstructionist Learning by Creating Bebras Tasks

12:00 Meurig Beynon, Jonathon Foss, Antony Harfield, Elizabeth Hudnott and Nicolas Pope
Construing and Computing: Learning through Exploring and Exploiting Agency

12:30 - 13:30 Lunch & Scientific Committee Meeting #1

Location: Cafeteria (5 fl)

13:30 - 15:15 Session 8A: Papers: Case Studies & New Approaches

Chair: James Clayson
Location: Main Auditorium (2 fl)

13:30 Tamar Fuhrmann, Marcelo Worsley and Paulo Blikstein
A new Model for Eliciting Engineering Expertise from Novices: Expect non-Experts to Behave Like Experts

14:00 Yoshiro Yoshiro, Nanako Ishido, Kazuhiro Abe and Mihoko Kamei
Communities of Learning Designers in Japan – From constructing products to constructing communities –

14:45 Umit Aslan and Uri Wilensky
Restructuration in Practice: Challenging a Pop-Culture Evolutionary Theory through Agent Based Modeling

13:30 - 15:15 Session 8B: Papers: Reflections & Next Steps for Constructionism

Chair: Ivan Kalaš
Location: Library (9 fl)

13:30 Chronis Kynigos
Constructionist activity with institutionalized infrastructures: the case of Dimitris and his students.

14:00 Laura Benton, Celia Hoyles, Ivan Kalas and Richard Noss
Building mathematical knowledge with programming: insights from the ScratchMaths project

14:30 Gary Stager, Tracy Rudzitis, Brian Smith and Amy Dugré
Making Constructionism Real in Real Schools Every Day

13:30 - 15:15 Session 8C: Papers: School Experiences

Chair: Gerald Futschek
Location: Constructionism Lab (10 fl)

13:30 Cristianne Butto-Zarzar, Joaquín Delgado and Ana Isabel Sacristán
Generalization processes: an experience using eXpresser with primary-school children

14:00 Antony Harfield, Rene Alimisi, Peter Tomcsányi, Nick Pope and Meurig Beynon
Constructionism as making construals: first steps with JS-Eden in the classroom

14:30 Conor Wickham, Carina Girvan and Brendan Tangney
Constructionism and microworlds as part of a 21st century learning activity to impact student engagement and confidence in physics

15:15 - 15:45 Break Location: 2nd & 9th fl open space

15:45 - 18:15 Session 9A: workshop

Location: Constructionism Lab (10 fl)

Valentina Dagiene, Gerald Futschek and Gabrielė Stupurienė
Developing Computational Thinking by Using Constructionist and Deconstructionist Learning

15:45 - 18:15 Session 9B: workshop

Location: Main Auditorium (2 fl)

Yoshiro Miyata and Mihoko Kamei
World Peace Song Project

15:45 - 18:15 Session 9C: workshop

Location: Library (9 fl)

Walter Bender, Cynthia Solomon, Devin Ulibarri and Claudia Urrea
Music Blocks Workshop

15:45 - 18:15 Session 9D: workshop

Location: Fab Lab (10 fl)

Gary Stager
Constructionist Archaeology - Digging into Papert Papers Lost and Found

Thursday, February 4th

09:00 - 10:00 Session 10: Plenary 3

Location: Main Auditorium (2 fl)

Brian Harvey
Report from the Future: The Next Generation High School

10:00 - 10:45 Session 11: Conference-wide brainstorming:
"Where we are? What is next?"

Chair: Paulo Blikstein
Location: Main Auditorium (2 fl)

10:45 - 11:00 Break Location: 2nd fl open space

11:00 - 12:15 Session 12: Group discussion based on themes that emerged from the brainstorming session.

Chair: Paulo Blikstein
Location: Main Auditorium (2 fl)

12:15 - 13:00 Lunch Location: Cafeteria (5 fl)

13:00 - 13:45 Session 13: Presentation by all three groups.

Chair: Paulo Blikstein
Location: Main Auditorium (2 fl)

13:45 - 14:45 Session 14A: Papers: Early Childhood

Chair: Richard Noss
Location: Main Auditorium (2 fl)

13:45 Jose A. Valente
Inquiry Based Learning Project: a study of doing science with elementary public school students

13:45 - 14:45 Session 14B: Papers: New Tools & Activities

Chair: Arnan Sipitakiat
Location: Library (9 fl)

13:45 Corey Brady, David Weintrop, Gabriella Anton and Uri Wilensky
Constructionist Learning at the Group Level with Programmable Badges

14:15 Ken Kahn
Integrating programming languages with web browsers

13:45 - 14:45 Session 14C: Papers: Working with Teachers

Chair: Ana Isabel Sacristan
Location: Constructionism Lab (10 fl)

13:45 Mareen Przybylla and Ralf Romeike
Teaching Computer Science Teachers: A Constructionist Approach to Professional Training on Physical Computing

14:15 Elena Prieto-Rodriguez and Daniel Hickmott
Preparing teachers for the Digital Technologies Curriculum: Preliminary results of a pilot study

14:45 - 15:00 Break Location: 2nd & 9th fl open space

15:00 - 17:30 Session 15A: workshop

Location: Library (9 fl)

Carina Girvan, Nathan Holbert, Chronis Kynigos, Celia Hoyles and Richard Noss
Considering approaches to research through the lens of constructionism

15:00 - 17:30 Session 15B: workshop

Location: Constructionism Lab (10 fl)

Arthur Hjorth, David Weintrop, Corey Brady and Uri Wilensky
LevelSpace: Constructing Models and Explanations across Levels

15:00 - 17:30 Session 15C: workshop

Location: Fab Lab (10 fl)

Arnan Sipitakiat and Paulo Blikstein
New Frontiers in Educational Robotics with the Raspberry Pi and the GoGo Board

18:30 - 20:30 Sunset excursion and conference dinner on a private cruise along the Chao Phraya river.

Friday, February 5th

09:00 - 10:00 Session 16: Plenary 4

Chair: Richard Noss
Location: Main Auditorium (2 fl)

Cynthia Solomon
Constructionism Lifetime Achievement Award

10:00 - 10:30 Break Location: 2nd fl open space

10:30 - 12:30 Session 17A: Papers: Math, Programming, Robots

Chair: Arnan Sipitakiat
Location: Main Auditorium (2 fl)

10:30	Pavel Petrovič and Richard Balogh **Summer League: Supporting FLL Competition**
11:00	Dave Catlin **Learning Intentions and Educational Robots**
11:30	Han Hyuk Cho, Jin Hwan Jeong, Jong Jin Kim, Yong Hyun Seo and Seung Joo Lee **Math-based Coding Education in Korean School**
12:00	Han Hyuk Cho, Hanik Jo, Cho Hee Lee, Eun Ji Lee and Hye Rim Jeong **3D turtle coding activities for Korean primary education**

10:30 - 12:30 Session 17B: Papers: Emotion & Engagement

Chair: James Clayson
Location: Library (9 fl)

10:30	Chris Shelton **Beyond lesson recipes: first steps towards a repertoire for teaching primary computing**
11:00	Ploybussara Gomasang, Pintippa Sangwan and Chanikarn Wongviriyawong **Assessing Interpersonal Relationship Among Peers in a Constructionist Classroom: a Probabilistic Method**
11:30	Michael Weigend, Lisa-Marie Jung, Sarah Lenzen, Maike Gebhardt, Caroline Flaßhoff, Alisha-Sophie Unger, Julian Wagner, Weronika Jarosz, Stella Krämer and Stefanie Schweitzer **Learning Emotional Aspects of Digital Competence By Creating Artefacts**
12:00	Nalin Tutiyaphuengprasert **Samba School of the 21st Century : Learning in the Break Dance Community in Bangkok**

10:30 - 12:30 Session 17C: Papers: Tools & Powerful Ideas

Chair: Ken Kahn
Location: Constructionism Lab (10 fl)

10:30 Maite Mascaró and Ana Isabel Sacristán
Exploring randomness and variability in statistics through R-based programming tasks

11:00 Angel Pretelín-Ricárdez and Ana Isabel Sacristán
Programming videogames with models of physical parameters: some examples

11:30 Eleonora Badilla-Saxe and Florencia Morado
The Imaginatorium

12:00 Arthur Hjorth, Corey Brady, Bryan Head and Uri Wilensky
Turtles All the Way Down: Presenting LevelSpace, a NetLogo Extension for Reasoning about Complex Connectedness

12:30 - 13:30 Lunch & Scientific Committee Meeting #2

Location: Cafeteria (5 fl)

13:30 - 16:00 Session 18A: workshop

Location: Constructionism Lab (10 fl)

David Weintrop, Arthur Hjorth, Corey Brady and Uri Wilensky
NetLogo Web: Bringing Turtles to the Cloud

13:30 - 16:00 Session 18B: workshop

Location: Main Auditorium (2 fl)

Ken Kahn
And now for something completely different: ToonTalk - a programming language that is not textual, block-based, or procedural

13:30 - 16:00 Session 18C: workshop

Location: Fab Lab (10 fl)

Angel Pretelín-Ricárdez and Ana Isabel Sacristán
Videogame construction with models of physical parameters

16:00 - 16:30 Break Location: 2nd fl open space

16:30 - 17:30 Session 19: Plenary 5

Chair: Jose Valente
Location: Main Auditorium (2 fl)

Jose A. Valente

Computational Thinking in School: reviving programming in the context of digital culture

17:30 - 17:45 Closing

Location: Main Auditorium (2 fl)

Table of Contents

3D turtle coding activities for Korean primary education

Han Hyuk Cho, *hancho@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Hanik Jo, *1990434@hanyang.ac.kr*
Dept of Education, Hanyang University, Seoul 04763, Korea

Cho Hee Lee, *choheequeen@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Eun Ji Lee, *eunjii@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Hye Rim Jeong, *rim6458@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Abstract

The recently revised national curriculum for primary school in Korea inserted constructive activities with "linking cubes" for 6-th grade math class to improve students' spatial abilities. In addition, with the worldwide trend to regard coding education as great important, the Korea Ministry of Education announced that the class for coding be mandatory at primary schools after 2018. However, due to the absence of appropriate expressive systems for the shape of 3D cube stacks and the paucity of coding educational tools for primary school students, the schools will be suffered from a lot of difficulties to teach it.

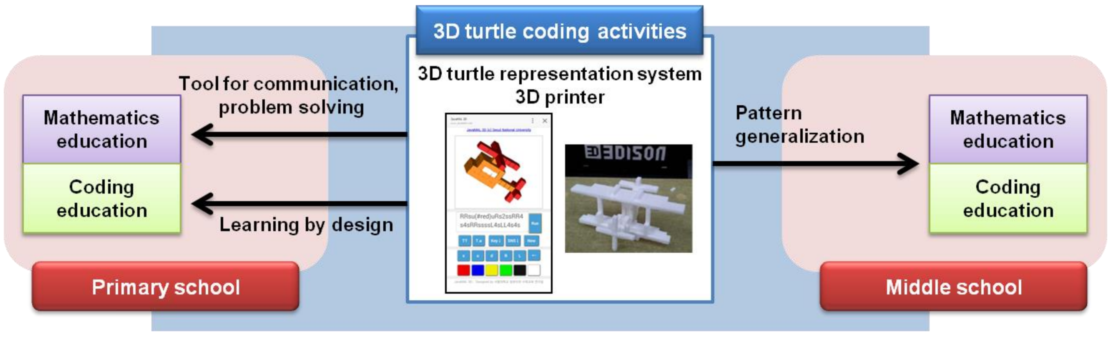


Figure 1. 3D turtle coding activity

As presented in Figure1, 3D coding activities with the 3D turtle representation system and the 3D printer were designed to present the solution to overcome the hindrance. Moreover, the 3D turtle coding activities are expected to play the role as a bridge between primary schools and middle schools in both mathematics and coding education, which will make the students possible to enhance spatial abilities in linked curricula.

Keywords

3D turtle representation system, turtle symbol, 3D printer, 3D turtle coding activity, powerful idea, spatial ability, mathematics education, coding education

1. Introduction

Logo is a learning environment where one can think and learn about abstract phenomena as a cognitive tool for constructing geometric objects. Based on the basic ideas of Logo, Cho et al. (2010) designed the 3D turtle representation system. In this representation system, the turtle moves stacking 3D cube in a Logo-based microworld. The 3D turtle representation system consists of turtle symbol such as s (moving forward), l/r (moving left/right), and u/d (moving upward/downward). An example of constructing the two types of 3D cube stacks with these turtle symbols is shown in Figure2.

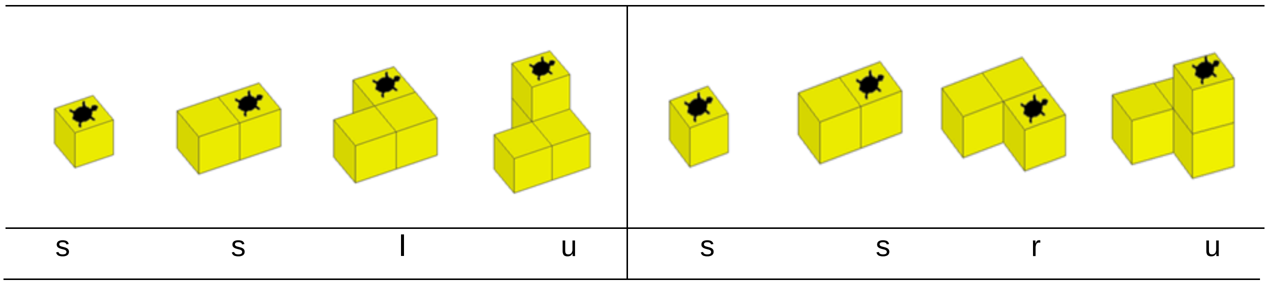
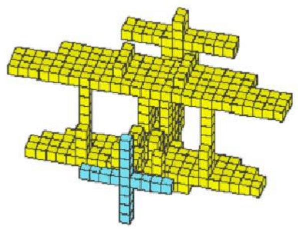
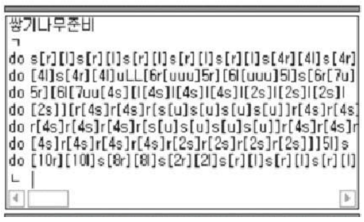
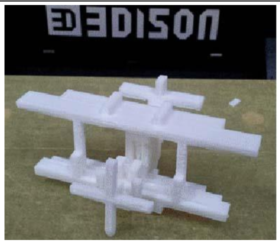


Figure 2. The 3D turtle representation system

Cho et al. (2012) had designed a creativity contest for students where their artifacts can be designed, expressed, and manipulated by using the 3D turtle representation system, thus leading to learning through design. The advent of the 3D printer made it possible to produce a 3D solid object which is designed and expressed by the turtle representation system in virtual space (Cho et al., 2014a). Figure3 shows an airplane created in virtual space with the representation system and its printout by using a 3D printer. The learning environment built into the representation system not only engages learners in constructing artifacts, but also encourages them to explore the ideas underlying their constructions in both virtual and physical worlds. The above-mentioned studies have elucidated the role of 3D turtle representation system as a tool not only for the expression of 3D objects and communication but also for learning mathematics(Cho et al., 2010; Cho et al., 2012; Cho et al., 2014a; Cho et al., 2014b).

```
쌓기나무준비
ㄱ
do s[r][l]s[r][l]s[r][l]s[r][l]s[r][l]s[4r][4l]s[4r]
do [4l]s[4r][4l]uLL[6r[uuu]5r][6l[uuu]5l]s[6r[7u]
do 5r][6l[7uu[4s][l[4s]l[4s]l[4s]l[2s]l[2s]l[2s]l
do [2s]][r[4s]r[4s]r[s[u]s[u]s[u]s[u]]r[4s]r[4s]
do r[4s]r[4s]r[4s]r[s[u]s[u]s[u]s[u]]r[4s]r[4s]r
do [4s]r[4s]r[4s]r[4s]r[2s]r[2s]r[2s]r[2s]]]5l]s
do [10r][10l]s[8r][8l]s[2r][2l]s[r][l]s[r][l]s[r][l]
ㄴ
```



airplane made in virtual world by using turtle representation | airplane printed by 3D printer

Figure 3. Airplane made in both virtual and real (Cho et al., 2014a)

The recently revised national curriculum for primary school in Korea inserted constructive activities with "linking cubes" for 6-th grade math class to improve students' spatial abilities. In addition, with the worldwide trend to regard coding education as great important, the Korea Ministry of Education announced that the class for coding be mandatory at primary schools after 2018. However, due to the absence of appropriate expressive systems for the shape of 3D cube stacks and the paucity of coding educational tools for primary school students, the schools will be suffered from a lot of difficulties to teach it.

The present researchers designed 3D coding activities with the 3D turtle representation system and the 3D printer to solve the difficulties. Moreover, the 3D turtle coding activities are expected to play the role as a bridge between primary schools and middle schools in both mathematics and coding education, which will make the students possible to enhance spatial abilities in linked curricula.

2. 3D turtle coding activities for Korean mathematics education

As presented in Figure4, the 3D cube stacks chapter of the Korean primary school 6th grade mathematics curriculum were added to the constructive activities with “linking cubes” recently. The activities aim at developing students’ spatial abilities by manipulating linking cubes on students’ own.

Which one will look like A when you turn it around or turn it upside down?

If you put B and C together, will they become D_1 (or D_2)?

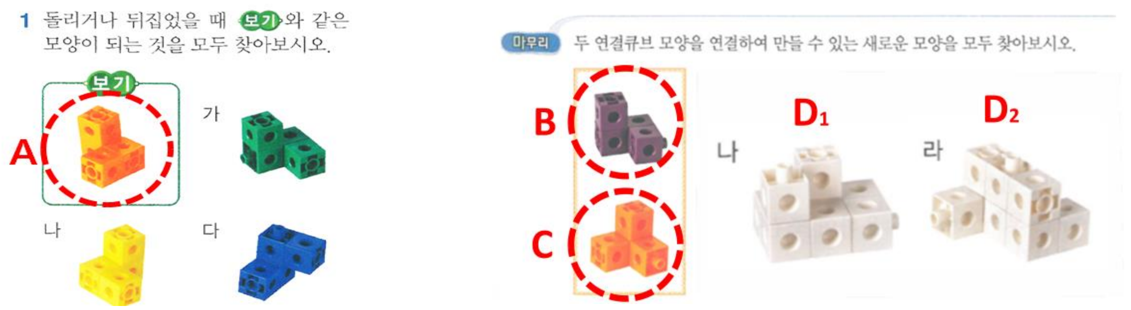

1 돌리거나 뒤집었을 때 보기와 같은 모양이 되는 것을 모두 찾아보시오.
보기
A
가
나
다
마무리 두 연결큐브 모양을 연결하여 만들 수 있는 새로운 모양을 모두 찾아보시오.
B
C
D1
D2
나
라

Figure 4. Contents of the revised mathematics textbook for 6th graders at Korean primary schools

However, there are some practical and physical limitations when using linking cubes in class due to their expense, weight and volume. What is more, there are no appropriate expressive systems for the shape of cube stacks. Consequently, students and teachers are expected to experience considerable difficulty in communication and problem solving.

To solve the problem on the right in Figure4, for example, it is necessary to eliminate C from D_1 (or D_2) and to check whether the remaining piece is identical to B. It is linked to the problem on the left in Figure4. The problems in Figure4 are related to mental rotation, where the subject is asked to compare two 3D objects rotated in some axis and stated if they are the same image of if they are mirror images. Mental rotation of 3D objects is shown in Figure5.

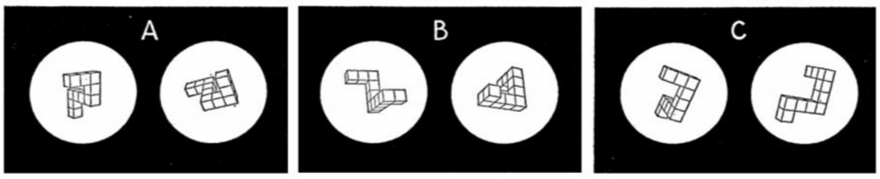


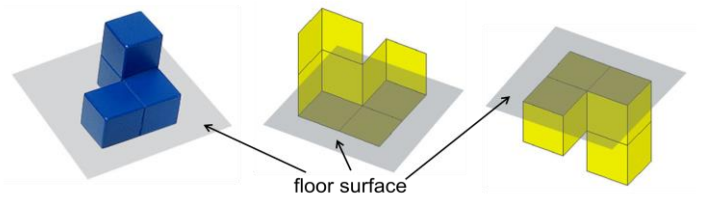
Figure 5. Mental rotation of 3D object (Shepard & Metzler, 1971)

Consequently, the present researchers designed and applied 3D turtle coding activities based on a study by Cho, et al. (2014) and with the two following principles. These activities were based on the 3D turtle representation system and 3D printers.

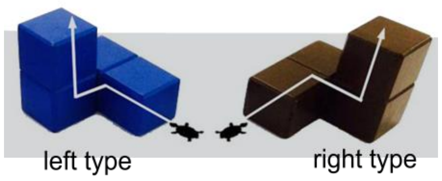
- **Principle 1.** Mentally constructing the floor surface of a 3D object by using a 3D printer metaphor.
- **Principle 2.** Coding 3D cube stacks as a sequential process from the perspective of turtle standing on the floor surface of a 3D object.

In other words, students were prompted mentally to construct the floor surface in virtual space and to look at 3D cube stacks presented according to diverse axes and angles, as on the left in Figure6. To achieve this, the present researchers introduced 3D printers, which started from the floor surface, piled layers on it, and printed out 3D objects. In addition, students were prompted to look at 3D cube stacks not as products but sequentially, from the perspective of the turtle based on the 3D turtle representation system.

Based on such 3D turtle coding activities, students were prompted to distinguish between the types of 3D objects, as on the right in Figure6. This not only makes it possible to communicate about 3D cube stacks but will aid problem solving in the “linking cube” activities.



Floor surfaces of 3D cube stacks in both real and virtual world



Left and right type in a Soma cube

Figure 6. Floor surfaces of 3D cube stacks and type of blocks.

To show the effects of the 3D turtle coding activities in mathematics education, pre and post tests were conducted based on the right of Figure4. All participants were pre and post-tested before and after activities. The two tests included the same items, which were revised and supplemented versions of a study by Lee (2015). In the tests, students were presented with a 3D object C consisting of B and a soma cube piece put together and to determine whether, after the removal of the piece B, the remaining 3D object was L or R, as in Figure7. Here, they were to make a distinction under the restriction of the left type (L) and the right type (R) of the soma cube.

Deciding whether, after the removal of the piece B from C, the remaining 3D object is L or R.

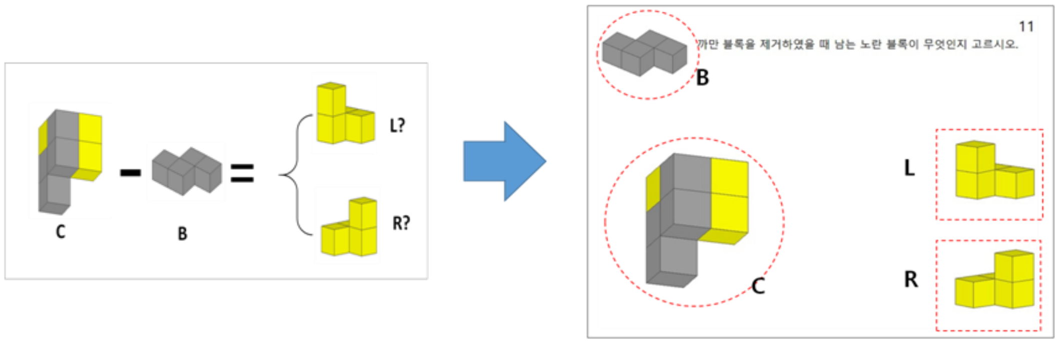

C
B
L?
R?
11
까만 블록을 제거하였을 때 남는 노란 블록이 무엇인지 고르시오.
B
C
L
R

Figure 7. The test item

The 23 male students in the first and second years at middle schools participated in the tests; however, the two students which did not take the post tests were excluded, resulting in the total data of 21 students. The tests were conducted on personal tablet computers, and students' answers and response time per question were recorded. Students were requested to solve a total of 24 test items, with a time limit of 20 seconds per question. When the data for 21 students were analyzed, from pre to post tests, students' average scores increased significantly (z = -2.56, p = .025, N = 21; Wilcoxon signed rank test) and average response time decreased significantly (z = -4.02, p < .05, N = 21; Wilcoxon signed rank test), respectively, as presented in Figure8.

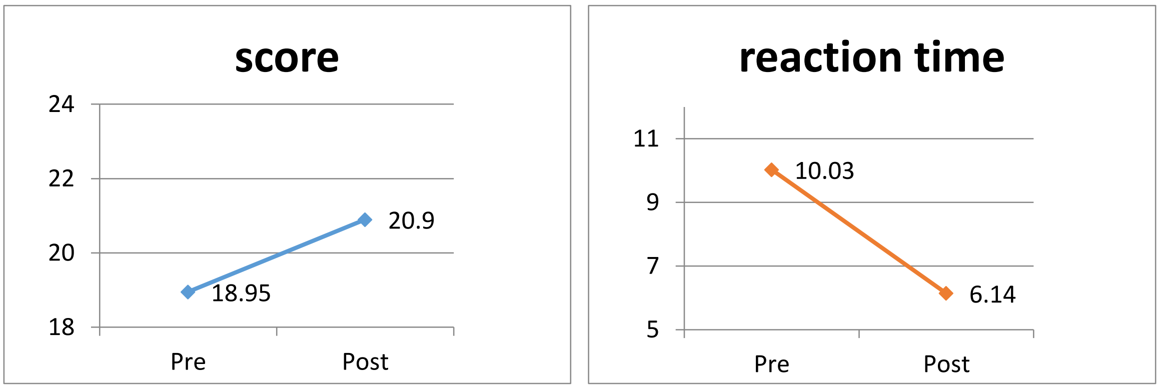


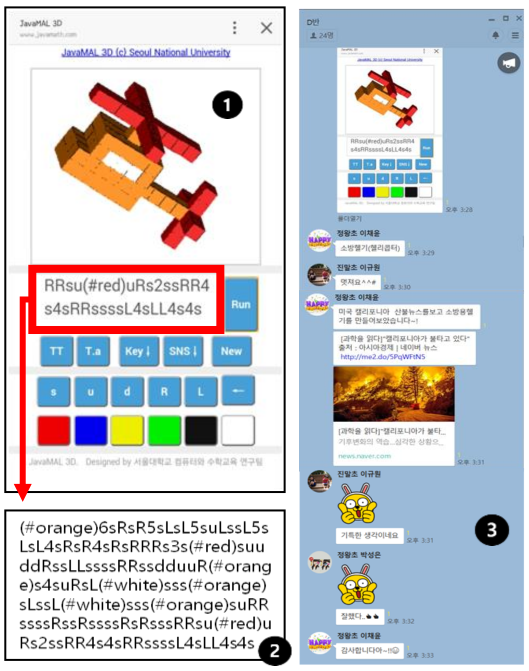
Figure 8. Test result

After the application of 3D turtle coding activities, students felt that they had taken the tests more easily and accurately. As a result, the present researchers claim that the idea of 3D turtle coding activities help students to communicate about 3D cube stacks effectively and solve the spatial tasks precisely. Through the activities, it will be possible for the idea of 3D turtle coding activities to provide mathematics education at primary schools with a powerful tool for communication and problem solving.

3. 3D turtle coding activities for Korean coding education

With the worldwide trend to regard coding education as great important, the Korea Ministry of Education announced that the class for coding be mandatory at primary schools after 2018. However, due to the paucity of coding educational tools for primary school students, the schools will be suffered from a lot of difficulties to teach it. Therefore the present researchers suggest a powerful tool for coding education at primary schools by providing a learning environment where students design personally meaningful artifacts in a smartphone environment through 3D turtle coding activities.

Resnick et al. (2005) has stated that best learning experiences come when people are actively engaged in designing and creating things, especially things that are meaningful to them or others around them. Figure9 shows the artifact that a student designed in a smartphone environment through 3D turtle coding activities and shared and communicated through social media by each other. ① shows the artifact called a "firefighting helocopter". A student designed the artifact in a smartphone environment after seeing news of a forest fire in California recently. ② shows the turtle symbol to constuct this artifact. ③ shows comments such as "It's cool," "What a wonderful thought," and "Well done" posted by other students after the creator of the artifact posted the link to the Internet news on the forest fire in California through social media.


JavaMAL 3D
www.javamath.com
JavaMAL 3D (c) Seoul National University
1
Run
TT
T.a
Key↓
SNS↓
New
s
u
d
R
L
←
JavaMAL 3D. Designed by 서울대학교 컴퓨터와 수학교육 연구팀
2
3

```
RRsu(#red)uRs2ssRR4
s4sRRssssL4sLL4s4s
```

```
(#orange)6sRsR5sLsL5suLssL5s
LsL4sRsR4sRsRRRs3s(#red)suu
ddRssLLssssRRssdduuR(#orang
e)s4suRsL(#white)sss(#orange)
sLssL(#white)sss(#orange)suRR
ssssRssRssssRsRsssRRsu(#red)u
Rs2ssRR4s4sRRssssL4sLL4s4s
```

풀더열기

정왕초 이채윤: 소방헬기(헬리콥터) 오후 3:29

진말초 이규원: 멋져요^^# 오후 3:30

정왕초 이채윤: 미국 캘리포니아 산불뉴스를보고 소방용헬기를 만들어보았습니다~!

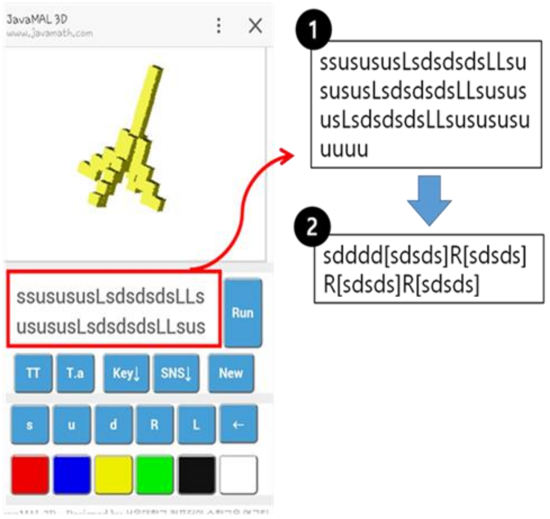
[과학을 읽다]"캘리포니아가 불타고 있다"
출처 : 아시아경제 | 네이버 뉴스
http://me2.do/5PqWFtN5

[과학을 읽다]"캘리포니아가 불타...
기후변화의 역습...심각한 상황으...
news.naver.com 오후 3:31

진말초 이규원: 기특한 생각이네요 오후 3:31

정왕초 박성은: 잘했다.. 오후 3:32

정왕초 이채윤: 감사합니다아~!! 오후 3:33



```
ssusususLsdsdsdsLLs
usususLsdsdsdsLLsus
```

```
ssusususLsdsdsdsLLsu
sususLsdsdsdsLLsusus
usLsdsdsdsLLsusSo sususu
uuuu
```

```
sdddd[sdsds]R[sdsds]
R[sdsds]R[sdsds]
```

Figure 9. Communication through design activity

Figure 10. Learning by design

As another example, Figure10 shows an "Eiffel Towel" created by another student in a smartphone environment. At first, this student constructed the artifcat using the turtle symbol such as ①. However, the student felt inconvenience because of the step for turtle to go and come back to a certain point. Subsequently informed of the "[ ]" symbol, this student revise commands by using it, which are presented in ②. Like the repeat sign in music, turtle remembers its position and direction in '[', takes actions according to the given commands between '[' and ']' and comes back to '[' when ']' is commanded. After revising commnands, this student said the usefulness of the "[ ]" symbol: "I think the [ ] symbol is useful because you can create shapes without having to return to the certain point. It's fascinating :)"

By providing students with a learning environment where they could design artifacts, the present researchers enabled them to feel the power of the turtle symbol and to experience learning by design, where knowledge is mentally constructed in the design process. In addition, students also shared and communicated their work through social media in real time. Through this, it will be possible for design activities based on 3D turtle coding activities to provide coding education at primary schools with a powerful tool.

Closing remark

Papert (1980) has stressed the importance of “powerful ideas” that can be used as tools to think with over a lifetime. Through 3D turtle coding activities based on the powerful idea of 3D turtle representation system, the present researchers provided not only a solution to difficulty in mathematics and coding education at primary schools but also ideas with leverage to students. The present researchers confirmed the possibility of expanding the cognitive skill of spatial ability by prompting students to take a structural and systematic approach to 3D cube stacks through 3D turtle coding activities. Furthermore, these design activities experienced at primary schools will provide a bridge for seeing 3D objects as patterns, as in Figure11, thus leading to the exploration of pattern generalization including substitution process with variables. Also, 3D turtle coding activities are expected to play the role as a bridge between primary schools and middle schools in both mathematics and coding education, which will make the students possible to enhance spatial abilities in linked curricula.

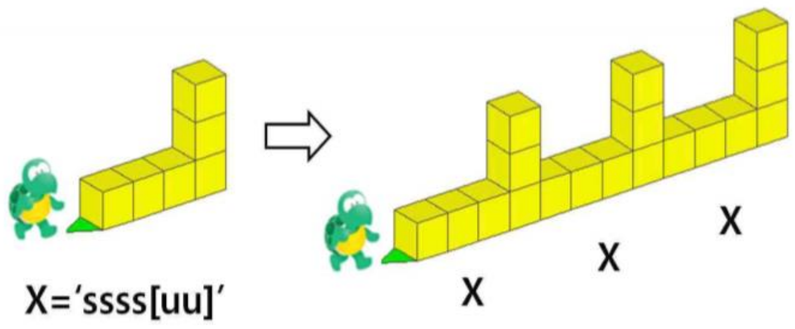


Figure 11. Substitution (Cho et al., 2014b)

References

Cho, H. H., Kim, H. K., Song, M. H. & Lee, J. Y. (2010). *Representation systems of building blocks in Logo-based microworld.* The Constructionism 2010 Conference. Paris: AUP.

Cho, H. H., Lee, J. Y. & Song, M. H. (2012). *Construction and design activities through Logo-based 3D microworld.* The Constructionism 2012 Conference. Athens, Greece.

Cho, H. H., Lee, J. Y. (2014a). *3D Turtle representation system and mental rotation using 3D turtle perspective.* The Constructionism 2014 Conference. Vienna, Austria.

Cho, H. H., Song, M. H. (2014b). *On the SMART Storytelling Mathematics Education Based on Executable Expression.* Journal of Educational Research in Mathematics: Research in Mathematical Education, 24(2), 269-283.

Lee, Jiyoon. (2015). *Type and Role of Spatial Cognition Strategies in 3D Object Discrimination Tasks: Focusing on Embodied 3D Turtle Expression and Turtle Strategy.* Ph. D. Dissertation, Seoul National University, Seoul, Korea.

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas.* Cambridge, Massachusetts: Perseus Publishing.

Resnick, M. & Silverman, B. (2005). *Some reflections on designing construction kits for kids*. Proceeding of interaction design and children conference, Boulder, Co.

Shepard, R. N. & Metzler, J. (1971). Mental Rotation of Three-Dimensional Objects. Science, 171, 701-703.

Assessing Interpersonal Relationship Among Peers in a Constructionist Classroom: a Probabilistic Method

Ploybussara Gomasang, *bussara.bussee@mail.kmutt.ac.th*
Dept of Electronic and Telecommunication Engineering, King Mongkut's University of Technology Thonburi

Pintippa Sangwan, *pintippa32@gmail.com*
Darunsikkhalai School for Innovative Learning, King Mongkut's University of Technology Thonburi

Chanikarn Wongviriyawong, *chanikarn.won@kmutt.ac.th*
Institute of Field Robotics, King Mongkut's University of Technology Thonburi

Abstract

This paper focuses on learner's interpersonal relationship with their peers in classrooms, and proposes one approach to probabilistically assess some aspects of learner's interpersonal relationship using minimal information, which can complement other approaches. We collected the data in a constructionist classroom of 8 students in Grade 2 at Darunsikkhalai School for Innovative Learning in Bangkok, Thailand. Total time spent with peers and the number of peers surrounding a student whose interpersonal relationship we are interested in studying (student X) were recorded on a random 17 days in one semester (comprising 60 days). We present a probabilistic analysis to estimate the expected number of peers surrounding any student who is not X. In this way, the amount of peers surrounding X to that of a generic student can be compared. Preliminary results showed that X had varying degree of interpersonal relationship in different learning environments. This method allows us to maximally explore the potential of extracting maximum information from minimum data collection, and may help facilitators identify or design learning environments suitable for certain students.

Keywords

Constructionism; Learning Analytics; Interpersonal Relationship; Probabilistic Approach

Constructionist Classroom

An approach to "learning to learn" as proposed by Seymour Papert called Constructionism is practiced at Darunsikkhalai School for Innovative Learning in Bangkok, Thailand. This school has Grade 1 through Grade 12 classes. Our research was conducted in one of the Grade 2 classrooms. A facilitator incorporated constructionism into her classroom through various activities. For example, during a project class, children would work on a project of their choice, usually under a common theme. They would conduct experiments, build, and test their artefacts, and later have conversations through a project show and share session. Some other activities included learning to use tools in Fabrication lab (Fablab) to finish off their artefacts. Even in English, Math, and Thai class, learners would, first, learn through constructing *an object to think with*. For example, in Thai class, they learned about a festival called Loi Krathong (Festival of light) by first making Krathong, which is a floating decoration used as an offering to the water spirits, usually made from banana tree's trunk, its leaves, and colourful flowers. The class's facilitator integrated other skills needed for learners to make such objects or artefacts, such as figuring out where banana trees grow, and how to get trunks of banana trees, or other substitutes to make Krathong, etc. Children would then present their work, and the class would converse on facilitated topics such as, the implication of Krathong, the significance of a festival, etc. This process allows learners to go back and forth between becoming embedded in their own world (through answering others' questions or presenting their own artefacts) and emerging from the embeddedness (through listening to others and processing new information), i.e. dancing back and forth between dwelling in and stepping out—a necessary process to reach deepen understanding (Kegan, 1982; Ackermann, 1996). Children learned Thai alphabets through, first, writing their own names, and sharing it with others. Then they naturally became curious about other alphabets that were in neither their names, nor their friends'. Learning continued on outside the classroom. They learned English through singing songs, playing games or having conversations with native English teachers.

When one student screamed and yelled at the others in class, the facilitator turned "a problem to be solved" into "a problem to be learned". Using facilitated dialogue, she created an atmosphere and space where learners learned to connect with others, have empathy, and coexist with others who are different. In this way, even a problem became a platform for collective learning, as Papert called "playing with problems," rather than "solving problems" (Papert, 1996).

This facilitator observed that children were better at constructing knowledge when they had a certain degree of awareness of self and environment such as knowing when to talk or listen, and how to respect others. When learned of Waldorf education and its benefits to help children develop connections with environments, she decided to integrate Waldorf activities into her class, which she thought would improve their learning. In Waldorf-inspired activities, children are invited to join in group activities such as circle time, story time, or creative play, engaging both bodies and artistic sensibilities to develop their bodily-will intelligence (Schmitt-Stegmann, 1997). Waldorf education emphasises the appropriate awakening of each of the 4 bodies (physical body, etheric body, astral body—that which responds to feelings or sensation, and the "I"—that of the human Ego) at appropriate ages. From age 6/7 to 14 (children in this classroom), connection with others, rhythm, repetitions and imaging are essential in developing what Rudolf Steiner termed "etheric body," when children have information imprinted in their minds as pictures and imagination (Steiner, 1996). In activities such as creative play, or circle time, everyone will sit in a circle and participate in activities that require movements. Most activities are carried out while children are together.

Appropriate learning environments can facilitate and is necessary in the process of knowledge construction. As Papert's works imply, knowledge is formed and transformed within an environment. It is context dependent, and is constructed and reconstructed through personal experiences (Ackermann, 2001). Hence, there should be certain learning environments that can best facilitate knowledge construction. Moreover, knowledge in this sense is not transmitted from one person to the next, but is co-constructed felicitously in an environment where they share and learn from others. Others (as a part of the learning environment) are necessary in this constructionist learning process. Therefore, a suitable learning environment (including people and non-people) not only helps, but also is necessary in the process of knowledge construction. The facilitator noticed that, for the most part, one student in her class stood out as lacking interactions with others. However, she was unsure if this student's low interaction was subjected to class activities and learning environments (external), or the inherent lack of ability of this student to interact with others (internal). Because of a wide range of activities that this student went through on different days, the facilitator was unsure if she could conclude her findings about the student's peers interactions based on the infrequent observational data she collected, which was often lacking in details. Additionally, the facilitator questioned if adding Waldorf-inspired activities to constructionist learning environment allows her student to bring out such inherent ability to connect better with others than if Waldorf activities were not to be added at all.

Our work sets out to assess some of their primary aspects of the learning environment with a focus on peers interactions, and infer some potential causes of their variations across different learning environments.

Assessing Interpersonal Relationships

Research has found that interpersonal relationships that provide students with a sense of belongingness can serve as strong motivators for their interests in school (Deci, 1992).

Additionally, positive relationships with peers are associated with academic performance (Wentzel, 1998) as well as a positive perception of the self (Steinberg, 1990; Youniss & Smollar, 1987). Therefore, deriving a practical approach to quantify student's interpersonal relationship with their peers is of great importance for facilitators.

Conventional approaches to assessing learner's interpersonal relationships are based on questionnaires about student's perception of their relationship with others (Anand, 1999; Dryer & Horowitz, 1997; Locke, 2000) or open-ended interviews (Chung & Asher, 1996; Erdley & Asher, 1996). In both approaches, students should be able to accurately communicate and express their feelings regarding their relationships with others, which may or may not be suitable for kids at very young age. These approaches still rely on student's cognitive maturation and development of their response strategies. Another important approach is observing individual learner's behaviours while interacting with others. While observing learners' behaviours during activities are necessary for facilitators to understand all learners, doing so would require a full attention of facilitators during class time. This can be extremely challenging to carry out and at times is impractical in the setting of Thai classroom where a typical ratio of students to teachers is 16:1 in primary schools (World Bank, 2015).

We propose one approach to assess certain aspects of interpersonal relationships or interactions with peers during activities based on probabilistic analyses. Although this approach cannot measure *all* aspects of peers interactions, it can offer *primary* insights and be practically implemented in a dynamically complex classroom. This approach requires only a small set of behavioural data for the estimation of the expected number of peers surrounding other students in class, and the likelihood that a student of interest would be surrounded by more peers than any other student in the same class. Such assessment can allow us to infer some degree of positivity in peers relationship of one student in comparison to a generic student, making our estimation independent of demographics of students in class.

Methods

Demographics

Data was collected from a Constructionist-Waldorf classroom that contains 8 students (5 male and 3 female students) with average age of 8.4 years old and standard deviation of 0.52. After the objective and research methodology were explained to parents of all students, all of them agreed to participate and signed the consent forms.

Data Collection

A facilitator wanted to assess the degree of peers relationship of a particular student (student X) who has been identified as having difficulties with his/her peers. On the days that this facilitator spent an entire day (from 9am to 4pm) with all 8 students (17 days in total), she randomly observed and wrote down the number of classmates surrounding student X. We tabulated this data together with the class schedule to obtain the frequency and time spent in each "type" of activity, which are 1) Base Activity (Story time, Project show and share, Circle time, Creative play, and Student club), 2) Main Activity (Projects, and Experiments), 3) Breaks (Lunch break, Milk break, and Snack break), 4) Core Subjects (English, Math, and Thai), 5) Curriculum Activity (Physical education, Scout), and 6) Fablab. Note that most activities in Base Activities (Story time, Circle time, and Creative play) were inspired by the Waldorf education pedagogy, whereas Main Activity, Project show and share, and Fablab were based on Constructionism.

Data Analysis

Since this is the preliminary study to assist facilitators in assessing some aspects of social engagements during class activities, we aimed at computing 1) an index of relative popularity (*IR*) of student X when compared with other students—that is the number of students surrounding student X in an activity period, and 2) period of social interaction or attention to peers (*Attn*)—that is the expected total time student X spent with his peers. Note that we assumed that *IR* and *Attn* could represent some, but not all, certain aspects of interpersonal relationships.

Index of Relative Popularity (*IR*)

Taking into account the individuality of each student, we generate all possible combinations of groups that contain a student of interest (student X in *Fig. 1*) and all possible sub-groupings that contain the rest of the students. Since our data has shown that there is no single event where we observed a group that consists of one student, we then eliminated all generated groups containing one student. We then computed the probability of occurrence for each event. This was used to compute the expected number of classmates surrounding any student who is not X in any event ($EN_{\sim X}$). Additionally, the expected number of classmates surrounding X in any event (EN_X) was also computed. The index of relative popularity of a student (*IR*) is defined as the expected number (*EN*) of peers surrounding a student in a class. We also computed Δ_{IR}, which is the relative difference between IR_X and $IR_{\sim X}$, as $(IR_X - IR_{\sim X})/IR_{\sim X}$. These parameters were calculated for all activities, which are Fablab, English, Thai, Math, main activity, base activity, gym, scout, play, snack break, lunch break, and milk break.

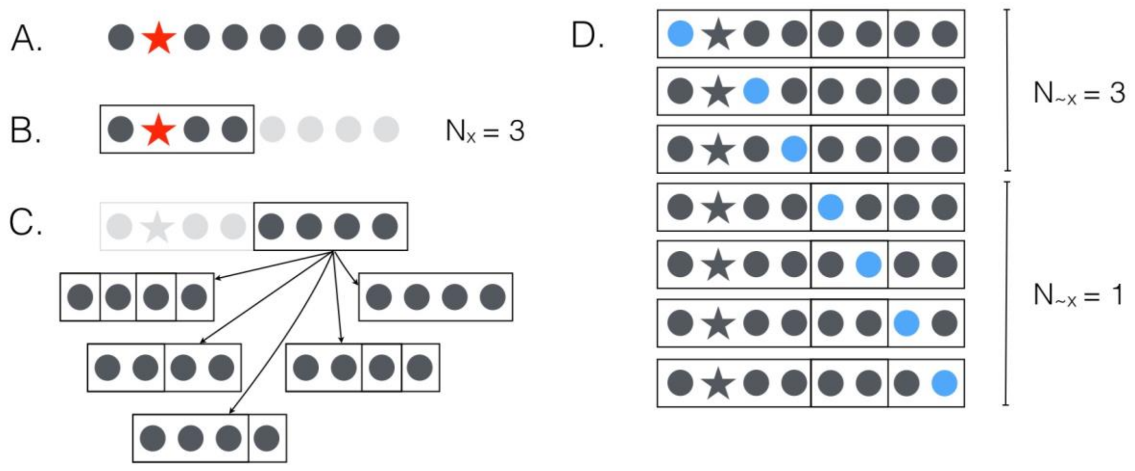


Figure 1. Images showing the steps to generate all possible combinations of groups containing a student of interest, X (a red star). A) Entire class. X is a student of interest among all 8 students. It is marked in red as a star, while other students are marked as circles in dark grey. B) A case when the facilitator observed that X was surrounded by 3 students. Hence, the number of peers surrounding X (N_X) is 3. This is one example of all possible observations. Facilitator noted the total time for each observation. C) All possible combinations of groups without X. An algorithm populates all possible grouping, and counts the number of groups, assuming that each student is uniquely different. Probability of each grouping was then computed. D) All possible grouping for a student who is not X. We computed the number of peers surrounding a student who is not X ($N_{\sim X}$) in each grouping and, hence, the probability of a student who is not X to have $N_{\sim X}$ peers surrounding him/her.

Period of Social Interaction or Attention to Peers (*Attn*)

The second aspect of peers interaction we set out to assess is the length of time a learner spends with his/her peers in any given class activity. If a learner spends more time with peers during an activity, we hypothesised that a learner pays more attention to peers than those who spend less time with peers. We assumed that such expected total time spent with peers represented X's attention on others ($Attn_X$). Similarly, we computed the expected total time any student who is not X spent with his/her peers in class was computed as ($Attn_{\sim X}$). $\Delta Attn$ was defined as the relative difference between $Attn_X$ and $Attn_{\sim X}$, as $(Attn_X - Attn_{\sim X})/Attn_{\sim X}$. These parameters were calculated for all activities, which are Fablab, English, Thai, Math, main activity, base activity, gym, scout, play, snack break, lunch break, and milk break.

Results

Using this probabilistic approach to compute the number of friends surrounding a student of interest as an index of relative popularity (*IR*), and the time this student spent with any of his peers as an indicator of attention on others (*Attn*), we found that both *IR* and *Attn* of this student is not statistically significantly different from their peers. However, in a particular learning environment such as FabLab, this student had a significantly reduced *IR* (-19.6% compared to his peers) and *Attn* (-38.3% compared to his peers).

On average, this student has slightly smaller *IR* than when compared to that of other friends in the same class by 1.6%, however, not statistically significantly different from his peers. On average, X paid slightly less attention to peers than when compared to others in the same class by 3.6%, however, not statistically significantly different from his peers.

However, *IR* of this student is significantly higher than others during milk break (11.7%) and scout class (6.4%). Note that compared to peers, *IR* of this student was markedly lower than that of others by 19.6% during FabLab, 6.9% in English, 3.6% in Thai classes. Similarly, *Attn* of this student is significantly higher than others during milk break (10.1%) and scout class (15.5%). Note that *Attn* of this student was markedly lower than that of others by 38.3% during FabLab, 13.4% in English, 7.8% in Thai classes.

IR and Attn of this student during play time, math class, gym class, lunch and snack breaks were not different from that of others (*Fig. 2A* and *Fig. 2B*). Despite having fewer peers surrounding X during scout class than milk break, X spent more time with them, making the attention paid to these peers higher than during milk break.

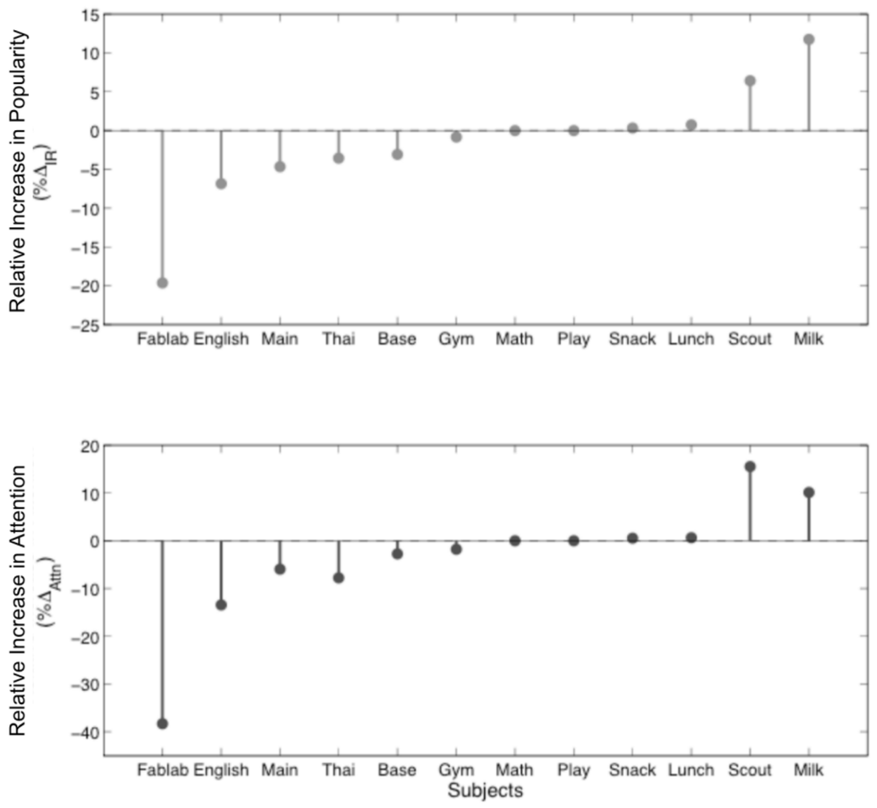


Figure 2. Plots of Index of Relative Popularity (IR) and Attention (Attn) when compared with a Student's Peers in Various Learning Environment for a Constructionist-Waldorf Class. (Top) Percent Increase in Popularity (%ΔIR) for various subjects of a student we are interested in compared to his peers. Only during FabLab, English and Thai classes, and main and base activities, this student has lower IR than his peers. (Bottom) Percent Increase in Attention (%ΔAttn) for various subjects of a student we are interested in compared to his peers. Only during FabLab, English and Thai classes, and main and base activities, this student has lower Attn than his peers. This student's Attn was higher than his peers during scout class and milk break.

Discussion

Our objective was to assess some primary aspects of learner's interactions, namely the number of peers surrounding the student of interest (*IR*), and the total time the student spent with his peers (*Attn*). Because recording the number of classmates surrounding each student in class during each class in a dynamic classroom can be an extremely overwhelming task for a facilitator to do in real time, we developed a simple approach to probabilistically estimate a number of peers surrounding any student in class from as few data points as possible. This approach requires a facilitator to only record a number of classmates surrounding a particular student of interest in each class. Using even such small set of data, we could infer certain aspects of peers interactions. Although these two measures do not represent the entire spectrum of learners' interactions with one another, they are two simple measures that are practical to collect by a facilitator while conducting a class. Moreover, they revealed some information on this student's interactions in various learning environments, which can become useful feedbacks for facilitators in designing classroom activities.

Interpretation of Results

We assessed and studied two aspects of student's peers interactions: *IR and Attn*. A positive (negative) *IR* means that X has more (less) surrounding peers than a generic student. A positive (negative) *Attn* means that X has spent more (less) time with his surrounding peers than a generic student did. We found that neither *ΔIR* nor *ΔAttn* of this student was significantly different from zero, that is, the interpersonal relationship or peers interaction of this student was not statistically significantly lower or higher when compared to that of other students. This could mean that X's peers interaction was comparable to an average generic student. Despite the lack of any significant difference among students, we found significant variations in X's peers interactions in certain learning environments. Our data shows that in subjects other than FabLab, Thai and English, this student's interpersonal relationship is on par with others.

Nevertheless, in FabLab (representing a Constructionist learning environment), this student was with fewer people when compared to his peers. However, in main and base activities, X were surrounded by approximately the same number of peers, but slightly less peers for main activity (representative of a Constructionist learning environment). During Fablab, X spent an average time of 136 out of 360 mins with only one classmate. This could have resulted from the nature of activity that may promote individual work rather than a collaborative work such as creative play or circle time in a Waldorf learning environment. This implies that in this particular Constructionist-Waldorf classroom, activities in Fablab may need to be adjusted to promote more interactions among students if a facilitator chooses to harness more collective intellectual development. For example, activities that encourage teamwork may need to be added to the existing Fablab class. Certain practice such as reciprocal teaching as proposed by Palinscar and Brown could also be applied in this classroom. In other words, discussion structured with four strategies: predicting, questioning, summarising and clarifying could be used to assist in constructing reasoning and understanding of certain concepts (Palinscar & Brown, 1984). Such discussions may prove to be a successful means to improve peers engagement of student X and, hence, further enhance his/her learning as Palinscar and Brown's work suggests.

Factors Influencing Learner's Peers Interaction

There are many factors that could influence peers interaction, such as learning environments, the nature of activities, inhomogeneity in peers relationship or personal preferences, etc. Even though our results showed that *IR and Attn* varied across activities, it did not suggest the cause of such observed variation.

One potential cause that could influence X's behaviours is the inhomogeneity in relationship with different peers. For example, X is unlikely to join the group that contains a classmate he/she does not get along with. And if that classmate were popular, X would have likely not joined others. Since we assumed that X's preference did *not* change across different activities, it is unlikely the cause of the observed variation in *IR and Attn*. Another potential cause of the observed variation in *IR* *and Attn* could be the difference in learning environments. If learning environments in Thai, English class, or Fablab were modified to be similar to that during breaks, or scouts, we could expect X's behaviours to be improved. Reasons for this could be that the learning environment in language (Thai and English) classes or FabLab did not, for this student, foster the desire to interact with others. For example, if the classroom is too restrictive or clustered, compared to an open space, learners may find it difficult to gather in groups. When there is not enough space for multiple students to comfortably gather, and work together, the social interaction can be reduced. In addition to the limitations in the physical space, activities organised during classes could also have a significant impact on encouraging or discouraging interactions. For example, if activities during language classes or Fablab are individual-based, it is unlikely that learners will gather in groups.

Knowing the detailed nature of activities in each environment may help us identify factors that influence learner's peers interaction. During Fablab session, out of 360 mins, X spent on average time of 136 mins with only one classmate. It could be that X was deeply engaged in making his artefact. In this case, we would then conclude that the reduced *IR and Attn* were not due to his lack of ability to interact with others. However, it could be that X was disengaged from learning, and not paying attention to group activities in Fablab. In this case, we would have concluded that the reduced *IR and Attn* could have been due to his low peers interaction. This is when detailed observations of X's interactions become very useful and could help in identifying the true cause of low *IR and Attn* in such learning environment. In a Waldorf-inspired activity, one would expect the entire class to stay together. Therefore, when *IR and Attn* of X was lower than that of an average student, we can expect a behaviour such as seclusion during activities. This could be the reason why the facilitator of this classroom "felt" that X did not interact much with his peers, even though our results showed that *IR and Attn* of X were not statistically significantly different from others. Our future work would include collecting detailed observational data and video recordings to help normalise for the effect of the nature of activity on peers interactions.

Limitations of our Approach and Future Work

Some limitations of our probabilistic approach, their potential effects on the interpretations of results, and future work are discussed here.

Identifying causes of different peers interactions. Although we were able to estimate variations in certain aspects of peers interaction, we could not identify the cause of the reduced interaction in certain class activities, whether the cause of low peer interaction was internal or external. However, details of the physical environment, the nature of activities, and quality of the interactions among students would greatly help deepen our understanding of this classroom. Moreover, it may help identify the cause of reduced peers interactions.

Qualitative description of peers interaction. Qualities of such interaction will allow us to understand detailed dynamics among children, which would be important when we want to further understand mechanisms that enable some learning environments to improve peers interactions, and some to discourage peers interactions. This could be done through analysing interviews, video recordings, or conversation that happened during the interaction of interest. Such information can be used to further identify and understand the specific mechanism that encourages or discourages peers interactions.

Impact of inhomogeneous peers interaction or personal preferences. In our analysis, we assumed that all groupings had equal probability. However, there could be an underlying negative relationship among peers that could impact potential grouping of learners. For example, given that X does not interact well with Y, when Y is interacting with others peers, it is unlikely that X will join in. Since the inhomogeneity in X's relationship with others is not likely to change in different activities, we expected that the observed variations in *IR* *and Attn* would remain unaltered. However, detailed observations and qualitative assessment of peers interactions would add interesting dimensions to our future analysis.

Nevertheless, our probabilistic approach offers a practical means to assess certain aspects of interpersonal relationship among peers via two measures: 1) the surface popularity index and, 2) attention measure provide advantages. Due to the limited data collected real time by class's facilitator, our approach would make it easy for facilitators to collect research data based on an observation of one student as compared to what could have happened in a general case. Furthermore, our analysis provides an alternative to quantitative assessment of primary aspects of peers interactions.

Conclusions

This paper's main contribution is presenting a probabilistic approach to assess primary aspects of interpersonal relationship among peers, such as popularity of a student, and the length of time the student spent with peers during class activities. Analyses of these primary aspects yielded interesting results. Peers interactions of a student whom a facilitator *felt* was having too little peers interaction was not statistically significantly different from others. However, a variation of peers interaction across class activities was observed. In Thai, English, and Fablab, these primary aspects of this student's peers interaction were significantly lower than others. Nevertheless, during scout and milk break, these primary aspects of this student's peers interaction were significantly higher than others. We speculated that causes of such variations in peers interaction could be due to differences in the nature of activities, inhomogeneity of peers interactions, or learning environments, etc. In the future, this analysis can and may be useful in identifying learning environments that enhance interpersonal relationships for various students, if systematically tested. Moreover, this approach offers a tool to gather primary data, which if combined with more qualitative data collection, could help deepen our understanding in this endeavour.

References

Ackermann, E. (1996). Perspective-taking and object construction: two keys to learning. Constructionism in practice: designing, thinking, and learning in a digital world, Lawrence Erlbaum, Mahwah, NJ, 25-35.

Ackermann, E. (2001). Piaget's constructivism, Papert's constructionism: What's the difference. Future of learning group publication, 5(3), 438.

Anand, S. P. (1999). A study of interpersonal interactions in schools. Social Science International.

Chung, T. Y., & Asher, S. R. (1996). Children's goals and strategies in peer conflict situations. Merrill-Palmer Quarterly (1982-), 125-147.

Deci, E. L. (1992). The relation of interest to the motivation of behavior: A self-determination theory perspective. In Renninger, A., Hidi, S., & Krapp, A. (Eds). The role of interest in learning and development (pp. 43-70). Psychology Press. Hillsdale, NJ: Erlbaum.

Dryer, D. C., & Horowitz, L. M. (1997). When do opposites attract? Interpersonal complementarity versus similarity. Journal of personality and social psychology, 72(3), 592.

Erdley, C. A., & Asher, S. R. (1996). Children's social goals and self-efficacy perceptions as influences on their responses to ambiguous provocation. Child development, 67(4), 1329-1344.

Kegan, R. (1982). The Evolving Self. Cambridge: Harvard University Press.

Locke, K. D. (2000). Circumplex scales of interpersonal values: Reliability, validity, and applicability to interpersonal problems and personality disorders. Journal of personality assessment, 75(2), 249-267.

Palinscar, A. S., & Brown, A. L. (1984). Reciprocal teaching of comprehension-fostering and comprehension-monitoring activities. Cognition and instruction, 1(2), 117-175.

Papert, S., & Harel, I. (1991). Situating constructionism. Constructionism, 36, 1-11.

Papert, S. (1996). A word for learning. Constructionism in practice: Designing, thinking and learning in a digital world, 2-24.

Schmitt-Stegmann, A. (1997). Child Development and Curriculum in Waldorf Education.

Steinberg, L. (1990). Autonomy, conflict, and harmony in the family relationship.

Steiner, R. (1996). The education of the child. SteinerBooks.

Wentzel, K. R. (1998). Social relationships and motivation in middle school: The role of parents, facilitators, and peers. Journal of educational psychology, 90(2), 202.

World Bank (2015). World Bank Data Catalog. Pupil-Facilitator Ratio, Primary. http://data.worldbank.org/indicator/SE.PRM.ENRL .TC.ZS

Youniss, J., & Smollar, J. (1987). Adolescent relations with mothers, fathers and friends. University of Chicago Press.

Beyond lesson recipes: first steps towards a repertoire for teaching primary computing

Chris Shelton, *c.shelton@chi.ac.uk*
Institute of Education, University of Chichester

Abstract

In 2014, the UK government introduced a new National Curriculum for state schools in England with a greater emphasis on computer science and computational thinking. Teaching this new curriculum presented challenges to many primary school teachers and led to a demand for professional development and exemplar teaching resources. This paper argues that many of the resources created in response to the revised curriculum are 'recipes' for lessons that fail to prepare teachers to teach challenging and purposeful computing lessons. It argues that, instead of providing recipes, we need to develop teachers' 'repertoire' of strategies for teaching computing and that our approach to doing this should take account of the context in which primary teachers now work.

The paper describes professional development practices designed to help less confident teachers take their first steps away from model lessons and towards computing projects that reflect the needs and interests of the pupils they teach.

In particular, this paper will focus on two aspects of these practices: a teaching sequence intended to scaffold teachers in planning and teaching computing, and an approach to meeting the needs of the range of learners in a primary classroom through self-directed challenges. These were intended to support primary school teachers in improving their confidence and capability to plan and teach computer programming.

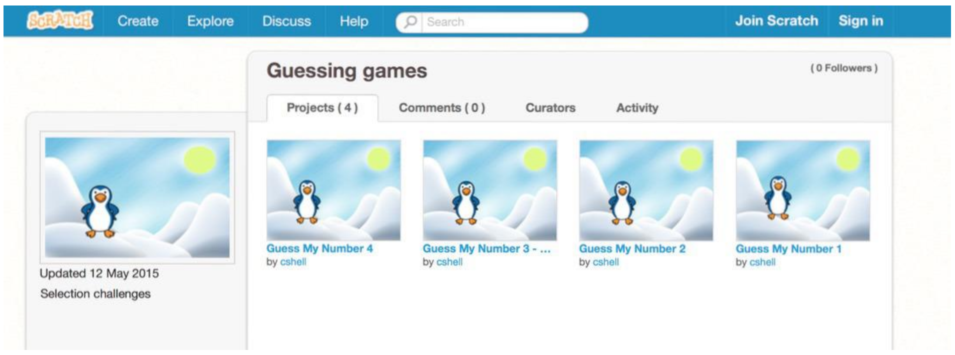


Figure 1. Scratch studio of learning challenges

Keywords:

professional development, computing, computational thinking, primary schools, elementary schools, pedagogy

Introduction

This paper describes professional development practices designed to develop primary school teachers' confidence and capability to plan and teach computer programming. Within the constructionist community, there is a long tradition of very successful approaches to professional development for teachers. In the UK, however, recent changes to the National Curriculum for England and the wide availability of new example teaching materials have changed the landscape for primary computing and led to new demands for teacher professional development. This paper discusses an approach to support teachers to plan purposeful computing activities, including those teachers who are risk-averse or lack the confidence to step outside familiar and 'safe' teaching approaches.

As this paper claims that the context in which teachers are situated should shape our approach to professional development, it will first set out the context for primary computing in England. It will then discuss two aspects of professional development: a teaching sequence intended to scaffold teachers in planning and teaching computing, and an approach to meeting the needs of the range of learners in a primary classroom through self-directed challenges.

Context and background

In 2014, the UK government introduced a new National Curriculum for state schools in England (DfE 2013). One of the biggest departures from the previous curriculum (that had been in place since 2000) was that the subject of Information and Communication Technology (ICT) was renamed 'Computing' and the content of this subject was revised with a much greater emphasis on computer science and computational thinking. These changes presented challenges for teachers of primary pupils (ages 5-11), the majority of whom are non-specialists with limited or no experience of computer science.

Unsurprisingly, teachers' lack of subject knowledge alongside a curriculum that introduced complicated and unfamiliar vocabulary posed a challenge to the introduction of the new curriculum (though arguably, this vocabulary could be seen as fairly uncomplicated concepts disguised in technical language). While there is a strong tradition of high quality computing in the UK, most notably, the teaching of LOGO enthusiasts and a small core of schools where children achieved much more than the requirements of the National Curriculum, achievement in primary schools prior to 2014 varied widely. According to Ofsted (2011), in nearly two thirds of primary schools the teaching of ICT was good or outstanding but there were particular weaknesses in the teaching of programming or 'control' technology.

Solutions to this challenge have been numerous. The UK Department for Education provided some funding for professional development for teachers notably through a website: 'Barefoot' computing (http://barefootcas.org.uk) and the Computing At School (CAS) subject association. In parallel to this, initiatives such as the 'Hour of Code', code clubs, and commercial publishers of teaching materials offered an increasingly wide choice of resources to teachers. This has included online 'coding' puzzles, lesson plans for lessons using computers and 'unplugged' activities that teach principles of computer science away from the computer itself. Such resources offer teachers 'easy wins' in their high pressured and busy schedules. At a local and regional level, there has been a rise in demand for continued professional development as schools search for teachers with expertise to offer training to their schools. Many of these professional development events have needed to concentrate on building teachers' confidence and providing reassurance. As teachers have developed their subject knowledge and been able to relate the vocabulary of the curriculum (computational thinking, algorithm, selection, etc.) to their practice they have grown in confidence and their initial fears have been quelled. As a result, they have adopted some of the example lesson ideas they have been given and started to teach computing to their pupils.

However, although these resources and basic computing skills training has provided an initial solution to teachers' anxieties about what to do in their new 'computing' lessons, they fail to equip teachers to be able to design the rich, open-ended learning opportunities that will most benefit their pupils and that are a feature of the practice of constructionist teachers.

In fact, many of the teaching resources provided in response to the demands of the 2014 curriculum are best understood as lesson 'recipes'. Alexander (2010) draws a distinction between teaching 'recipes' and teaching 'repertoire'. While lesson ideas and exemplar plans can provide a lesson recipe for teachers to follow, and this can be exactly what an anxious and inexperienced teacher desires, for a teacher to be able to design activities that meet the needs of the pupils they teach, they need to have a repertoire of teaching strategies that they can use and adapt based on their professional judgement. However, it can be very challenging for teachers to have the confidence to step away from model lessons designed by 'experts' and to take full control of their lessons even though classroom teachers are best placed to know the individuals they teach and to make connections between the required curriculum and the interests and aptitudes of their pupils. Alexander suggests that in primary education, educators need to "work towards a pedagogy of repertoire rather than recipe, and of principle rather than prescription" (2010, p511). This is particularly the case for computing in England at this time. As the initial shock of a new curriculum and the new language of computer science begin to fade, English primary teachers need support to move beyond the recipes they have been using in the first year of the computing curriculum and to develop a teaching repertoire that will support them and their pupils to achieve much more than the National Curriculum requires. In addition, it is a grounding in principles of good practice (and, in particular, constructionist principles) that can help to provide teachers with the confidence to do this.

This paper discusses some of the practices that have developed over the last year of attempting to develop primary teachers' computing repertoires in the new context they find themselves. The principal aspect of the approach described here has been to present a particular teaching sequence that can be seen as a first strategy to adopt. It is envisaged that this sequence, far from being the only way to teach computing, is a first step towards building a teachers' repertoire: a single strategy but one that can be applied in many lessons and which lends itself to being adapted for different topics. At its heart is the assumption that children will learn computing best through the creation of meaningful projects rather than limited puzzles (Resnick, 2014). The second aspect of this approach is to help teachers to learn to meet the needs of the learners in their class through self-directed challenges.

In addition, this paper will make explicit the connections between these teaching approaches and two other concerns of primary teachers: dialogue and 'mindset'. This serves to help teachers to connect what they need to do when teaching computing to their understanding of good practice in the rest of their teaching, specifically, the importance of dialogic teaching (Alexander, 2006) and the benefits of encouraging a 'growth mindset' (Dweck, 2006). These are increasingly becoming accepted ideas amongst UK educators and as such, provide a familiar anchor from which to start improving computing teaching.

A teaching sequence for primary computing: "UpTIME"

Soon after the launch of 2014 National Curriculum, it became clear that the initial professional development focus on teachers' programming or 'coding' skills and knowledge of computer science concepts and vocabulary, whilst important, was not going to give teachers the pedagogic skills necessary to move beyond the lesson 'recipes' that had suddenly become popular in primary schools. Therefore, the challenge for teacher professional development was to encourage and support teachers to take greater control of their planning and teaching for computing. While some teachers were quick to appreciate that the language of the National Curriculum can be interpreted very broadly and that it provides scope for varied and exciting lessons, others needed much more support and reassurance.

During 2014-15, staff at the University of Chichester were asked to run a sequence of computing workshops for pupils from local primary schools. These focused on using Scratch to meet and exceed the requirements of the National Curriculum (and also on empowering pupils to return to their schools and support others who were learning computing). In discussing with teachers how these sessions were designed and the principles underpinning them, it was clear that an analysis of these workshops could provide a template for future planning. This template became the teaching sequence described below. The intention of this work was not to provide another lesson 'recipe' but rather to provide a scaffold – a structure and strategy for planning that could be applied across many aspects of the computing curriculum. Secondly, the sequence was designed in such a way that by introducing the sequence, both new and experienced teachers would be made aware of some of the principles that underpin computing teaching.

A single page summary of the teaching sequence written for teachers and intended to be used in professional development sessions can be found online at:

https://challengingcomputing.wordpress.com/uptime/

The teaching sequence was given the acronym "UpTIME" which stands for:

- Use/ play
- Tinker
- Improve
- Make
- Evaluate

It is designed as a simple way to plan effective sequences of activities to teach computing that teachers can use or refer to when planning. There is intentionally no indication of how long teachers should spend teaching the sequence or on any individual part of the sequence and, in fact, teachers are encouraged to consider how it might be used within a single lesson or over a much longer series of lessons.

For experienced teachers, one of the most exciting things about project-based learning in computing is that children will surprise us by developing ideas and products that we could not have imagined at the start of a project and that 'teachable moments' present themselves when least expected. However, for the teachers for whom this teaching sequence is intended, this can be a daunting prospect. Therefore, the sequence encourages teachers to identify a specific aspect of computing that they wish their pupils to learn based on their prior assessment of pupils' learning. It is intended as a 'learning driven' rather than 'activity driven' approach which complements the pedagogic practices that they are familiar with from English primary teaching.

U/p – USE / play

The first stage of the sequence is intended to allow teachers to provide an authentic and purposeful context for the computing concepts and skills that they want their pupils to learn. It encourages teachers to allow their pupils to use programs that make use of the specific elements of computing being taught and, if this program is in the form of a game, then to play the game. For example, if the teacher has identified that their pupils need to develop their understanding of variables then they might play a game that makes use of variables in a number of different ways, e.g. for scores, lives or levels.

Some of the lesson 'recipes' used in UK schools begin by introducing abstract concepts or vocabulary (often away from the computer). While there can be a place for this, teachers need to motivate and enthuse their class and help them to see the relevance of what they are being taught. Playing games that use the concepts that will be introduced later is one way of providing a context for later work. Alternatively, sometimes the initial program that children use or play may be incomplete or 'broken' to draw their attention to the part of the program that has gone wrong.

One aim of the sequence is to maximise opportunities for children to learn through productive talk with peers and adults so at this stage teachers can be encouraged to ensure that pupils try to explain to a partner or the class how they think the game works.

T – TINKER

The second stage in the sequence asks teachers to let pupils investigate the particular learning focus by tinkering with the program or game that they have already used. This involves pupils looking closely at the program, discussing with their partners how it works, changing the program and seeing what happens. Papert (1993) described learning itself in terms of tinkering or ‘bricolage’: “building up a set of materials and tools that one can handle and manipulate” (p173) and here the pupils are learning through their manipulation of the program.

This is also an opportunity for the teacher to encourage ‘dialogic’ talk that is cumulative, reciprocal, supportive and purposeful (Alexander, 2006). They can share their pupils’ ‘tinkerings’ with the rest of the class and let others build on these ideas. They can also elicit key concepts and model the vocabulary they want their pupils to use.

I – IMPROVE

As soon as pupils start tinkering, they discover ways to ‘break’ programs and ways to improve them. At this stage, teachers are encouraged to allow the children to make small purposeful changes to programs. Some of the ideas for improvement will come from the children while others may be designed by the teacher. It is at this stage that teachers are encouraged to cater for the range of abilities that can be found in their class through setting a variety of different learning challenges (see below).

M – MAKE

This stage is the most crucial and the teaching sequence is designed to scaffold teachers towards this. By giving children the opportunity to design and make their own projects, they develop independence, are motivated to learn new concepts to achieve specific outcomes, and enabled to make connections between abstract concepts and their application. Here, teachers can support pupils to design programs, decompose them into manageable tasks, create, debug and share their work.

However, by placing these open-ended (and potentially daunting) projects within a focussed series of activities, the sequence supports teachers in two ways. Firstly, by helping them to connect such projects to the specific demands of the National Curriculum and, secondly, by allowing them to gain confidence through fairly tightly defined activities before ‘letting go’ and enabling their pupils to explore more widely.

E – EVALUATE

The final part of the teaching sequence is ‘evaluate’ but this is not meant to imply that pupils only evaluate their work (both their learning and their programs) at the end of the sequence. Children will evaluate their work at every stage of this teaching sequence: they will evaluate other peoples’ programs at the start of the sequence and they will improve and debug their own programs continually when programming. But, in addition to this, teachers need to create space in their timetable for individuals and groups of children to reflect upon and evaluate the technologies they have used, their own creations, their skills and their learning.

Setting suitable learning challenges

If there is one event that most clearly demonstrates the limitations of model lessons and teaching ‘recipes’, then it is when these lessons fail to meet the needs of many of the children in a group. In any primary classroom, the pupils will have a wide range of attainment in computing. While some children may have had limited access to technology outside of school, others may use technology frequently at home and enjoy to explore and creative with digital tools. As a result, lesson ‘recipes’ designed for an imaginary class of children can frequently fail to meet the needs of the most or least confident pupils.

In the UK, primary classes are rarely set or streamed and teachers are experienced in catering for a wide range of attainment in one class. However, media images of children as expert computer users (cf. Selwyn 2003) can lead teachers to view their pupils as very likely to be much more knowledgeable about computing than they are and this can lead to anxiety about teaching computing.

As a result of this concern, a second aspect of professional development for primary teachers has been to demonstrate and promote an approach to meeting the needs of the range of learners in a primary classroom through self-directed challenges. This is intended to help teachers ensure that all their pupils are suitably challenged, in particular that higher attaining pupils are extended but that those who might struggle with computer programming do not become frustrated and disengaged.

As mentioned above, one of the aims of this approach is to make connections between the teaching of computing and primary teachers' understanding of pedagogy in other subjects that they may be more confident teaching. In this particular case, it is useful to introduce the approach with reference to Dweck's (2006) work on 'growth mindsets' and the application of this to the primary classroom through the 'Learning without Limits' project (Hart *et al*., 2004).

Carol Dweck's work on 'growth mindsets' has grown in popularity in the UK and increasingly teachers are becoming aware of the difficulties caused when teachers view pupils as being of fixed ability and when pupils view themselves this way. In addition, Hart *et al.* (2004) have demonstrated how the practice, commonly found in England, of classifying children as 'high ability' or 'low ability' learners can lead to unhelpful fixed mindsets.

If we apply these ideas to teaching computing, then those pupils labelled as 'low ability' may believe that they will never be able to master computing while those labelled as 'high ability' may become reluctant to take risks for fear of making mistakes – a particular problem for a subject like computing that has 'tinkering' and learning through error at its heart.

One of the strategies used by schools in the 'Learning without Limits' project to encourage children to fully participate and take responsibility for their learning was the use of learning challenges. In applying this to the teaching of computing and the demands of the National Curriculum, teachers are encouraged to set three or four challenges for their pupils and to allow the children to select the most appropriate challenge for themselves. Pupils are not required to work through all of the challenges in order but are encouraged to take responsibility for their learning.

For example, at Key Stage Two of the National Curriculum for England (Key Stage Two refers to pupils aged 7-11), pupils are required to "work with variables". If a teacher is using the 'UpTIME' sequence, then the children might begin by playing games that include variables for scores, levels, etc. Then they will tinker to discover how these variables work and the teacher will lead the class in exploring and discussing the concept of variables and how (and why) they are used. The tinkering stage should be quite open-ended so that pupils have some freedom to explore for themselves but this should not overwhelm anyone so the teacher might set some 'tinkering challenges'. Then he/she might ask the children to improve the games and could set simple challenges (e.g. change how many points are scored each time) or more complex ones (e.g. add a new variable to represent how many 'lives' the player has left and make the game finish if they lose all their lives).

If the teachers are using Scratch then studios can be very effective ways of sharing the set of challenges. Below is another example – a Scratch Studio of challenges called 'Guessing Games' (see Figure 1).

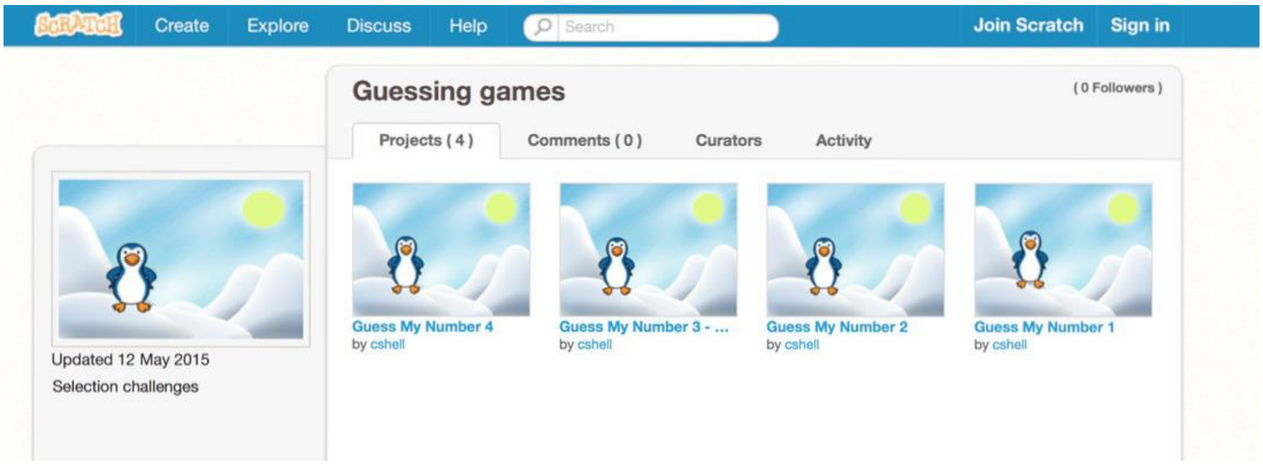


Figure 1: Guessing Games Studio

This studio was designed to help teach selection using ‘if’ statements. The four different Scratch projects all use the ‘if’ or ‘if…else’ block but get progressively more complex. Guess My Number 1 is the simplest program and allows pupils to focus on how the ‘if’ condition works. The second version improves the guessing game but is more complicated (for example, the ‘if’ block is nested within another block). The third game shows how the game can work with words as well as numbers while the fourth introduces ‘more than’ and ‘less than’ operators. The four games can be used to challenge pupils in several different ways. For example, a teacher might ask his/her pupils to choose one of the games and explain how it works to a partner, or they might ask pupils to improve one of the games and let them pick one that they feel they understand as their starting point. For a different class, a teacher might need to give the children more challenging opportunities, for example, a challenge to create a game that uses random numbers or that keeps a count of how many attempts it takes a user to guess the correct answer.

In summary, encouraging teachers to set a range of learning challenges in computing gives them a strategy to meet the needs of the range of learners in their class while also helping to make connections between this ‘new’ subject of computing and their existing understandings of effective primary pedagogy.

Conclusion

Faced with the multiple challenges of life as a teacher in the 21st century, it is no surprise that educators rush to collect teaching ‘recipes’ that offer easy solutions to planning computing. But while the limitations of such recipes are well known, finding accessible ways of moving from recipes to a varied teaching repertoire is challenging. While we know that enthusiastic teachers will find ways of navigating through curriculum documents and teach in innovative ways, more support is needed to help less confident teachers.

The teaching sequence offered here is not a recipe to teach a single lesson but rather a tool for teachers to use and adapt to enable them to meet the needs of their pupils. The teaching sequence described in this paper might be thought of as a scaffold to support teachers as they move from following recipes to designing purposeful and authentic teaching and learning experiences. It is not intended as an end in itself but rather as a first step in the direction of a wide constructionist teaching repertoire.

A number of questions remain about how teachers develop a full and varied pedagogic repertoire for computing that require further empirical investigation. We are beginning to explore how the teaching sequence described here is used by teachers and what other approaches might function as first steps to encouraging teachers who are confident enough to abandon their recipe books and design their own teaching and learning opportunities. In addition, while there is excellent evidence for the use of dialogic teaching that uses digital technology to enhance learning across the curriculum (e.g. Wegerif and Dawes, 2004), more research is needed to investigate how teachers can be supported to model and promote exploratory talk when teaching computing. Similarly, while there is excellent work on growth mindsets in certain other subjects, there are opportunities to do more to investigate how these concepts relate to the teaching of computing.

References

Alexander, R. (2006). *Towards dialogic teaching: Rethinking classroom talk.* Dialogos, York.

Ed. Alexander, R. (2010). *Children, their World, their Education: final report and recommendations of the Cambridge Primary Review.* Routledge, Abingdon.

Department for Education (2013) *The national curriculum in England: Key stages 1 and 2 framework document*. Available at: https://www.gov.uk/government/publications/national-curriculum-in-england-primary-curriculum [Accessed 14 October 2015]

Dweck, C. (2006). *Mindset: The new psychology of success.* Random House, New York.

Hart, S., Dixon, A., Drummond, M.J. and McIntyre, D. (2004) *Learning without Limits.* Maidenhead: Open University Press

Ofsted (2011) *ICT in schools 2008–11: An evaluation of information and communication technology education in schools in England 2008–11.* Available at: https://www.gov.uk/government/publications/ict-in-schools-2008-to-2011 [Accessed 14 October 2015]

Papert, S. (1993). *Mindstorms: Children, Computers, And Powerful Ideas (Second Edition).* Basic Books, New York.

Resnick, M (2014) *Give P's A Chance: Projects, Peers, Passion, Play.* Paper presented at Constructionism 14 Conference, August 19th - 23rd 2014, Vienna, Austria. Available at: http://constructionism2014.ifs.tuwien.ac.at/papers/1.2_1-8527.pdf [Accessed 14 October 2015]

Selwyn, N. (2003). *Doing IT for the Kids': Re-examining Children, Computers and the Information Society'*. Media, Culture & Society, 25(3), 351-378.

Wegerif, R., Dawes, L. (2004) *Thinking and Learning with ICT: Raising Achievement in Primary Classrooms*. Routledge, London.

Building mathematical knowledge with programming: insights from the ScratchMaths project

Laura Benton[*], l.benton@ucl.ac.uk; **Celia Hoyles**[*], *c.hoyles@ioe.ac.uk;* **Ivan Kalas**[*^], *i.kalas@ioe.ac.uk; kalas@fmph.uniba.sk;* **Richard Noss**[*], *r.noss@ioe.ac.uk*
[*] London Knowledge Lab, UCL Institute of Education, 23-29 Emerald Street, London, UK
[^] Dept. of Informatics Education, Comenius University, Bratislava, Slovakia

Abstract

The ScratchMaths (SM) project sets out to exploit the recent commitment to programming in schools in England for the benefit of mathematics learning and reasoning. This design research project aims to introduce students (age 9-11 years) to computational thinking as a medium for exploring mathematics following a constructionist approach. This paper outlines the project and then focuses on two tensions related to (i) the tool and learning, and (ii) direction and discovery, which can arise within constructionist learning environments and describes how these tensions were addressed through the design of the SM curriculum.

Keywords

Programming; mathematics; Scratch; primary education; design research

Introduction

Computer programming is undergoing a renaissance in English schools. Recent policy and curriculum initiatives have resulted in ICT being replaced by computing across all ages from 6 to 16 years. These changes have been motivated by a concern about students leaving school with little understanding of computer science or the creative side of computing (Furber 2012). From September 2014, schools in England[1] have to teach the new National Computing Curriculum (DfE 2013), which requires students to learn about how computational systems work, to use technology safely and to design and build their own programs. At least at the policy level, computing is recognised as not just about programming *per se*, but programming as a modeling tool: a key component of thinking that allows ideas to be brought to life and explored in different subject areas and contexts. How far this will happen in practice is of course a complex matter shaped by schools, teachers and available resources (material and people) to support this work.

Much of the research in the field of programming within schools was conducted in the latter part of 20th century before the advent of the many new blocks-based programming environments developed specifically for young users (Weintrop & Wilensky 2015). One is Scratch, used by a huge number of young children in and out of school (with over 2 million registered users aged under 12 years). The popularity of this style of programming for use with novice programmers is in part due to its ease of readability, composition and browsability alongside its interactivity, and visual and dynamic outcomes (*ibid*).

In this paper, we introduce the ScratchMaths (SM) project, which aims to build mathematical knowledge through programming in Scratch during a 2-year intervention for students aged 9-11 years. We set out to exploit programming to support mathematical reasoning in pre-specified mathematical content areas and thus will explore among other areas, Papert's (1972) claim that learning to program in carefully designed ways should "make it easy to learn algebra and

[1] Within England a growing group of schools known as academies do not have to adhere to the national curriculum so can opt out of computing – an interesting dilemma

geometry" (p.4). We describe the design process leading to a constructionist curriculum and professional development program where students are able to exploit the powerful ideas of computational thinking and programming tools to engage in mathematical thinking. We borrowed from Brennan (2015) the identification of a number of tensions in supporting constructionist approaches[2], in this paper we focus on two: the tensions between (i) tool and learning, and (ii) direction and discovery. We describe how these tensions were addressed in the design and implementation of the SM curriculum.

Background

Tension between tool and learning

In the 1970s and 80s the earliest research in schools took place that explored the potential of learning mathematics through programming languages, (such as BASIC and Logo), (Hoyles & Noss 1992). As Logo became more integrated into schools, a perception grew that programming was too difficult to have any widespread impact on mathematics learning. An often-cited reason for this was the perception of programming as an *overhead* – something to be squeezed into an already-overcrowded curriculum. As Resnick et al. (2009) point out, the difficulty of mastering programming syntax, and the lack of specific skills/knowledge that was required by teachers to effectively guide or challenge students in capitalising on these early programming tools, posed problems in exploiting this potential. There are now growing concerns that this perception may come full circle with the 'floor' being lowered so far that novice programmers are discouraged – or at least, not encouraged – from engaging with the underlying concepts, which may in part be due to the implicit ways compilation errors are handled in tools such as Scratch. Thus, they can achieve a visually pleasing outcome on the screen almost 'by accident' with no desire to ask *why* it happened.

The accessibility of the programming *tool,* and the process of *learning* through programming using the tool, is identified as the first of Brennan's tensions. Brennan (2015) describes this as the necessity of achieving a balance "between knowledge about the tool and understanding of how to engage in creative design activities, using the computer for personal expression and problem solving" (a similar analogy in mathematics education is the focus on what is termed instrumentation) . Furthermore, there remains the critical challenge to exploit knowledge that has been gained within programming contexts to promote engagement with mathematical ideas and reasoning (Hoyles & Noss, 1992).

Despite recent innovations of programming tools, which aim to support a constructionist approach to learning whilst making the tool more accessible to a more diverse range of learners, the tension between content knowledge of the programming tool and pedagogical knowledge is still an important issue to address within the classroom. In their commentary on Brennan's paper Gash and McCloughlin highlight the close relationship between content and pedagogy, suggesting that teachers may find it easier to address the pedagogical issues. Furthermore through their observations and interviews with teachers (primary, secondary and university) during a series of Scratch workshops, Bustillo and Garaizar (2014) suggest that often students and teachers "have a limited, immediate, and concrete vision of using Scratch (e.g. a step-by-step guide to program a video game during a semester), instead of realizing the cross-curricular potential of computational thinking". They advocate a set of best practices, learning guides and curriculum models to help teachers and students encounter the richness of the pool of ideas embedded within Scratch. We concur but would go further: teachers need to appreciate the core goals of the programming activities, the power of computational thinking skills and the purpose of exploiting them in mathematics.

[2] The complete list of tensions include (i) tool and learning, (ii) direction and discovery, (iii) individual and group, (iv) expert and novice, and (v) actual and aspirational.

Tension between direction and discovery

Much research in this field has focused on extra-curricular activities that were either voluntary or involved specially selected students, with rather few studies in naturalistic classroom settings (Lye & Koh 2014). Israel et al. (2015) also note the lack of research examining how teachers implement school-wide computing initiatives at the elementary level and particularly highlight this as the case for diverse students (in terms of background and including those affected by poverty or disability). Bers et al. (2014) claim that one key factor in the successful implementation of a programming-based curriculum is to understand how to support individual teachers' needs, especially in terms of curriculum modifications, classroom management alternatives and forms of adult support. Furthermore one failure with early programming initiatives was a neglect to design explorations that linked to young learner's interests or experiences (Resnick et al. 2009).

Defining the role of the teacher and the details of the curriculum design also present a challenging dilemma that needs to be addressed. This was framed as the 'play paradox' by Noss and Hoyles (1996): the problem of designing a learning activity in a way that allows students to explore and construct ideas for themselves, but also ensuring that they encounter the powerful ideas embedded within the activities; thus balancing exploration and guidance. Brennan's second tension (2015) resonates with this paradox: i.e. the tension between "direction (providing resources in advance, anticipating and steering learner needs) and discovery (making resources available when they are needed, in response to learner needs)". We claim however that an overarching challenge for those whose interest lies in teaching mathematics more effectively is not only one of pedagogy, but of defining and elaborating new kinds of mathematical knowledge that can be expressed by programming. We now turn to the SM project, which has attempted to tackle this challenge.

The ScratchMaths project

Programming in schools has been shown to have the potential to develop higher levels of mathematical thinking in relation to aspects of number linked to multiplicative reasoning, mathematical abstraction including algebraic thinking as well as problem solving abilities (Clements 2000). More recently, attention has been paid to defining computational thinking (McMaster et al. 2010; Wing 2008), which is seen by Wing, for example, as part of a 'family' of different aspects of mathematics thinking (Wing 2008). This relationship helps to explain why programming and computer-based mathematical instruction have been found to have a positive effect on both student attitudes, and on attainment in mathematics (Clements 1999). But of course, such results depend fundamentally on the design of materials, support and implementation. The SM project aims to maximise the benefits of programming for students' mathematical thinking, reasoning and attainment. The overarching goal of the research is to iteratively design and evaluate, both quantitatively and qualitatively, materials for students and teachers that directly address the learning of computational thinking and its exploitation to enhance mathematical engagement and attainment.

ScratchMaths is a 3-year research project involving a 2-year intervention with students aged 9-11 years. The intervention is intended to comprise approximately 20 hours teaching time across each of these two school years, with the first year focusing on computational thinking with an implicit mathematical component, and the second year foregrounding explorations of key mathematical concepts using programing tools. The intervention has been subject to cycles of iterative design research with the final quantitative outcome measure being based on national standardised mathematics test scores, taken by all students at the end of primary school.

Methodology

Design workshops

At the start of the project seven teachers from four London primary schools were recruited to act as 'design partners' for the SM intervention. The teachers were either class teachers or had responsibility for teaching computing for this age range, and had a variety of experience with using

Scratch ranging from none to reasonably experienced. The teachers attended five 'twilight' design workshops at the university to help them to understand the main features of Scratch as a computational tool as well as to encourage them to work collaboratively with the project team to design, test and evaluate potential student activities. The teachers also shared some of their experiences of teaching computing and maths, with project team members visiting each of the schools to observe some lessons. This design phase established that the intervention needed to be appropriate for teachers with a range of experience in computing and in using Scratch as well as for students of wide attainment levels and support needs[3]. It was apparent that the materials needed to be clear as to ways the intervention could fit within an already full teaching schedule through making explicit links with the existing curriculum, as well as signpost critical teaching points and progression and suggest discussion opportunities to reinforce understanding.

The project team also conducted several intensive internal design workshops to set out a high-level overview of the entire SM curriculum guided by existing knowledge and experience from within the team's prior research and the findings from the design workshops and school visits. This overview identified the key concepts to be introduced and an overarching structure. Each year would be structured into several modules, each with multiple investigations, and each of which comprising a series of activities, mixing hands-on and unplugged. The activities for the first investigation were then planned in detail so it could be trialled in the design schools.

Design research in schools

A design research approach was followed primarily to establish the suitability of materials for use within the current primary school context and how far the teacher was able to understand and communicate the key learning objectives of each activity as well as feel comfortable with the content. The activities needed to be: sufficiently adaptable so as to be accessible for all students while offering challenges for the higher attainers, and also presenting a balance between scaffolding of concepts and space for exploration. Further the structure of the content was modified so it could be taught in lessons of varying length and frequency. The design research was undertaken in four design schools over a year with one school progressing through the entire Y5 curriculum with three Y5 classes and the remaining three design schools testing a subset of the materials with Y5 students in their schools. During the lessons in which these materials were trialled, three researchers with a range of backgrounds, which included expertise in computer science, mathematics teaching and primary school research, conducted observations. This involved the researchers writing extensive field notes during these observations and speaking with both teachers and children to gauge the appropriateness of the different materials. After each lesson observation the researchers met to discuss what worked well and where improvements could be made to the materials. Minor changes were agreed amongst this sub-team, with more significant changes discussed with the wider project team. The materials were then trialled again with a different class (or in the case of substantial modifications the same class) to check the appropriateness of these changes.

Addressing the challenges through design

One key outcome of the curriculum design process described above was a 'framework for action' (DiSessa & Cobb 2004), which we named the "5Es"[4]. This framework (consisting of five unordered constructs) was clearly framed by a host of research into good practice in teaching mathematics but also emerged from the early design workshops and was refined through the design research in schools. It has been developed to provide guidance on the pedagogical strategies teachers may

[3] The SM project is charged with "narrowing the attainment gap", which requires as many students as possible to be successfully engaged

[4] This is different from, but clearly intersects with, the BSCS 5E Instructional model, which includes five phases: Engage, Explore, Explain, Elaborate and Evaluate, and is primarily targeted at science education (Bybee et al. 2006)

adopt to successfully implement different aspects of the SM intervention. This framework is described in more detail below in relation to the two of the tensions identified by Brennan.

Explore

Papert (1980) believes that children should use computers to explore their thinking processes, suggesting with respect to Logo that the primary learning experience is about "getting to know the Turtle, exploring what a Turtle can and cannot do". Constructionist approaches value learning in this way through design activities (Brennan 2015), which provide opportunities to explore ways to deal with different constraints and ambiguity through employing skills such as iterative thinking, problem solving and creativity. Therefore this first construct suggests the importance of developing and supporting activities that allow learners to investigate ideas, try things out *for themselves* and debug conceptual and technical errors where necessary. Part of this endeavour is to shift students towards 'taking control of their own learning' and to seek out the reasons behind different outcomes.

The activities designed around this construct addressed two of the tensions that were highlighted by Brennan, tool and learning as well as direction and discovery. During the trials in design schools it was noted that the children's previous experience and knowledge of the tool had a major influence on their approach to SM activities. Students with previous experience of using Scratch would often use the blocks and strategies with which they were already familiar, whereas students with no experience of Scratch were more cautious about trying things out that they had not been explicitly told to by the teacher. Early in the SM curriculum students are introduced to tools for exploration within the Scratch environment. For example the use of what Blackwell (2002) describes as direct manipulation - a single action with a single visible effect. Within the SM curriculum this term is seen as encompassing the manipulation of all objects both physical (i.e. BeeBots) and digital, and therefore we use the term 'direct drive' to refer just to digital objects. Direct drive is used to firstly explore the blocks on their own (Fig. 1 left), by clicking them and observing the reaction, an important precursor we claim to using them to build more complex scripts (Fig. 1 right). This gradual progression from direct drive of blocks to planning and building behaviours built into scripts provides a structured approach to exploration encouraged throughout the SM curriculum.

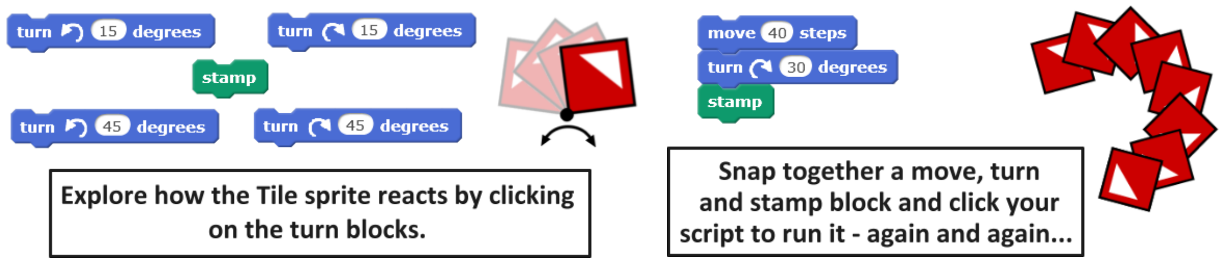

turn ↺ 15 degrees
turn ↻ 15 degrees
stamp
turn ↺ 45 degrees
turn ↻ 45 degrees
Explore how the Tile sprite reacts by clicking on the turn blocks.
move 40 steps
turn ↻ 30 degrees
stamp
Snap together a move, turn and stamp block and click your script to run it - again and again...

Figure 1. Example direct drive activity (left) progressing on to building simple scripts (right)

Explain

A crucial aspect of understanding ideas is being able to explain what has been learned and articulating the reasons behind a chosen approach (using different modes of communication). This helps clarify ideas, by expressing them explicitly as well as in answering questions from peers. Several theorists have highlighted the cognitive benefit of generating verbal explanations (Harel & Papert 1990). For example, Brown (1988) has shown that being encouraged to explain and represent knowledge in multiple ways can increase motivation and levels of understanding as well as subject mastery. In relation to mathematics, Hoyles (1985) discusses how language can facilitate reflection and internal regulation, and how part of this process is the identification of which parts of the mathematical idea are important and which are not. This reflection, or thinking about one's own thinking is a key component of the constructionist approach (Han & Bhattacharya 2001), with the programming language itself becoming the tool "to think with". This second construct suggests the importance of incorporating reflective questions and opportunities for discussion with peers as well whole-class interactions orchestrated by the teacher.

The activities designed around this construct seek to address the tension between the tool and learning. Once students are familiar with the tool, it was observed that the ease of building scripts may encourage students to create extremely long scripts which then appeared to have status as demonstrating a lot of 'work'! However, it is difficult to understand and predict what these scripts would do when clicked. Another key idea of the SM curriculum is definitions, an under-used component of Scratch but which helps to reduce complexity and aids the readability of scripts. The SM curriculum promotes the use of definitions through the process of building a script and then giving it a meaningful name (e.g. Fig. 2). This in turn supports students in explaining step by step what their script is doing and what outcome they intend to happen.

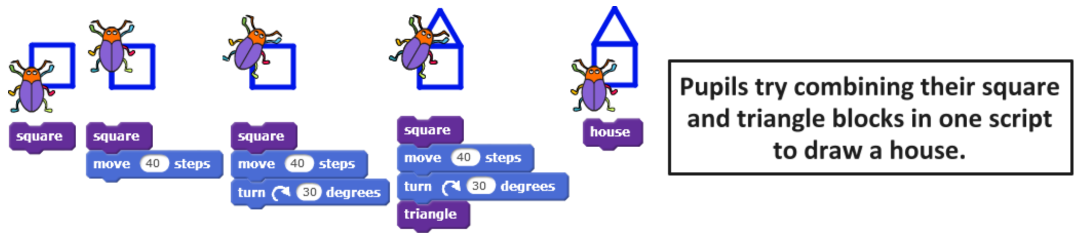


Figure 2. Example activity where students are encouraged to define blocks that then help them explain how they have drawn their houses

Envisage

It is important to have a goal in mind when building a computer program and to predict what the outcome might be *before trying it out.* Papert (1980) describes the role of building computer programs in facilitating reflection on intuitive expectations and knowledge, and highlights that the link between the idea and the child's intuitive knowledge is seen as key in understanding the power of the idea (Papert 2000). However, Rader et al. (1997) found that in their work using a children's programming environment, children can easily create programs without "much knowledge of the underlying program mechanisms" (as mentioned above). As often novice programming tools can now manage much of the syntactic error handling, students are only required to 'debug' when they have a clear goal, in other words it is quite straightforward to generate *an* outcome but not necessarily a *specific* outcome decided upon in advance. Therefore to truly understand an idea it is necessary to take time to predict the outcome *before* building the program and then compare the actual outcome with this prediction. This enables the establishment of whether the original intuition was correct or whether this knowledge needs to be remodelled (Papert 1980). This third construct suggests a need for some learning activities to be conducted prior to exploration with the programming tool, to provide learners with the opportunity to consider the program goal and to predict the potential outcomes of using different strategies. It is important this construct is balanced with explore, providing exploration opportunities that allow discoveries to be made but also occasions to envisage the outcome first.

The activities designed around this construct intend to address the tension between the tool and learning. Many students observed during the trial were very happy and excited with any outcome that resulted in an attractive pattern being stamped on the screen or a fun animation being played out, which they were able to produce without necessarily fully understanding how they created it or even having a clear goal in what they were aiming to achieve. The SM curriculum thus includes a series of unplugged activities that require students to work off the computer, encouraging them to practice prediction, reflection and debugging skills before they test things on the computer, and to reflect on how to engage with similar activities in other areas of the curriculum (see bridgE below). It also promotes body syntoncity (Watt 1998) by encouraging both teachers and students to envisage themselves as the sprite through activities which require them to *act out* the scripts or through physical objects such as paper cut-outs and toys.

Exchange

Collaborating and sharing is a powerful way to learn, with constructionist approaches advocating the development of ideas through interactions with others (Han & Bhattacharya 2001). This allows you to 'decentre', while trying to see a problem from another's perspective as well as defend your own approach and compare it with others. Furthermore Hoyles (1985) suggests that others' ideas can potentially result in modifications to an individual's thought processes, particularly helpful in clarifying predictions or explaining ideas that are not yet fully formed. Bruckman (1998) has also undertaken research demonstrating the cognitive, social and psychological benefits that undertaking constructionist activities as part of an online community can provide. However, children are still developing their collaboration skills and may need help to work together, resolve disagreements and question one another (Hoyles 1985). Therefore the fourth construct requires the inclusion of meaningful opportunities to share and build on others' ideas.

The activities designed around this construct intend to address the tension between direction and discovery. Collaborative learning offers the potential to promote less directed exploration and discovery. During the trial in schools it was observed that pair work could encourage discussions, requiring students to explain their strategies and discoveries to their partner. Some teachers arranged mixed ability pairing encouraging the more able students to support less able students by 'teaching' them what they had already discovered for themselves. Individual discoveries were also observed quickly spreading around the whole class without teacher intervention through the students monitoring what their peers were working on.

bridgE

Powerful ideas should be embedded in any well-designed constructionist activity (Bers et al. 2014), and ideas are seen as powerful partly through their connections with other disciplines, such as mathematics (Papert 2000), and partly by virtue of the language in which they are expressed. In order to develop these connections the ideas need to be re-contextualised and re-built in the language of the other discipline. Therefore the final construct requires that activities or teacher moves be suggested to make explicit links to another context (in our case school mathematics).

The activities designed around this construct look to address the tension between the tool and learning. In one of the classes during an activity, which required circular repeated patterns to be stamped, students were observed calculating the value of the repeat block by dividing 360 by any chosen value in the turn block. Sometimes this resulted in a decimal number, e.g. 5.5, which they then inputted into the repeat block. Scratch automatically treats the input to repeat by rounding it prior to running (as it is not possible to stamp .5), which in this case was done by rounding up. To ensure these important mathematics learning opportunities are not overlooked due to the behaviour of the tool, explicit links with the mathematics curriculum are made and suggested teacher discussion starter questions are provided within the materials. The unplugged activities are also intended to make further consolidate these links away from the computer.

Conclusion

The ScratchMaths project is still in progress and here we primarily focus on describing the design process adopted to develop the curriculum materials. We have also provided glimpses of the myriad of challenges in implementation. In our presentation we will provide some interim results along with more examples to give a greater feel for the different activities and the progression we envisage.

Acknowledgments

We would like to thank the Education Endowment Foundation for funding this work. We also thank all of the teachers and students from our four design schools for their invaluable contributions to the design and development of the SM intervention.

References

Bers, M.U. et al., 2014. Computational thinking and tinkering: Exploration of an early children robotics curriculum. *Computers & Education*, 72, pp.145–157.
Blackwell, A.F., 2002. What is Programming? In *14th Workshop of the Psychology of Programming Interest Group*. pp. 204–218.
Brennan, K., 2015. Beyond Technocentrism: Supporting Constructionism in the Classroom. *Constructivist Foundations*, 10(3), pp.289–296.
Brown, A.L., 1988. Motivation to learn and understand: On taking charge of one's own learning. *Cognition and Instruction*, 5, pp.311–322.
Bruckman, A., 1998. Community Support for Constructionist Learning. *CSCW (Computer Supported Collaborative Work: The Journal of Collaborative Computing)*, 7, pp.47–86.
Bustillo, J. & Garaizar, P., 2014. Scratching the surface of digital literacy...but do we need to go deeper. In *IEEE Frontiers in Education Conference (FIE)*. pp. 1–4.
Bybee, R.W. et al., 2006. *The BSCS 5E instructional model: Origins and effectiveness*, Colorado Springs, CO: BSCS, 5, pp. 88-98.
Clements, D., 2000. From exercises and tasks to problems and projects unique contributions of computers to innovation mathematics education. *Journal of Mathematical Beh.*, 19(1), pp.9–47.
Clements, D.H., 1999. The future of educational computing research: The case of computer programming. *Information Technology in Childhood Education Annual*, 1, pp.147–179.
DfE, 2013. *Computing Programmes of Study: Key Stages 1 and 2*, DfE. Available at: https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study.
DiSessa, A.A. & Cobb, P., 2004. Ontological Innovation and the Role of Theory in Design Experiments. *Journal of the Learning Sciences*, 13(1), pp.77–103.
Furber, S., 2012. *Shut Down or Restart? The way forward for Computing in UK schools*,
Han, S. & Bhattacharya, K., 2001. Constructionism, Learning by Design, and Project Based Learning. In *Emerging perspectives on learning, teaching, and technology*.
Harel, I. & Papert, S., 1990. Software design as a learning environment. *Interactive Learning Environments*, 1(1), pp.1–32.
Hoyles, C., 1985. What is the point of group discussion in mathematics? *Educational studies in mathematics*, 16(2), pp.205–214.
Hoyles, C. & Noss, R., 1992. *Learning Mathematics & Logo*, Cambridge, MA, USA: MIT Press.
Israel, M. et al., 2015. Supporting all learners in school-wide computational thinking: A cross-case qualitative analysis. *Computers & Education*, 82, pp.263–279.
Lye, S.Y. & Koh, J.H.L., 2014. Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, pp.51–61.
McMaster, K., Rague, B. & Anderson, N., 2010. Integrating mathematical thinking, abstract thinking and computational thinking. In *ASEE/IEEE Frontiers in Ed. Conf*. Washington, DC.
Noss, R. & Hoyles, C., 1996. *Windows on mathematical meanings: Learning cultures and computers*, Springer Science & Business Media.
Papert, S., 1980. *Mindstorms: Children, computers, and powerful ideas*, Basic Books, Inc.
Papert, S., 1972. Teaching Children to Be Mathematicians vs. Teaching About Mathematics. *International Journal of Mathematics Education and Science Technology*, 3, pp.249–262.
Papert, S., 2000. What's the big idea? Toward a pedagogy of idea power. *IBM Systems Journal*, 39(3.4), pp.720–729.
Rader, C., Brand, C. & Lewis, C., 1997. Degrees of comprehension: children's understanding of a visual programming environment. In *Proc. CHI '9*. ACM, pp. 351–358.
Resnick, M. et al., 2009. Scratch: Programming for all. *Comms of the ACM*, 52(11), pp.60–67.
Watt, S., 1998. Syntonicity and the psychology of programming. In *Proceedings of 10th Annual Meeting of the Psychology of Programming Interest Group*. pp. 75–86.
Weintrop, D. & Wilensky, U., 2015. To block or not to block, that is the question: students' perceptions of blocks-based progrmming. In *Proc. IDC '15*. ACM, pp. 199–208.
Wing, J.M., 2008. Computational thinking and thinking about computing. *Philos. Trans. Roy. Soc. London Ser. A: Math., Phys. and Eng. Sci.*, 366(1881), pp.3717–3725.

Constructionism and microworlds as part of a 21st century learning activity to impact student engagement and confidence in physics

Conor M Wickham, *cwickham@tcd.ie*
The Trinity Centre for Research in IT in Education, School of Computer Science & Statistics and School of Education, Trinity College Dublin, the University of Dublin, Dublin 2, Ireland

Carina Girvan, girvanc@cardiff.ac.uk
School of Social Sciences ¦ Cardiff University
Glamorgan Building ¦ King Edward VII Avenue ¦ Cardiff ¦ CF10 3WT, UK

Brendan Tangney, *tangney@tcd.ie*
The Trinity Centre for Research in IT in Education, School of Computer Science & Statistics and School of Education, Trinity College Dublin, the University of Dublin, Dublin 2, Ireland

Abstract

The affordances of microworld simulations to promote student engagement and motivation are well documented in the literature. These technologies which can be highly have the potential to enhance a student's learning experience. Nevertheless their widespread use in mainstream secondary school classrooms remains limited as these technologies do not sit well in conventional classroom settings, where short class durations, didactic pedagogy and an emphasis on teaching to the test prevail.

The problems in secondary school STEM education, such as declining number of students considering a career in science related disciplines, have often been linked to didactic teaching styles in classrooms, with an emphasis on transference of knowledge from the teacher to student and where text books are the main source of curriculum content. In physics, teaching is often focused on the application of mathematical formulae and lacks context and applicability to real world problems. As a result many students find physics a 'difficult and hard subject to study' leading to poor motivation and low engagement with the subject. This research brings three key elements together - microworld technology, a constructionist, contextualised pedagogy and a 21st century learning model – to investigate their combined impact on student engagement and confidence in physics. Students worked in teams using a constructionist microworld simulation to build electrical circuits. An exploratory case study was carried out involving 39 secondary school students (aged ~15/16) participating in 4 separate physics workshops.

An attitudinal questionnaire was used for quantitative data capture, while focus groups and observation provided rich qualitative data for triangulation. The findings from the study indicate positive changes in student engagement, confidence in physics and attitude to the use of technology for learning. The qualitative data provides context for these findings, which while being based on a modest sample in terms of the number of participants and duration of the learning experience, nevertheless support the hypothesis that a 21st century pedagogical approach is a suitable framework for exploiting the potential of microworlds to promote engagement and confidence in physics.

Keywords

Constructionism, Social Constructivism, Microworlds, 21st Century Learning, Bridge21, Engagement, STEM, Contextualised Learning.

1: Introduction

'***Microworld**: An interactive, exploratory learning environment of a small subset of a domain that is immediately understandable by a user and also intrinsically motivating to the user. A microworld can be changed and modified by the user in order to explore the domain and to test hypotheses about the domain.'* Rieber (2005)

Microworlds have been likened to playpens or sandboxes, providing the learner an opportunity for creative exploration and there is extensive literature highlighting the potential of simulations and microworlds to support student engagement and conceptual understanding. Simulations usually provide a representation of physical phenomena, with varying levels of interactivity which allow the user modify, create or alter parameters that will generate a response within the simulation. Several research studies have examined the affordances of simulations to develop deeper conceptual understanding (Girvan, 2014; Martínez, Naranjo, Pérez, Suero, & Pardo, 2011; Perkins, Moore, Podolefsky, Lancaster, & Denison, 2012; Zacharia & de Jong, 2014). In particular the ability of simulations to benefit a learners understanding through visualisation has been demonstrated in science subjects (Blikstein, Fuhrmann, Greene, & Salehi, 2012; Chiu, DeJaegher, & Chao, 2015). While research into constructionist virtual worlds to support creative mathematical thinking points to the importance of collaboration and sharing of created artefacts (Kynigos, Moustaki, Smyrnaiou, & Xenos).

The problems in secondary school STEM education, such as declining number of students considering a career in science related disciplines, have often been linked to didactic teaching styles in classrooms, with an emphasis on transference of knowledge from the teacher to student and where text books are the main source of curriculum content. In physics, teaching is often focused on the application of mathematical formulae and lacks context and applicability to real world problems. As a result many students find physics a 'difficult and hard subject to study' leading to poor motivation and low engagement with the subject. The use of technology in general and micorworlds in particular have the potential to help address at least some of these barriers. However traditional classroom environments, with didactic teaching style and short class durations are not ideal environments in which to exploit the power of learning through technology (McGarr, 2009). Instead the constructivist and constructionist pedagogies at the core of models for 21st century learning (Dede, 2010) may be much more suitable frameworks to capitalise on the power of technology in general and micro-worlds in particular (B. Tangney, Bray, A., & Oldham, E., 2015).

21st Century Learning Environment

Bridge21, a specific model of 21st century learning, has been shown to be an effective environment for technology mediated learning (Bray, Oldham, & Tangney, 2013; Conneely, Girvan, Lawlor, & Tangney, 2015; Johnston, Conneely, Murchan, & Tangney, 2014; B. Tangney & Bray, 2013). In particular in the area of mathematics Tangney et al (B. Tangney, Bray, A., & Oldham, E., 2015) have investigated the combination of mobile technology with contextual and social constructive pedagogies such as Bridge21. The initial results of this 'perfect storm' (as referred to by the authors) are very positive and 'student engagement and appreciation of mathematics content are favourably affected'.

This research study follows a similar approach, but in this case the subject area is physics and investigates the combination of microworld technology, a constructionist, contextualised pedagogy and the Bridge21 learning model. Physics is chosen as the subject domain because of the decline in the number of secondary level students studying physics, which research from several countries (Lyons, 2006) attributes to transmissive teaching styles and a perceived lack of relevance. Even with students who do study the subject research points to low levels of engagement and poor conceptual understanding of the core concepts in physics (Azevedo, 2006; Saleh, 2011).

Research Questions and Goals

The research was undertaken as part of a larger project, looking at the introduction of 21st Century Teaching & Learning in Irish secondary schools (age 12-16), and sought to investigate how microworlds, when used as part of a 21st century learning model, impact student engagement and confidence in physics?

2: Design & Methodology

Overview of the Learning Activity

The microworld selected for use in the study was from the Physics Education and Technology (PhET) project at the University of Colorado[5] and it was chosen because it allows for a high degree of interactivity, has strong construction capabilities and is highly immersive. The Circuit Construction kit is a low floor, medium ceiling, microworld, easy for students to engage with but sufficiently challenging for more advanced users. It is constructionist in its pedagogical approach, allowing for concrete and abstract representations and supports cognitive conflict and investigation of alternate models.

The Circuit Construction Kit microworld was incorporated into a 5 hour Bridge21 workshop at the researcher's institution. 3 separate workshops with ~10 students in each were run and formed the basis of this research sample. This together with an initial pilot workshop consisting of 8 participants gave a total sample size of n=39 students over 4 workshops. The workshops incorporated real world, problem based activities that required participants to construct both series and parallel electrical circuit designs for set of Christmas tree lights and explain their choice and potential design benefits.

The Mathematics and Technology Attitudes Scale (MTAS) developed by (Pierce, Stacey, & Barkatsas, 2007) measures 5 variables related to the learning of mathematics with technology: Mathematics Confidence (MC), Confidence with Technology (TC), Affective Engagement (AE), Behavioural Engagement (BE), and attitude to learning mathematics with technology (MT). The questionnaire was modified for use in this study and was administered pre and post the workshops.

3: Implementation

The 39 students, of mixed gender, came from five different schools. Many of the participants had attended Bridge21 workshops in the past and thus were already familiar with the structure and team collaboration elements. This helped ensure that the investigation could focus on the use of microworlds and was not overly influenced by the novelty of the Bridge21 experience itself.

Data Collection & Analysis

Quantitative data was collected using the modified MTAS questionnaire while qualitative data was gathered through pre and post questionnaires, focus groups, observation and student output during the workshop activities. Qualitative and quantitative data was collected at the same time, both were analysed separately allowing for an element of triangulation to provide deeper insight into the research questions. See Figure 4.

Focus group interviews were run immediately after each workshop and each group comprised of between 4 and 6 participants and the discussion was recorded (using an unobtrusive smartphone recording app). 8 separate focus group discussions were recorded which yielded well over an hour of data to be analysed.

[5] https://phet.colorado.edu/

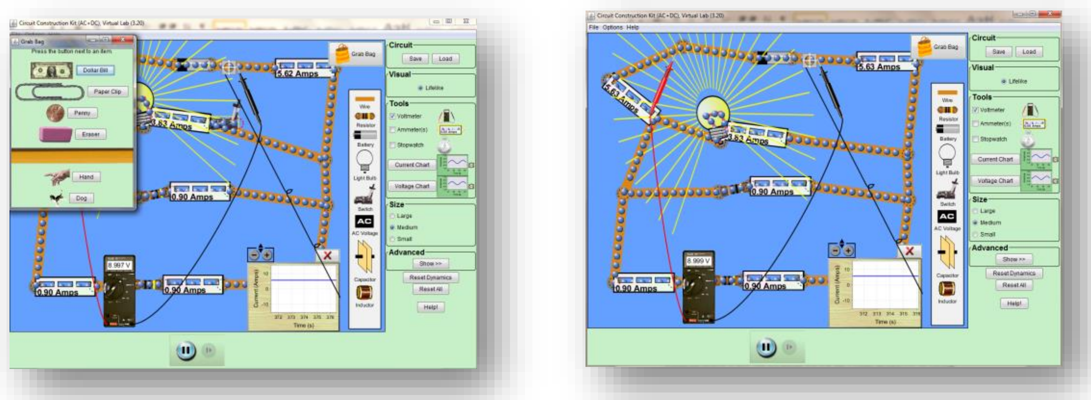


Figure 2: PhET Circuit Construction Kit illustrating complex circuit design. The user has access to accurate measuring tools such as ammeters, voltmeters and results are immediate.

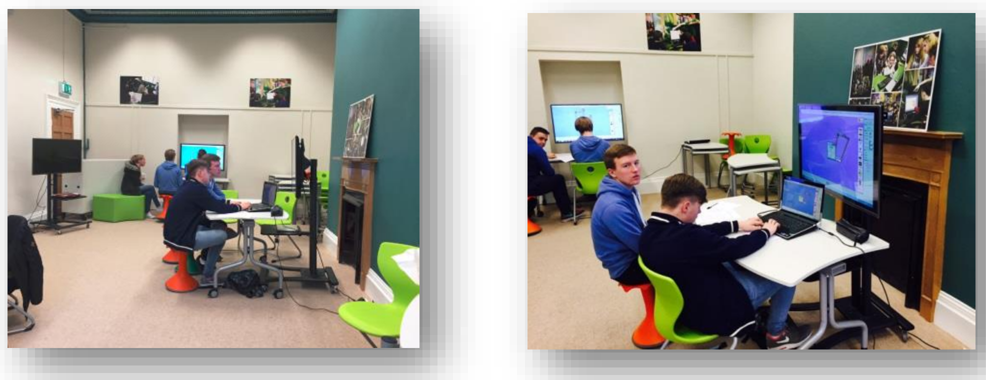


Figure 3: The Bridge21 learning space. Workstations are fully configurable and can be easily rearranged to suit specific needs. Large monitors support sharing within the team.

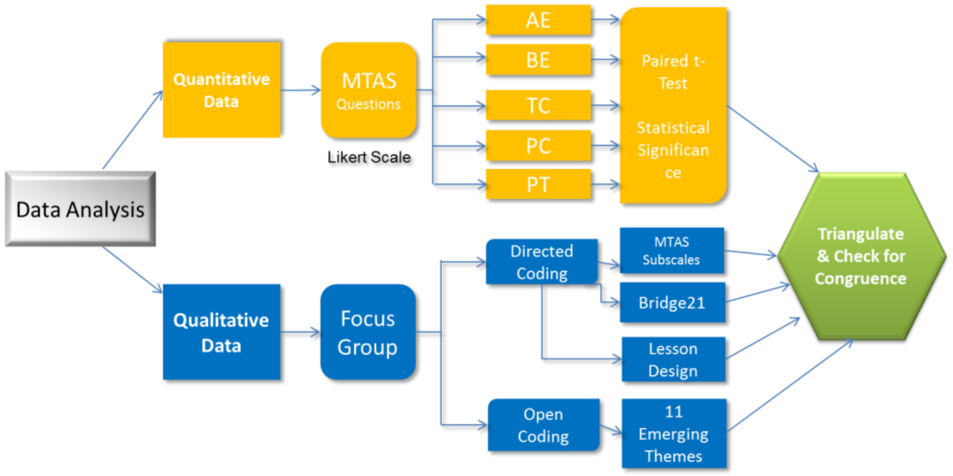


Figure 4: The data analysis framework used in this parallel mixed method case study. Congruence between quantitative and qualitative data supported the research findings.

4: Findings

MTAS Survey – Statistical Analysis

Examining the t values in Table 1 shows that 4 of the 5 MTAS subcategories show significant positive differences after the workshop namely: **Affective Engagement (AE)**, **Behavioural Engagement (BE)**, **Physics Confidence (PC**) and **Attitude to the use of technology in teaching physics (PT)**. For PT (t=6.894) the change was very significant indicating that the participants had a positive reaction to the PhET Circuit Construction Kit microworld. Only Technology Confidence (TC) with t= 1.275 showed no significant change pre and post workshop activity.

		Paired Differences						
					95% Confidence Interval of the Difference			
		Mean	Std. Deviation	Std. Error Mean	Lower	Upper	t	Sig. (2-tailed)
Pair 1	BE Post - BE Pre	.68421	1.67824	.27225	.13259	1.23584	2.513	.016
Pair 2	TC Post - TC Pre	.42105	2.03525	.33016	-.24792	1.09002	1.275	.210
Pair 3	AE Post - AE Pre	1.23684	2.18637	.35468	.51820	1.95549	3.487	.001
Pair 4	PC Post - PC Pre	1.18421	2.16677	.35150	.47201	1.89641	3.369	.002
Pair 5	PT Post - PT Pre	3.92105	3.50584	.56872	2.76871	5.07339	6.894	.000

Table 1: Results of the paired t-test analysis on the pre and post workshop MTAS questionnaires. At 95% confidence interval a t value greater than 2.021 is significant.

Qualitative Data

Both Open and Directed coding techniques were used to analyse the data and the results of directed coding of the focus group data against the MTAS subscales aligns well with the quantitative data. The frequency of occurrence of the open coding themes was highest for a positive reaction to the physics workshop for PT, PC, AE and BE, again in that order. This matches the order of the measured impact from the MTAS data.

Engagement Impact

The research identified several factors that contributed to the increase in behavioural (BE) and affective engagement (AE). These are: Microworld Technology; Collaboration; Self-Discovery.

The constructionist capability of the PhET microworld was a significant factor in the improvement of participants' affective engagement. The PhET microworld allowed students to create their own models and use these to confirm or challenge their understanding of the underlying concepts. The microworld also enabled students to see and understand abstract concepts such as electron flow, capacitor charging, neither of which cannot be easily grasped through textbooks or real physical experimentation.

Collaboration played an important part during the workshops in promoting engagement. Students worked in teams, discussed their solutions and were asked to present back to the whole group and this aspect was seen as positive by the participants.

> *'I think like, when you found the problem and you solved it and the light came on and it worked, and you got passed the problem as a team and you had a working circuit then, I think that was the best bit.'*

Impact on Confidence in Physics

The research findings offer statistical evidence that these workshops made a significant improvement in the students' attitudes to physics and to their confidence with the subject and two main factors contributing to this emerge from a structured analysis of the qualitative data, namely the Bridge21 workshop format (when contrasted with their conventional class experience) and the contextualised nature of the learning.

The contrast between the Bridge21 model with the participants own class experience was very evident from the workshop focus groups as typified by statements such as

> *'Because it wasn't like school. It was like, I don't know—because you didn't tell us we have to do this, and then this and then this. We got to just learn how to do it ourselves, kind of. And it makes you understand it more and it is more fun than just being told what to do.'*

> *'Like with that Ohm thing, I learned that in five minutes. But in school I couldn't even remember it from the book. Because it showed you exactly what was going on with the computer.'*

By contextualising the learning of physics to real life situations during the Bridge21 workshops students had a basis from which to understand the concepts and then apply the underlying mathematics to solve the problem.

> *'Because you know which way the circuits were actually moving. Like you know which way it is all set up. Instead of just looking at picture of piece of wires on the ground. You actually look at the simulation and see how it all fits together.'*

> *'Yeah, I just viewed it as a school subject. But after the workshop you see it is a lot more real-world stuff. '*

Technology Effectiveness

With a t value =6.894, student attitudes to using technology in physics learning (PT) showed the most statistically significant change as a result of the workshops. The choice of microworld technology was a major factor in this improvement. The data suggests several features of the PhET microworld that led to this positive change; visualisation; experimentation and construction capabilities. The microworld allowed students to "do physics" and imagine and create new models in ways that is not possible with existing teaching methods.

> *'I didn't know that electricity was the flow of electrons. You could see the way that they were going around. So that was good.'*

> *'I think if because you could see it and you didn't have to imagine it like you normally did in class and stuff. And then you could move them around and make stuff so it helped.'*

A recurring theme during the focus groups related to the constructionist features of the microworld and participants being able to 'do things'. With the freedom to create, students developed their own models and views of the circuits they constructed. Since the PhET software is based on real electrical principles students often created circuits that did not work or because of voltage irregularities gave unexpected outcomes. This challenged students to understand what was happening and propose a solution.

> *'And doing things in a more interactive, and like hands on way…. start learning things a lot easier when they do it hands on.'*

> *'Because we were actually able to see how a circuit gets put together. And we done a lot of experimentation as well. You know and, I suppose you can be told how something works but unless you, kind of, try it yourself and try different ways of doing things you don't really learn much.'*

Summary

The authors are engaged in a large scale design based research project which is working with teachers and schools across Ireland to embed 21st Century Teaching and Learning practices in mainstream secondary schools (Conneely et al., 2015; Johnston et al., 2014). A core premise of that work is that a 21st century learning model, such as Bridge21, is a more suitable framework for exploiting the potential of technology to support constructivist and constructionist learning than the traditional didactic model. This study looks at student engagement and confidence in physics when following such an approach. Carried out in an out of school setting the data indicates that both engagement with, and confidence in, physics improved as a result of the learning experience. The data also suggests that the attitude towards using technology in the study of physics improved. Factors which contributed to these improvements include the constructionist nature of the simulation, collaboration between peers and the contextualised nature of the learning activity.

The limitations of the study are its short duration, the lack of longitudinal follow-up with the participants to see if the intervention had any lasting impact and the fact that the activity took place in an out of school setting. Nevertheless this study, combined with our other experience of transitioning learning activities from our learning space to schools, gives confidence that the approach followed is transferrable to a real classroom setting and ongoing research will investigate this in practice.

References

Azevedo, F. S. (2006). Personal Excursions: Investigating the Dynamics of Student Engagement. *International Journal of Computers for Mathematical Learning, 11*(1), 57-98.

Blikstein, P., Fuhrmann, T., Greene, D., & Salehi, S. (2012). *Bifocal modeling: mixing real and virtual labs for advanced science learning.* Paper presented at the Proceedings of the 11th International Conference on Interaction Design and Children.

Bray, A., Oldham, E., & Tangney, B. (2013). *THE HUMAN CATAPULT AND OTHER STORIES–ADVENTURES WITH TECHNOLOGY IN MATHEMATICS EDUCATION.* Paper presented at the 11th International Conference on Technology in Mathematics Teaching (ICTMT11).

Chiu, J. L., DeJaegher, C. J., & Chao, J. (2015). The effects of augmented virtual science laboratories on middle school students' understanding of gas properties. *Computers & Education, 85*(0), 59-73. doi:http://dx.doi.org/10.1016/j.compedu.2015.02.007

Conneely, C., Girvan, C., Lawlor, J., & Tangney, B. (2015). An Exploratory Case Study into the Adaption of the Bridge21 Model for 21st Century Learning in Irish Classrooms, in Shaping our Future: How the lessons of the past can shape educational transformation. *Butler D., Marshall K., and Leahy M., Editors, Liffey Press: Dublin.*, p. 348-341.

Dede, C. (2010). Comparing frameworks for 21st century skills. *21st century skills: Rethinking how students learn*, 51-76.

Girvan, C. (2014). Constructionism, Creativity and Virtual Worlds. *Constructionism and Creativity: Proceedings of the 3rd International Constructionism Conference 2014. Vienna, Austria: Austrian Computer Society.*, p367-377.

Johnston, K., Conneely, C., Murchan, D., & Tangney, B. (2014). Enacting key skills-based curricula in secondary education: lessons from a technology-mediated, group-based learning initiative. *Technology, Pedagogy and Education*(ahead-of-print), 1-20.

Kynigos, C., Moustaki, F., Smyrnaiou, R., & Xenos, M. HALF-BAKED MICROWORLDS AS EXPRESSIVE MEDIA FOR FOSTERING CREATIVE MATHEMATICAL THINKING.

Constructionism and Creativity: Proceedings of the 3rd International Constructionism Conference 2014. Vienna, Austria: Austrian Computer Society.

Lyons, T. (2006). Different Countries, Same Science Classes: Students' Experiences of School Science in Their Own Words. *International Journal of Science Education, 28*(6), 591-613.

Martínez, G., Naranjo, F. L., Pérez, Á. L., Suero, M. I., & Pardo, P. J. (2011). Comparative study of the effectiveness of three learning environments: Hyper-realistic virtual simulations, traditional schematic simulations and traditional laboratory. *Physical Review Special Topics - Physics Education Research, 7*(2), 020111.

McGarr, O. (2009). The development of ICT across the curriculum in Irish schools: A historical perspective. *British Journal of Educational Technology, 40*(6), 1094-1108. doi:10.1111/j.1467-8535.2008.00903.x

Perkins, K., Moore, E., Podolefsky, N., Lancaster, K., & Denison, C. (2012). Towards research-based strategies for using PhET simulations in middle school physical science classes. *AIP Conference Proceedings, 1413*(1), 295-298. doi:10.1063/1.3680053

Pierce, R., Stacey, K., & Barkatsas, A. (2007). A scale for monitoring students' attitudes to learning mathematics with technology. *Computers & Education, 48*(2), 285-300. doi:http://dx.doi.org/10.1016/j.compedu.2005.01.006

Rieber, L. P. (2005). Multimedia learning in games, simulations, and microworlds. *The Cambridge handbook of multimedia learning*, 549-567.

Saleh, S. (2011). The Level of B.Sc.Ed Students' Conceptual Understanding of Newtonian Physics. *International Journal of Academic Research in Business & Social Sciences, 1*(3), 249-256.

Tangney, B., & Bray, A. (2013). Mobile Technology, Maths Education & 21C Learning. *QScience Proceedings*(12th World Conference on Mobile and Contextual Learning [mLearn 2013).

Tangney, B., Bray, A., & Oldham, E. (2015). Realistic Mathematics Education, Mobile Technology & The Bridge21 Model For 21st Century Learning – A Perfect Storm. In H. T. J. Crompton (Ed.), *Mobile Learning and Mathematics: Foundations, Design and Case Studies* (pp. 96-105): Routledge.

Zacharia, Z. C., & de Jong, T. (2014). The Effects on Students&apos; Conceptual Understanding of Electric Circuits of Introducing Virtual Manipulatives within a Physical Manipulatives-Oriented Curriculum. *Cognition and Instruction, 32*(2), 101-158.

Constructionism as making construals: first steps with JS-Eden in the classroom

Antony Harfield, *antonyh@nu.ac.th*
Department of Computer Science & Information Technology, Naresuan University, Thailand

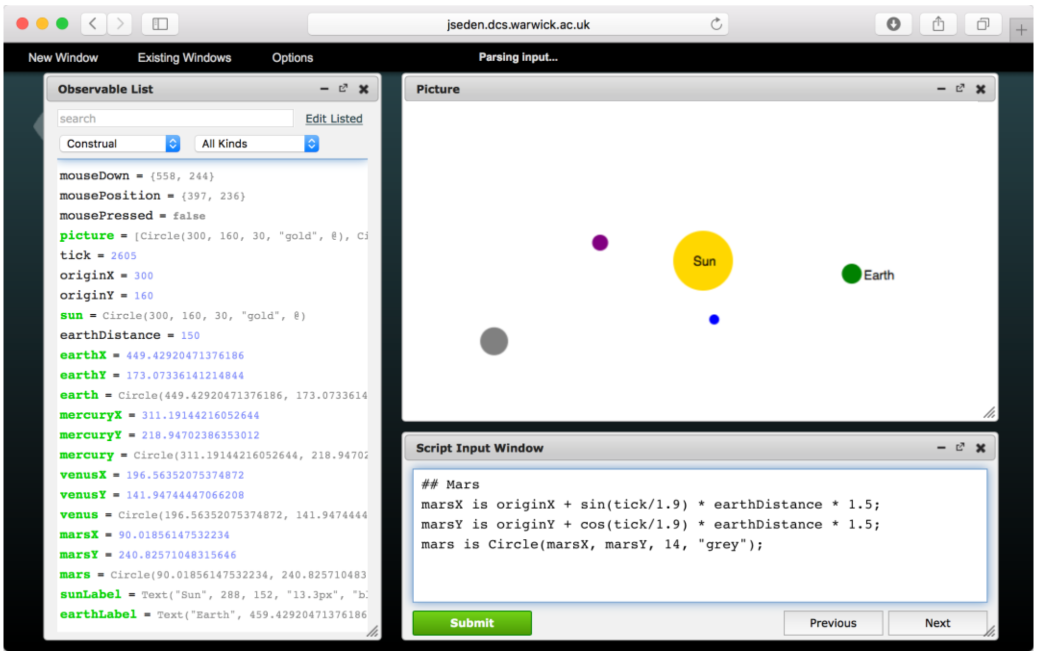
Rene Alimisi, *info@edumotiva.eu*
Edumotiva-European Lab for Educational Technologies, Greece

Peter Tomcsányi, *tomcsanyi@fmph.uniba.sk*
Department of Informatics Education, Comenius University, Bratislava, Slovakia

Nick Pope, *nick@dcs.warwick.ac.uk*
Meurig Beynon, *wmb@dcs.warwick.ac.uk*
Department of Computer Science, University of Warwick, UK

Abstract

JS-Eden is an environment for learners to build software artefacts that relies on construction by 'making construals' using observables and dependencies. JS-Eden is proposed as an alternative to procedural or object-oriented constructionist environments. In this paper we present a small experiment in which JS-Eden is introduced to 25 high school students. The observations and feedback suggest that although there are improvements to be made to JS-Eden's user interface for learners, the principles of constructionism by making construals can be readily applied in a classroom for domain learning. Comparisons are drawn with existing constructionist environments, and it is argued that making construals in JS-Eden is a better paradigm for children engaged in the "instructing", "animating" and "modulating" activities that are key in working with digital media.



```
mouseDown = {558, 244}
mousePosition = {397, 236}
mousePressed = false
picture = [Circle(300, 160, 30, "gold", @), Ci
tick = 2605
originX = 300
originY = 160
sun = Circle(300, 160, 30, "gold", @)
earthDistance = 150
earthX = 449.42920471376186
earthY = 173.07336141214844
earth = Circle(449.42920471376186, 173.0733614
mercuryX = 311.19144216052644
mercuryY = 218.94702386353012
mercury = Circle(311.19144216052644, 218.94702
venusX = 196.56352075374872
venusY = 141.94744447066208
venus = Circle(196.56352075374872, 141.9474444
marsX = 90.01856147532234
marsY = 240.82571048315646
mars = Circle(90.01856147532234, 240.825710483
sunLabel = Text("Sun", 288, 152, "13.3px", "b
earthLabel = Text("Earth", 459.42920471376186
```

```
## Mars
marsX is originX + sin(tick/1.9) * earthDistance * 1.5;
marsY is originY + cos(tick/1.9) * earthDistance * 1.5;
mars is Circle(marsX, marsY, 14, "grey");
```

Figure 1. The solar system construal.

Keywords

Constructionism, programming, learning technology, Empirical Modelling, making construals

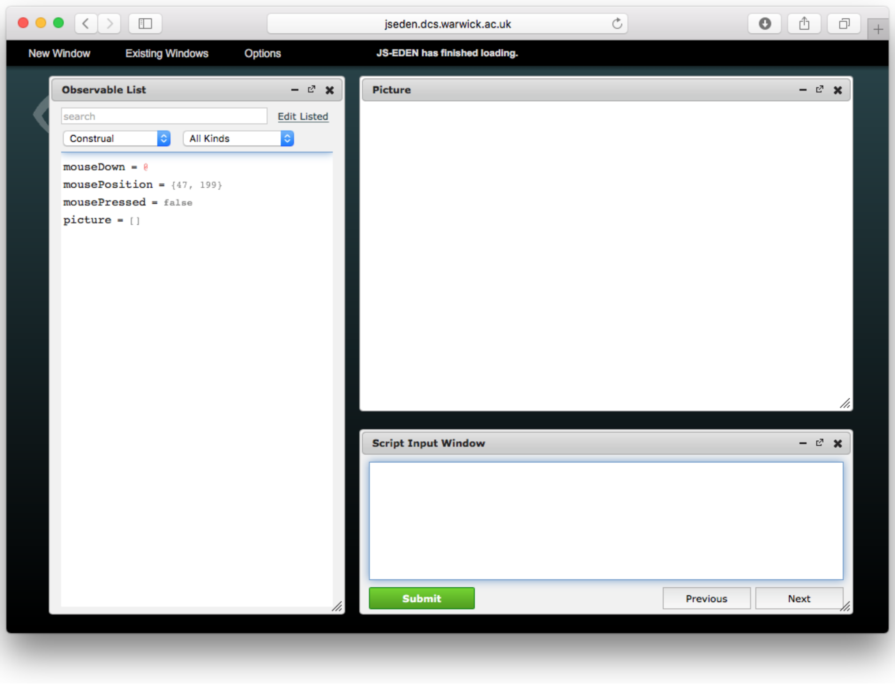
Introduction

A recurring tenet of Papert's constructionism is that children should not learn programming for the sake of programming. Instead, children can use programming to create contexts in which a wide variety of learning experiences might occur (Papert, 1980). Many of the most popular constructionist programming environments today (e.g. Alice, Greenfoot, Scratch) take a computer scientist's view of programming as procedural or object-oriented (Utting et al., 2010) and, due to the challenging nature of programming, the learning often focuses on the tool or language. While there are advocates of computational thinking as a learning outcome in itself (Wing, 2006), Papert's vision for constructionism was clearly much greater. Brennan contends that, although our classroom technologies have changed, we are still "stuck in a technocentric view" (Brennan, 2015). On the other hand, Ackermann points out that a digital native's concept of 'programming' spans a broader range of activities involving "instructing", "animating" and "modulating" (Ackermann, 2012). Taken together, these critiques suggest that the traditional procedural view of programming is not only unsatisfactory for supporting current learning activities but also insufficient for the vision of construction as a way to learn in any domain. In this paper we propose to support constructionism by 'making construals' – an activity that is more general than conventional programming, and promotes a less technocentric view more closely aligned with the activities that Ackermann identifies. Our claim is that this non-procedural, non-object-oriented approach is more accessible to young people while still supporting the breadth of constructionist activities in which today's digital natives are engaged.

The idea of 'making construals' originates in a reconceptualisation of computing called Empirical Modelling (EM) that is radically different from the paradigm of computational thinking (Beynon, 2012). Rather than focusing on writing programs to achieve specific functional goals, EM puts its primary emphasis on making construals which are built up incrementally and have states and transitions closely connected with an object of study in the learning domain. The characteristic guiding principle of EM is that the learner should be able to directly experience a correspondence between the state of a construal and the state of an external object. The learner works as a modeller, building the construal by creating observables and dependencies. If we consider a learning situation where the object is the motion of planets in the solar system then the observables might be the Sun, the Earth and the position of the Earth in relation to the Sun. Dependencies are connections made between observables, for example how the position of the Earth depends on the position of the Sun and the day of the Earth year.

JS-Eden for making construals

Construals are created by making connections using observables, dependency and agency (Beynon, 2012). JS-Eden is a web-based environment for creating construals that has been developed by researchers at University of Warwick (Beynon et al., 2015). JS-Eden has previously been utilised as a constructionist tool for high school students to build construals for learning mathematics (Harfield and Beynon, 2012). Construals are constructed by entering script-like definitions into JS-Eden which interprets the definitions on the client-side of the web browser. The primary interface for JS-Eden consists of a Script Input Window, an interactive canvas ("Picture") and an observable inspector ("Observable List"). A developer, or teacher, or learner, constructs a construal in a progressive manner, by adding or modifying observables, dependency and agency on-the-fly without recompiling. Unlike many procedural programming environments which have a two stage process of creating and running, JS-Eden is interpreted and constantly "live", representing state as it is in the current moment. JS-Eden can be used to build models from scratch, and to exercise or 'remix' existing models. The focus of this paper is on the use of JS-Eden for building simple models from scratch. Figure 2 depicts the typical starting screen of JS-Eden showing an 'empty' construal.



```
mouseDown = @
mousePosition = {47, 199}
mousePressed = false
picture = []
```

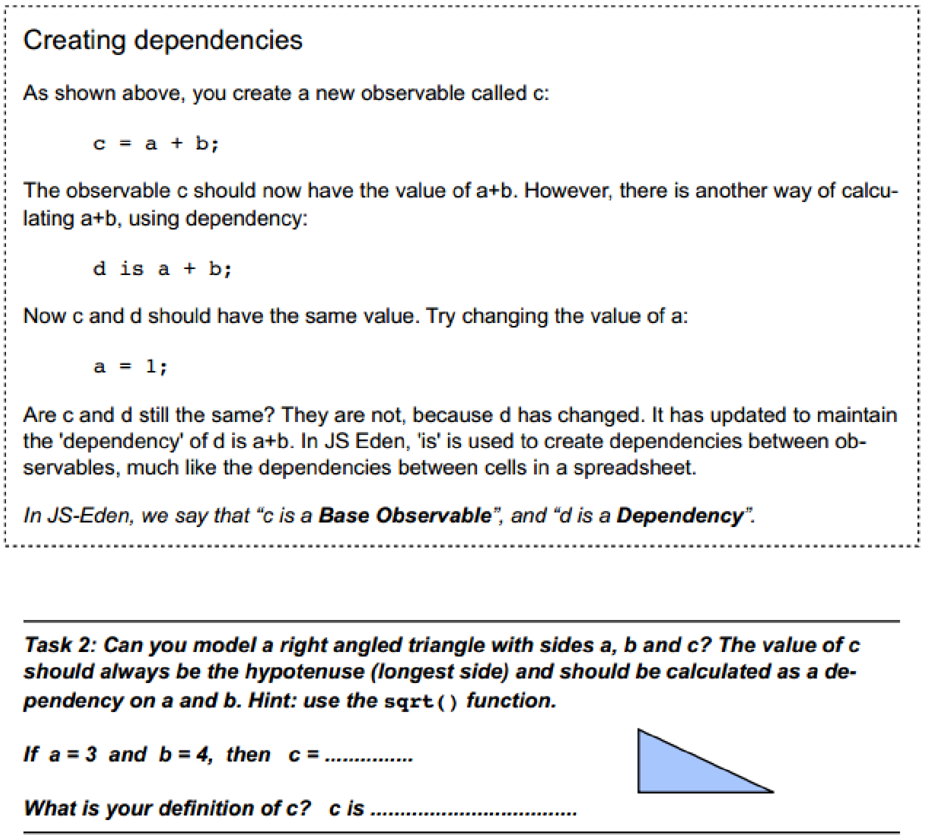
Figure 2. JS-Eden, ready to start a construal making activity.

Early JS-Eden trial in school

CONSTRUIT! is an on-going project to develop online resources for making construals. As part of the project, a workshop was organised in Athens, Greece, of which one part was to test the viability of teaching using JS-Eden in the classroom. The main aim of this experiment was to provide teachers with a model class and to analyse student reactions to the JS-Eden environment and the learning material. Furthermore, the experiment was a way to test the claim of this paper that JS-Eden is an accessible environment for digital natives that supports a breadth of constructionist activities.

The workshop was held at 2nd Experimental Lyceum of Athens, Greece, for 3 teaching periods (approximately 2.5 hours) on Tuesday 22nd September 2015. The school was chosen because the teachers and students have experience with trying new approaches and technologies. The participants were 15-16 year old students. There were 25 students in total, and they mostly worked in pairs. Two teachers and a number of assistants and observers from the CONSTRUIT! Team were present during the session.

The students had no prior experience of JS-Eden. The workshop was designed to introduce the basics of making construals in JS-Eden and help the students become familiar with the kind of things they could build with JS-Eden. The aim of the workshop was not to teach programming or computational thinking. At the beginning of the workshop, no more than 5 minutes were spent explaining the 3 main components of JS-Eden (the Script Input Window, Picture, and Observable List as shown in Figure 2) with a short example. The students were then given a worksheet that explained what to do next and walked them through the basic building blocks of the environment. Each topic involved at least 1 question. In total there were 11 tasks in the worksheet.



Creating dependencies

As shown above, you create a new observable called c:

```
c = a + b;
```

The observable c should now have the value of a+b. However, there is another way of calculating a+b, using dependency:

```
d is a + b;
```

Now c and d should have the same value. Try changing the value of a:

```
a = 1;
```

Are c and d still the same? They are not, because d has changed. It has updated to maintain the 'dependency' of d is a+b. In JS Eden, 'is' is used to create dependencies between observables, much like the dependencies between cells in a spreadsheet.

In JS-Eden, we say that "c is a ***Base Observable****", and "d is a* ***Dependency****".*

Task 2: Can you model a right angled triangle with sides a, b and c? The value of c should always be the hypotenuse (longest side) and should be calculated as a dependency on a and b. Hint: use the `sqrt()` ***function.***

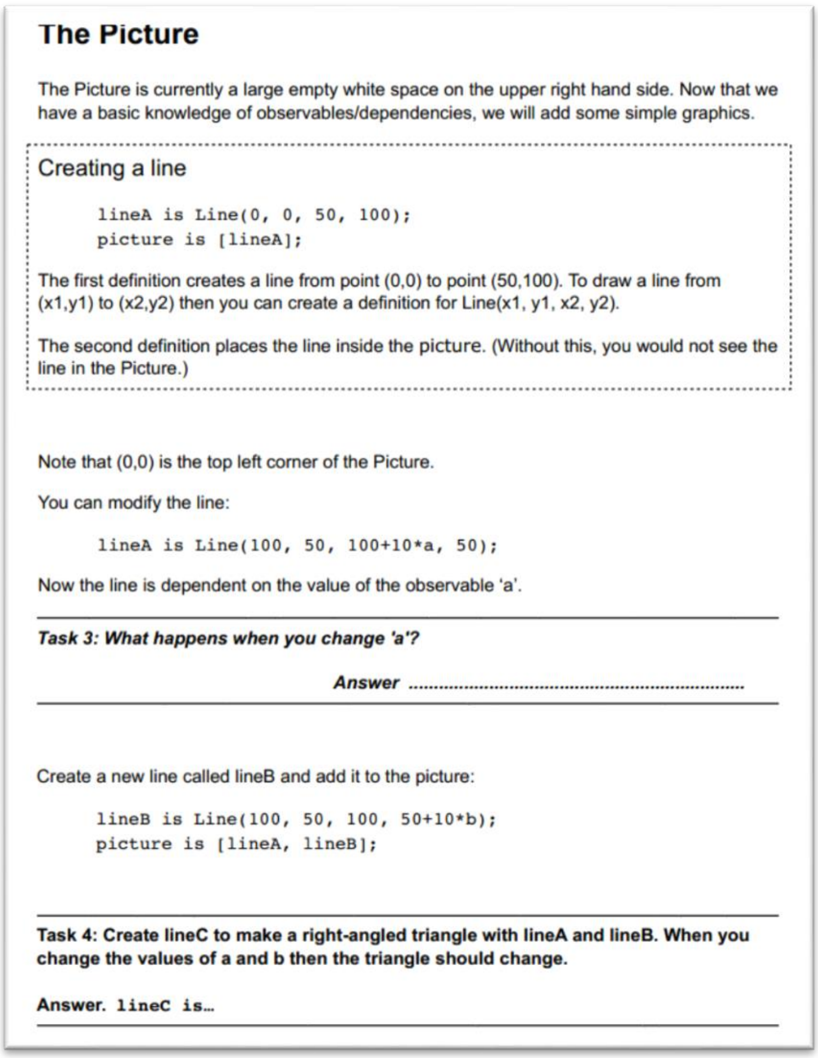
If a = 3 and b = 4, then c = ...

What is your definition of c? c is ....

Figure 3. Introduction to dependencies and the Pythagoras's theorem task.

Figure 3 shows the part of the worksheet where the notion of dependency was explained for the first time (the part framed by dashed line). Some of the observers were able to see the "Aha! effect" in some groups of students when they observed that the value of observable d changed after changing the value of observable a. This is an important concept for the students to grasp as it is unrelated to procedural programming. The immediately following Task 2 was oriented not only to practice the new knowledge of dependency, but also to use some of their existing mathematical pre-knowledge in a new situation. Figure 4 shows the next part of the worksheet which introduces the "picture" observable and the Line graphic object. Tasks 3 and 4 concentrate on both creating graphics in JS-Eden and the use of dependencies. They both also relate to drawing a triangle that illustrates Pythagoras's theorem featured in Task 2. Later in this paper we will discuss the outcomes of Tasks 2-4 in more detail.

Figure 5 shows an example of one of the later tasks. This relates to building a model of the solar system, where the students must create dependencies that position the Earth dependent on the day of the year.



The Picture

The Picture is currently a large empty white space on the upper right hand side. Now that we have a basic knowledge of observables/dependencies, we will add some simple graphics.

Creating a line

```
    lineA is Line(0, 0, 50, 100);
    picture is [lineA];
```

The first definition creates a line from point (0,0) to point (50,100). To draw a line from (x1,y1) to (x2,y2) then you can create a definition for Line(x1, y1, x2, y2).

The second definition places the line inside the picture. (Without this, you would not see the line in the Picture.)

Note that (0,0) is the top left corner of the Picture.

You can modify the line:

```
    lineA is Line(100, 50, 100+10*a, 50);
```

Now the line is dependent on the value of the observable 'a'.

Task 3: What happens when you change 'a'?

Answer ....

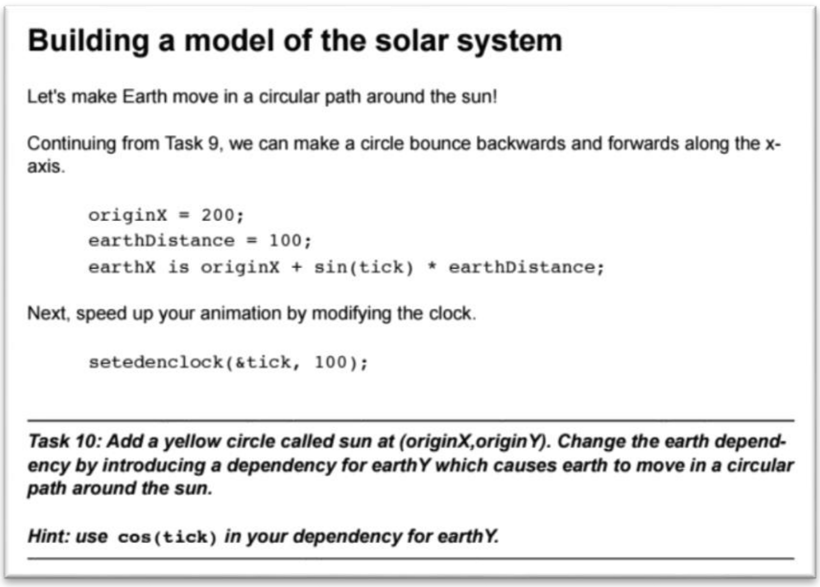
Create a new line called lineB and add it to the picture:

```
    lineB is Line(100, 50, 100, 50+10*b);
    picture is [lineA, lineB];
```

Task 4: Create lineC to make a right-angled triangle with lineA and lineB. When you change the values of a and b then the triangle should change.

Answer. `lineC is…`

Figure 4. Introduction to graphics and using the knowledge about dependencies in a new context.



Building a model of the solar system

Let's make Earth move in a circular path around the sun!

Continuing from Task 9, we can make a circle bounce backwards and forwards along the x-axis.

```
originX = 200;
earthDistance = 100;
earthX is originX + sin(tick) * earthDistance;
```

Next, speed up your animation by modifying the clock.

```
setedenclock(&tick, 100);
```

Task 10: Add a yellow circle called sun at (originX,originY). Change the earth dependency by introducing a dependency for earthY which causes earth to move in a circular path around the sun.

Hint: use `cos(tick)` ***in your dependency for earthY.***

Figure 5. Later task on building a model of the planets in the solar system.

By the end of the workshop the students could have created a fully animated solar system with 4 planets moving around the sun at their correct relative velocities. The status of the construal after completing all the tasks in the workshop is shown in Figure 1. The three definitions in the Script Input Window represent the dependencies required to create a circle for Mars and animate it so that it moves around the Sun in a circular path. The first 2 lines define the x and y position of Mars, which are dependent on the position of the Sun (*originX* and *originY*), the day of the Mars year (based on a factor of *tick*) and the distance from the Sun to Mars (approximated as *earthDistance * 1.5*).

Results of the trial

During the session, the students' interactions with the environment were recorded by capturing their submitted inputs in the background. The recordings of student interactions showed that all of the student groups reached task 8 out of a total 11 tasks. The last 3 tasks related to building the solar system model, and while more than 75% of the students made some progress on these tasks, only one group was able to complete the final task (which was purposefully difficult and open-ended to push the students who found the earlier material easy). The later tasks involved the use of sine and cosine which had not been taught as part of their regular classes which may have slowed the progress of some students. However, the assistants were active in helping students through the difficult exercises which contributed to the high completion levels.

Examples of the struggles and breakthroughs that the students experienced can be found in the recordings. Consider the tasks 2 (shown on Figure 3), 3 and 4 (both shown on Figure 4) which involved drawing 3 lines to make a triangle and required the students to apply Pythagoras' theorem in order to complete the tasks. Figure 6 shows three extracts of the recordings relating to tasks 2-4: the script entered by the students is displayed in bold and the comment above shows the timestamp of JS-Eden session (e.g. *26m 16s* is 26 minutes and 16 seconds into the session). Note that it is not evident from the scripts whether the students received any assistance during these periods – hence their breakthroughs could be the product of an individual, the group, intervention from an assistant, or a combination of all three.

In the case of Group A, after 26 minutes the students seem to be considering how to use the square root function (sqrt) to solve task 2 which requires them to calculate the hypotenuse of a right angled triangle using Pythagoras' theorem ($a^2 + b^2 = c^2$). After 30 minutes they enter their definition of c as sqrt(a*a) + sqrt(b*b). This is a valid JS-Eden definition, but mathematically it is wrong. After a further 3 minutes they have realised their mistake and entered a new definition for c (perhaps through their observation of the values of a, b and c in the Observable List). *Group A make a mathematical error that they are able to observe and correct.*

The observations of Group B start with them making some mistakes about how to make a definition. They might be new to programming as they struggle with the equals operator between the 26th and 28th minutes. They make a further error in the 30th minute, but their understanding of Pythagoras's theorem is correct. It takes a further 2 minutes for them to refactor to obtain a syntactically correct definition for c (although it uses "c =…" instead of "c is…" which means there is no dependency). *Group B have a correct mathematical understanding, but struggle to represent it in the language of JS-Eden.*

In contrast to Group B, Group C obtain a syntactically and mathematically correct definition of c on their first attempt at task 2. However, in task 4 (after 29 minutes) they are initially unable to apply their knowledge to create the line of the triangle representing the hypotenuse. After 31 minutes, Group C is trying to use the sqrt function to create the line that connects LineA and LineB (mixing up the mathematics of triangles with the drawing of triangles). At 41 minutes they have mastered the syntax for drawing the line and finally at 49 minutes they find a definition that satisfies the requirements to create the triangle. *Group C have a misconception about how to draw the lines of a triangle, but through experimentation they are able to make sense of the problem and solve it.*

This small sample of extracts demonstrates how the students overcame three struggles: a misunderstanding of Pythagoras's theorem, a misrepresentation of Pythagoras's theorem in JS-Eden and a misconception about how Pythagoras's theorem relates to drawing triangles in JS-Eden.

Group A	**Group C**

## 26m 16s 	
a = sqrt(9);
 	
## 26m 43s 	
b = sqrt(16);
 	
## 30m 31s 	
c is sqrt(a*a) + sqrt(b*b);
 	
## 33m 17s 	
c is sqrt(a*a+b*b);
	## 15m 12s
	c is sqrt(a^2 + b^2) ;
	
	## ...
	
	## 29m 8s
	LineA is Line(100, 50, 100+10*a, 50);
	
	## 31m 36s
	LineC is sqrt(LineA^2 + LineB^2) ;
	
	## 32m 28s
	LineC is sqrt (LineA^2 + LineB^2);
	
	## 32m 44s
	LineC is sqrt(LineA^2 + LineB^2) ;
	
	## 37m 51s
	LineC is Line(100, 50, 100, 130) ;
	
	## 40m 29s
	LineC is Line(100, 50, 130, 100);
	
	## 41m 39s

Group B ## 24m 33s **a = 3;** ## 24m 45s **b = 4;** ## 26m 46s (error) **c = c * c;**	

## 27m 24s 	
c = a+b;
 	
## 28m 3s (error) 	
c*c=;
 	
## 30m 58s (error) 	
c*c=a*a+b*b;
 	
## 32m 17s 	
c = sqrt(a * a + b * b);
	LineC is Line(100, 50, 100+10*a, 50);
	
	## 43m 52s
	LineC is Line(100 , 50+10*b)^2 + (100+10*a , 50)^2;
	
	## ...
	
	## 49m 58s
	LineC is Line(100, 50+10*b, 100+10*a, 50);

Figure 6. Three interaction extracts that demonstrate the difficulty faced by the students in applying Pythagoras' theorem.

Immediately after the session, the teacher was interviewed, specifically to understand her reaction to the activities undertaken by the students. The teacher expressed the view that the students reacted well to the activity, which was evident from the fact that their attention was undisturbed for 3 periods when usually they would have taken breaks. The teacher said that they would like to continue using this tool in their classes. They observed that it is different style of 'programming' to Scratch and Logo (with which they have some experience) because it has more reliance on typing and entering definitions. For this reason, they felt it was closer to traditional programming that requires textual input. The example given by teacher was that it is similar to Prolog in the way that it is an interpreted, definition-based language.

Feedback was also obtained from the CONSTRUIT! project members who were present during the session. They highlighted problems with the JS-Eden environment which mostly focussed on the difficulties that students had with the lack of suitable error messages and poor feedback. This is possibly an indication that the problems were not conceptual, but more a consequence of JS-Eden's migration from being an expert's tool to a learner's tool. The observers noticed that the students appeared to be experimenting with the behaviour of the environment, either in a focused way, or by random 'prodding' to understand how it works.

Several of the observers noted that the tasks involved the students applying their basic mathematical knowledge (e.g. modulo arithmetic, Pythagoras's theorem, and sine/cosine). According to observers, some students were not aware or could not access this knowledge. Were the students unaware of the required mathematical knowledge? This might be a possible explanation – although the class teacher stated that these mathematical topics have been taught and they are integral parts of the school mathematics curriculum. Another possible explanation may be related to the context of the activity – students might have had an expectation that they were programming and therefore they did not treat the exercise as being related to 'doing maths'. In other words, being 'hooked' in their perception of the context of the activity, they were not fully prepared for an exposure to an interdisciplinary learning experience.

Other observers said that although they struggled to apply their mathematical knowledge within JS-Eden, the students were not afraid to use the environment to try out their answers (e.g. Group C's line drawing experiments in Figure 6). In a previous study (Beynon and Harfield, 2010), a high school student commented that you don't actually need much knowledge of the Eden environment to use it to find solutions to the tasks.

A week after the session, students were asked to reflect on their experience in the form of a student diary. The diary requested students to comment on "what went well", "what did not go well", "what did you like the most", and "what did you like the least (dislike)". Of the 25 students in the class, 20 student diaries were collected. Many of the positive comments were focussed on the environment, the group of experts and the English practice. The most negative comments were that there was too much content to cover in the time available, that the level of difficulty was too high, that the environment contained bugs or user interface issues, and that using English was a barrier to communication.

Some of the students noticed the synergy with software development and expressed enthusiasm in building programs: “Finally we did some serious programming!” and “I was involved in serious programming tasks!”. Many of these students had already been exposed to Scratch or Logo, and so their comments seem to corroborate the teacher’s view that JS-Eden has some characteristics that make it feel more like “grown-up” programming than other educational environments.

Although the students said that the tasks were difficult, their diaries suggest that the tasks that the students enjoyed were some of the most difficult. One female student commented that “the animation” was what they liked the most, and another female student said “the planets moving around the sun was the one that I mostly liked”.

The students recognised that the tasks were difficult, but they were mostly able to overcome obstacles, and the fact that they could do this was satisfying. “It was very difficult but we made it” and “We followed the worksheet, made notes on it and we managed to understand the programming language”. Another student complained that one task “did not go well because it was very difficult, I struggled at finding a solution and I did not submit the task on time”, but after that he said, “I liked it when we were making the figures. It was fun and enjoyable!”. Is this a contradiction? No, we think that it is an excellent case of the ‘hard fun’ Papert proposed as the essence of good education (Caperton, 2005).

Discussion: JS-Eden vs procedural constructionist environments

While the positive results of the trial no doubt share common elements with studies of other popular constructionist environments, there are noticeable conceptual differences in their approaches to creating software. Most constructionist environments have a necessary programming component, either procedural or object-oriented, that must be mastered (e.g. Scratch). The three activities noted by Ackermann’s are evident in these environments, as part of a two stage process of designing and executing. “Instructing” involves writing some instructions and then running; the “animating” and “modulating” activities are the same. JS-Eden offers a more direct approach as there is no two stage process to these activities. Interaction in JS-Eden is more like using a spreadsheet than writing a traditional program. When a learner “instructs” something to happen in JS-Eden, that is all they do because they are simply creating a connection. When a learner is “modulating”, they are also tweaking a connection directly. This ‘directness of interaction’ is a characteristic feature of making construals with JS-Eden.

Within the short time of the workshop, the students undertook a number of tasks that involved applying existing mathematical knowledge to solve the problem of drawing and animating objects on the screen. One example was when they first struggled to apply Pythagoras’s theorem in practice, but were able to realise and correct their mathematical misunderstandings, syntactic errors and tool misconceptions with the help of JS-Eden. Admittedly, their realisation was supported by a team of assistants who were there to provide hints and act as scaffolders in the learning process. However, the important point is that JS-Eden enabled students to tackle their domain learning within a relatively short period of time and with little prerequisite knowledge of programming.

Some of the students were able to complete the animation of the Earth revolving around the Sun in the final tasks. A Logo or Scratch equivalent of rotating the Earth around the Sun might involve first finding the starting position in relation to the Sun and then looping over a set of actions that include moving forward a small step and turning by a small amount (e.g. 1 degree). Furthermore, to animate more than one planet with the correct relative speeds would involve a significant number of extra calculations. In JS-Eden, the students were able to create these animations by making connections between the day of the year and the position of the planets in the construal. For example, if the Earth moves once around the sun in 365 days then after 182 days it will be approximately half way (or 180 degrees), whereas Mars (with an orbit period of 687 days) has moved less than a quarter of its orbit. The connections made in JS-Eden have a more meaningful association with the domain than the individual procedural steps of moving an arbitrary small distance forward and turning. Being able to find and model a 'meaningful association' closely is one of the principles behind JS-Eden.

Concluding remarks

This small-scale study presents a setting where students were introduced to JS-Eden and 'making construals' for the first time. They were asked to use dependency to make connections between observables, to create simple construals that are representative of a situation or domain problem, and to reflect upon their experiences. While the students reported that the activities were challenging, the evidence suggests that they overcame their difficulties either in their group or with assistance. The interaction recordings demonstrated that the challenge may in part have been the friction between learning how to make construals and applying domain knowledge from mathematics. In some way, this can be taken as a positive in that there was a reasonably low barrier of entry to learning JS-Eden, and that domain learning is achievable after a short introduction to the tool. It provides some support for the claim that the making construals approach is accessible to young people for constructionist activities.

Furthermore, the making construals approach offers a more accessible environment for the kind of 'programming' activities of digital natives that were defined by Ackermann. Firstly, "instructing", "animating" and "modulating" are all activities that are involved in the direct manipulation of observables and dependencies in JS-Eden. Secondly, "instruction", "animating" and "modulating" are activities that might evoke a meaningful association between the software in the computer and a situation or object in the world. While there is typically a good correspondence between the products of computational thinking andprocedural tasks (e.g. performing a sequence of steps), the approach taken by JS-Eden of making connections between observables is well-suited to modelling a wide variety of everyday situations (cf. the movement of one object rotating around another). It is the principles of directness of interaction and meaningful association that we wish to emphasise in proposing making construals with JS-Eden as an attractive alternative to the technocentric constructionist tools that rely on computational thinking.

References

Ackermann, E. K. (2013) *Programming For The Natives: What is it? What's In It For The Kids?* Constructionism 2012, National & Kapodistrian University of Athens, Greece.

Beynon, M. et al. (2015) *Making construals as a new digital skill: dissolving the program - and the programmer - interface*. Proceedings of the Interactive Technology and Games (ITAG) conference, Nottingham Trent University, 22-23 October 2015.

Beynon, M. and Harfield, A. (2010) *Constructionism through Construal by Computer*. Constructionism 2010, The American University of Paris, August 2010

Beynon, M. (2012) *Modelling with experience: construal and construction for software*. Chapter 9 in Ways of Thinking, Ways of Seeing (ed. Chris Bissell and Chris Dillon), Automation, Collaboration, & E-Services Series 1, Springer-Verlag, January 2012, ISBN 978-3-642-25208-2, 197-228.

Brennan, K. (2015) *Beyond Technocentrism: Supporting Constructionism in the Classroom*. Constructivist Foundations 10(3): 289–296.

Caperton, I. (2005) *For Seymour Papert "hard fun" is the essence of good games AND good education*. Telemedium: the journal of media literacy. 52 (1 & 2) 16-19. Madison, WI: National Telemedia Council.

Harfield, A. and Beynon, M. (2012) *Empirical Modelling for constructionist learning in a Thai secondary school mathematics class*. In the 9th International Conference on eLearning for Knowledge-Based Society (eLearningAP 2012), Siam Technology College, Thailand, 13-14th December 2012.

Papert, S. (1980) *Mindstorms: Children, computers and powerful ideas*. Basic Books, New York, USA.

Utting, I., Cooper, S., Kölling, M., Maloney, J., and Resnick, M. (2010) *Alice, Greenfoot, and Scratch: A Discussion*. ACM Transactions on Computing Education, Vol. 10, No. 4, Article 17, November 2010.

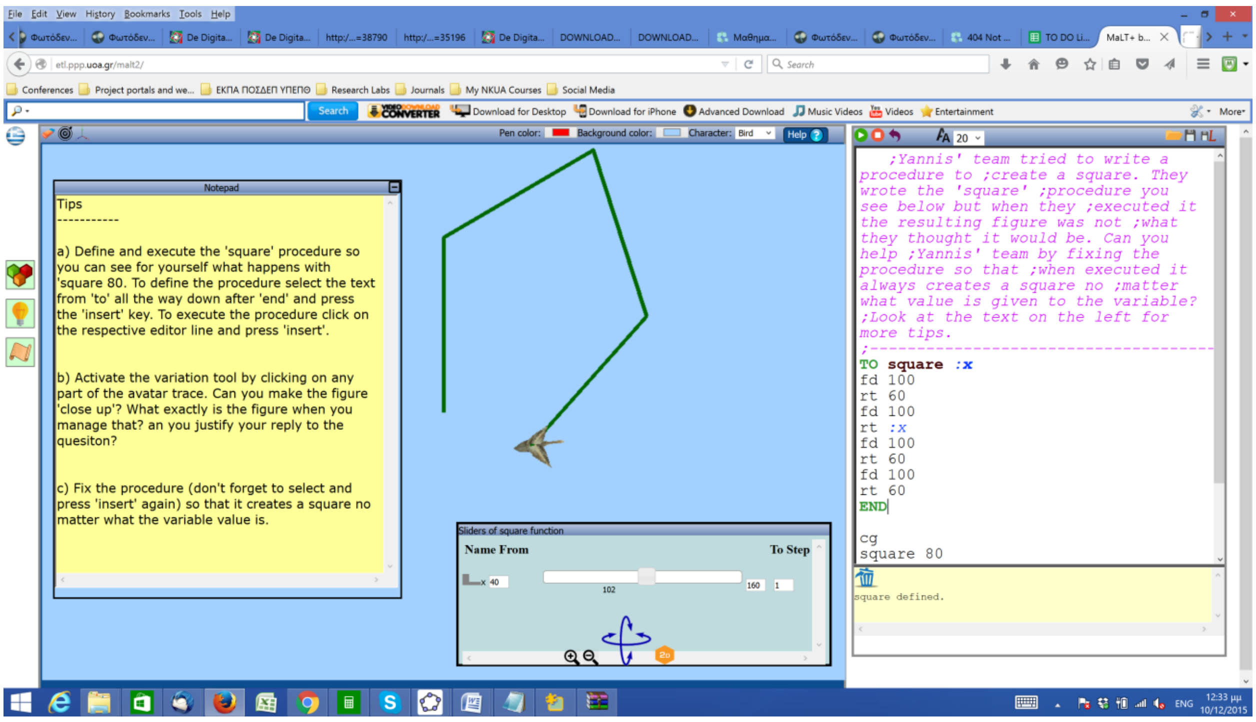
Wing, J. M. (2006) *Computational thinking*. Communications of the ACM 49 (3): 33.

Constructionist activity with institutionalized infrastructures: the case of Dimitris and his students.

Chronis Kynigos, *kynigos@ppp.uoa.gr*
Educational Technology Lab, Sch. of Philosophy, National Kapodistrian University of Athens and CTI-Diophantus

Abstract (style: Abstract title)

The paper discusses the case of Dimitris, a secondary mathematics teacher, who selected three micro-experiments from an institutionalized portal, re-mixed them and then gave his version to his students who in turn made their own changes and constructions. The case is discussed in the frame of the potential for institutionalized portals and digital infrastructures to afford constructionist activity for educators, designers, teachers and students.



Notepad

Tips
----

a) Define and execute the 'square' procedure so you can see for yourself what happens with 'square 80. To define the procedure select the text from 'to' all the way down after 'end' and press the 'insert' key. To execute the procedure click on the respective editor line and press 'insert'.

b) Activate the variation tool by clicking on any part of the avatar trace. Can you make the figure 'close up'? What exactly is the figure when you manage that? an you justify your reply to the quesiton?

c) Fix the procedure (don't forget to select and press 'insert' again) so that it creates a square no matter what the variable value is.

```
   ;Yannis' team tried to write a
procedure to ;create a square. They
wrote the 'square' ;procedure you
see below but when they ;executed it
the resulting figure was not ;what
they thought it would be. Can you
help ;Yannis' team by fixing the
procedure so that ;when executed it
always creates a square no ;matter
what value is given to the variable?
;Look at the text on the left for
more tips.
;------------------------------------
TO square :x
fd 100
rt 60
fd 100
rt :x
fd 100
rt 60
fd 100
rt 60
END

cg
square 80
```

```
square defined.
```

Figure 1. A 'micro-experiment' on an institutionalised digital portal

Keywords (style: Keywords)

keyword; institutionalized portals, constructionist re-mixes, half-baked, micro-experiments

Introduction

For around ten years now, scratch has been paving the way for a number of emerging infrastructures for constructionist activity addressing mostly informal settings and gaining popularity based on their own merit. At the same time infrastructures for access to digital media in institutionalized settings such as ministries of education and organizations working on their behalf have grown but mostly seem to be indifferent to supporting coherent pedagogical reform and innovation. The agenda there is mostly to accrue the largest possible volume of resources provided they have a basic accreditation mainly with respect to the validity of and the rights to the embedded information. Very often, these portals give mixed messages with respect to the role and uses of digital artefacts, emphasizing video narratives, links to encyclopedic information, tightly defined exercises usually to be resolved by multiple choice questions and finally some simulations affording a simple experiment.

In this paper I'd like to raise the question of how might it be made possible to embed the potential of supporting constructionist activity in these institutionalized infrastructures without disputing their agendas to democratize access to information and easily understood affordances but at the same time seeding affordances for educators, teachers, designers and students to make structural changes to artefacts therein. To use Chevallard's metaphor, I ask whether it is possible to go to a human knowledge exhibition centre and place some exhibits allowing visitors to engage in producing their own knowledge (Chevallard, 2012). I take the case of the 'Digital School' infrastructure of the Greek Ministry of education created in the past four years and in particular two co-existing portals, the 'Interactive books' portal and the 'Photodendro' portal at http://photodentro.edu.gr/aggregator/?lang=en. The former contains the original unique curriculum books enriched by the inclusion of links to a variety of artefacts in amongst the text and in tight relation to it. The latter is a classic portal with carefully organized meta-data for each of the artefacts which began by containing the artefacts of the interactive books portal and has been growing with more since, but with no connection to the curriculum books.

My role in the design and development of this infrastructure was to coordinate the design and development of artefacts for the domain of mathematics from year 3 through 11. I worked with a team of 30 professionals selected so as to have diverse expertise, comprising of technical knowledge, pedagogical design knowledge and mathematical knowledge (Fischer, 2012, Kynigos, 2007). In a course of four years, an impressive number of 1800 original artefacts were developed and uploaded in the two portals, almost all of them constituting what we call 'micro-experiments', i.e. tightly focused microworlds, objects with which students can experiment and dynamically manipulate some simulation or problem embedding mathematical concepts in order to discuss and answer in the classroom a set of closed questions and occasionally a final open question involving invitation to some constructionist activity (Kynigos, 2012). To orchestrate a sound integration of the knowledge and experience inherent in the design team and also to create the conditions for creative designs, I made sure that each of artefacts was created by 2-3 designers with diverse expertise and internally reviewed by another team member in a way visible to all designers (for a discussion, see Clinton & Hokanson, 2012, Gero, 2010). The underlying authoring tools were 'Geogebra', 'MaLT' - a 3D web version of e-slate 'Turtleworlds', (Kynigos, 2004)- and some custom widgets built with flash and other tools.

The question I'd like to discuss in this paper is how may it become possible for this kind of infrastructure to support constructionist activity. I thus asked Dimitris, a mathematics teacher with a masters degree in mathematics education and some (but not extensive) experience with constructionist media, to pick a small number of artefacts, change them and then give them to his students to engage in mathematical activity.

The paper revolves around three examples of Dimitris' work (see Kynigos 2007b for a background discussion). The corresponding artefacts will thus be discussed with respect the way they were originally designed and the way they may be used by people who play different roles, such as the role of the designer, the educator and the student (Kynigos, 2002). The main perspective and pursuit is to talk about the fact that Photodentro and the interactive books portal can play the role of the resource, of the available infrastructure, but it can also play the role of a springboard for design, creation and development for all the people involved in education, from the educator to the student (Pepin et al, 2013, Guedet & trouche, 2012).

The context of large scale initiatives

Firstly, for Mathematics we had the possibility and the opportunity from very early on to work in conjunction with other major initiatives of the Ministry that happened to take place around the same time. One of those initiatives was the committee for the new curriculum, which for the first time placed great emphasis on mathematical activity, that is what is proposed for students to do in order to become personally engaged with the concepts of mathematics by actually utilizing them and have a mathematical experience. Additionally, great emphasis was been placed in this new curriculum on mathematical literacy, that is the approach to mathematics as an important societal asset, a cultural tool concerning everyone’s actual life and not the mere abstract end-product of a scientific field. The second major concurrent initiative of the Ministry was what was named 'second level Training program', which supports in-service teachers to make use of the two portals and all the infrastructure to a very large degree, focusing on mathematics education, which is the blending of new and established methods, techniques and teaching practices http://b-epipedo2.cti.gr/. Thus, the 'Digital School' initiative was for us the third field of contribution and intervention, the one that constitutes the infrastructure in terms of available material.

Beyond any doubt there are great many matters that are addressed with these infrastructures, such as the availability of textbooks or resources online for everyone, from anywhere and at any time. What we were interested in the domain of mathematics was to look for added pedagogical value that may be involved in utilizing them, what the educator or the student can do that would otherwise be very difficult to be done without these technologies. This was our focus for the subject of mathematics. Reinforcing the possibility for students to have a personal experience of mathematical reasoning in situations that are realistic for them, by utilizing the available infrastructures as the tool for expressing concepts of mathematics by using them in the context of mathematical literacy. The interweaving between the three major actions was especially important for us as to eliminate the confusion that is often caused by the feeling of fragmentation between intervention actions in the field of education.

The quest for added pedagogical value

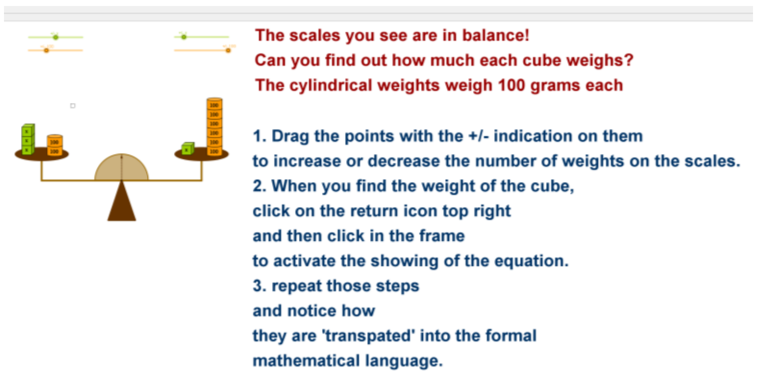
So, first a few words about the ways in which we designed the portals to afford added pedagogical value that comes as a result of this intervention, and immediately afterwards we will see the first example. In the paper there are three examples, of three artefacts, that incorporate different technologies and concepts of mathematics and I will describe the ways that they have been constructed by the designers, by Dimitris and by his students for each of them to express concepts of mathematics.

Students can utilize these infrastructures to strengthen their mathematical experience and the feeling that mathematics is something that is realistic, interesting, fun as well as beautiful and mainly that it’s something useful for our everyday lives. The added value is not only about what the student can do. The added value also concerns the designer as well as the educator with the role of the designer. It is a springboard for the design and development of artefacts, not only by specialists and researchers but also by the educators themselves. In the team of mathematics, we were 35 specialized colleagues and all the artefacts of mathematics in the Photodentro, around 1800 artefacts, were developed by Greek creative minds and hands. This team of colleagues consisted of technicians and educators that had overall knowledge of pedagogy, mathematics and technology. Not even a single artefact was developed without having all these three aspects of know-how underlying, by the people that made it. Beyond this highly specialized team, this infrastructure also provides the educator himself with the possibility of modifying his own artefacts, design activities for his students and his own repertoire can be contained in these activities, his own “suitcase” or “la valise” -as the French call it- of digital artefacts that are modifications of the ones that can be found at the Photodentro. And these possibilities also exist for trainers and councilors for them to strengthen and create seminars of training and practice between colleagues who discuss matters of didactics.

Emphasizing equivalence to solve an equation: Dimitris' re-mix of a 'scales' simulation

Let me show you what I mean by that by means of a first example. Dimitris chose a classic equation problem embedded in a scales task residing in the 8th grade interactive textbook (and also in the Photodendro portal). In the paper version the task refers to a picture of the scales. In the interactive book, it is dynamically manipulable, affording the adding or taking away of known and unknown weights and the simulation of a balanced or titled scale (fig. 2). So, the idea is that students will experiment and come up to a solution involving the isolation of one unknown weight on one scale and the number of known weights needed to balance the scale. They are supposed to connect this with the process of solving an equation. The digital tools available in this simulation are sliders that change the number of known and unknown weights on each one of the two sides of the scale. Dimitris however had other ideas regarding how to use it in order to make the mathematics more interesting for his students and changed the artefact completely, he created a modified version (fig. 3). In his version of the simulation, the semantics for the weights and their measure are not iconic, the user imagines the weights inside two pots (for a discussion of artefacts as representations, see Morgan & Kynigos, 2014). The semantics are numerals for the number of weights and color for known (red) and unknown (blue) weights, all changeable by means of respective sliders. Up till now, the simulation appears to be the same just with different, more abstract, representations. Things change dramatically however by a textual question asked to the student: “is there any chance that the scale is faulty”? Dimitris wanted to get the students to work that one out and then find the value of the unknown weight too. Let’s see now how this artefact was used by the students. One way that the children themselves thought in order to deal with this was to put the same number and combination of weights on each side, despite the fact that the blue weight was unknown. They then added known weights until it balanced to find out the extent of the fault. In this case, their thinking dealt with the unknown value as an object which is at the heart of mathematical thinking. It means that they could cope with imagining that the number had been found and additionally accepting that the scale is broken and there is no balance. So, the students found that the scale had an error of 40 kilograms since it was balanced when two red weights of 20 kilograms each were added to the equivalent weights. Afterwards, in their effort to find how much the unknown blue weights weighed, a strategy that a group of kids thought of was to put the scale in balance and then increase the number of blue weights by one and then see how many reds they needed to add in order for it to become balanced again and in that way figure out the solution. Dimitris' agenda was for the students to see the value of equivalence and appreciate that it lies behind the concept of equation. The students' changes allowed them to be able to use a faulty balance to work out their equation problem and estimate the extent of the fault. Although maybe in this case they did not change the functionality of the simulation, they did understand it to an extent allowing them to resolve a problem.

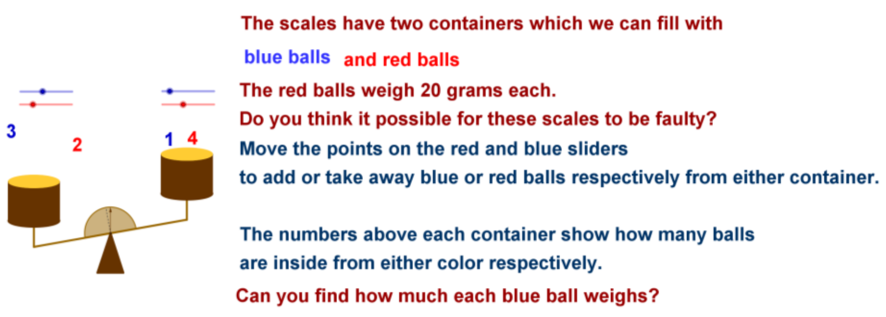
With respect to the process of designing the original artifacts for the two portals, special care was taken so that each one emerged out of the collaboration of 2-3 designers with diverse expertise, at least one with technical know-how and one with pedagogical design experience. First and foremost special care was taken so that the mathematics embedded in the artifact was correct in essence and in the way is was conveyed by means of its representation. Equivalently, the developers were asked to clearly negotiate the pedagogical agenda for its use, i.e. to anticipate what the students would do with it as an expressive tool. Thus, although the development of the two portals was funded simply to play the role of a resource, for the domain of mathematics it was almost exclusively designed as a set of artefacts for the students to do mathematics with, i.e. to experiment, to modify, to think around and to justify behaviors, properties and the changes they made.



The scales you see are in balance!
Can you find out how much each cube weighs?
The cylindrical weights weigh 100 grams each

1. Drag the points with the +/- indication on them
to increase or decrease the number of weights on the scales.
2. When you find the weight of the cube,
click on the return icon top right
and then click in the frame
to activate the showing of the equation.
3. repeat those steps
and notice how
they are 'transpated' into the formal
mathematical language.

Figure 2. The 'scales' micro-experiment



The scales have two containers which we can fill with
blue balls and red balls
The red balls weigh 20 grams each.
Do you think it possible for these scales to be faulty?
Move the points on the red and blue sliders
to add or take away blue or red balls respectively from either container.

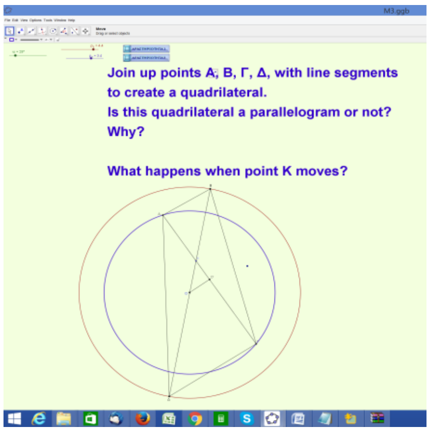
The numbers above each container show how many balls
are inside from either color respectively.
Can you find how much each blue ball weighs?

Figure 3. Dimitris' 'scales' experiment

Constructionism for all in a classic geometrical problem

The second example shows how students made structural changes to an artefacts already changed by Dimitris. We now have a classic geometrical example, which is in the interactive textbook (fig.4), two concentric circles are given, the two corresponding diameters and students are asked to tell what kind of quadrilateral is formed if the four intersection points of the diameters and the circles are connected and also justify their answer. Next, they are asked, by increasing and decreasing the lengths of the diameters and changing their direction to make various quadrilaterals and justify their constructions. But Dimitris wanted more in terms of the possibility of engaging his students in mathematical thinking. So, he made two line segments, where one of them was a diameter and the other was a mere chord of an arc whose size could be modified. He asked his students if a parallelogram is formed, as well as when and why. He asked them to explain what is happening on the shape and construct different parallelograms of their own choice. We can see that the questions are more open, they invite students to create quadrilaterals and to explain how a figure does not belong to a particular class when the properties necessary for it to do so do not apply.

Now let's look at some students' activity. A certain student thought of connecting these two midpoints, one of the diameter and the other of the chord, O and O’, and observe what happens to the OO' segment in order to provide his explanation. This is especially important, since the student felt that he had the right and that it is part of his role to tamper with the software, to add a line segment which will help him think and then dynamically manipulate it


Join up points A, B, Γ, Δ, with line segments to create a quadrilateral.
Is this quadrilateral a parallelogram or not?
Why?

What happens when point K moves?

Figure 4. Dimitris' geometry experiment

. It is also important that in order for such a thing to happen, the teacher needs to create and encourage this norm in the classroom, i.e. that these tools are tools for experimentation and engagement as well as modification and tampering. In support of this, let us look at the things the children were saying while using the tool. We have a student who says "this quadrilateral is not a parallelogram since the opposite sides are not parallel and equal". So far she answers the question. But, without being asked, she goes on to say, "if the point K is moved along the diameter that is there, the length of the chord, from which this point passes through, will change. So, as K comes closer to D, the length of the side BG increases and of GD decreases". This conclusion was not part of the question. It is however a mathematical formulation, a conjecture and a topic for thought in class, that this particular student felt it is a valid thing to do, it's within the norm, within reason to do so. Despite the fact that there was a specific question, children felt free to think of the mathematics around the question and not just answer it and move on to the next matter and the next problem. These digital infrastructures, are thus infrastructures that encourage and allow children to express themselves, to attribute meaning in what they're doing. They include interdependent representations as is shown in the first example, which is a very important matter in mathematics, they help children become detached from the representation and understand that there are concepts behind it. They allow dynamic manipulation, which is a new way to represent concepts in mathematics as shown in the second example, a way that was not available before. They allow teachers and students alike to have deep access to functionalities. In the first two examples, Dimitris modified the scale simulation and took away a property of one segment (diameter) to drastically change a mathematical problem. The students respectively changed the functionality of the experiment by employing equivalence to resolve an equation and by adding a segment to help them study a property.

Employing Algebra for a Geometrical Construction

The third example is about how we can combine concepts of mathematics that lie in different sections in the curriculum, to such an extent so that they are mistakenly considered to be unconnected. It is also about how students can for themselves invent ways to use mathematical concepts. The respective artefact from the digital school is for the 7th grade and it is about the rhombus and the square, apparently geometry again. This artefact incorporates mathematical expression through computer programming, it is made with a new web-based 3D version of E-slate Turtleworlds which we call 'Turtlesphere' which we developed at the Educational Technology Lab. Students are put in the position of an engineer, and are asked to de-bug a model that one of their fellow students has apparently made and that is not working correctly. "Yiannis' team", we read in the problem text (fig. 1), "tried to make a procedure to construct a square, without success, can you help Yannis fix the procedure so that when it is executed a square is always constructed"? Students can experiment with the sliders by altering the variable values dynamically, look at the code to figure out what change is needed, which property of the square is missing in this procedure. What is missing here is that the angles should be 90 degrees. They get into the formalism of mathematics, correct the code, run the procedure and observe if it works or not. What I wanted to discuss first is Dimitris' modification (fig. 5 first column). He turned this problem to be about parallelograms providing one that doesn't work in order for the students to experiment and find out, think of what they need to do to make this shape become correct once again. After a lot of discussion, the children managed to realize that the angles of the turns are supplementary and that this suffices for the parallelogram to be fixed and nothing else is required. Dimitris then gave them another problem, he asked if they can make a rectangle, that can never be a square. The students came up with various strategies For example, they put a variable x for one of the two opposite sides and they used x+20 in one case and 2*x in another, for the other side (fig. 5, columns 2 and 3). What did the children do here? In order to express and make a model, without anyone telling them, they thought of incorporating the linear function as an element of changing and modifying the model. With this artefact the students themselves came up with an algebraic solution to a geometrical problem and incorporated a linear relationship between two generic numbers, the length of the opposite sides.

To parellelogram :x :y	To change :x	To myrectangle :x
Fd 50	fd :x+20	Fd :x
Rt 30	rt 90	Rt 90
Fd :y	fd :x	Fd :x*2
Rt :x	rt 90	Rt 90
Fd 50	fd :x+20	Fd :x
Rt 30	rt 90	Rt 90
Fd 100	fd :x	Fd :x*2
Rt 150	Rt 90	Rt 90
end	end	end

Figure 5. Students' constructions as responses to Dimitris' task

A short discussion

So this is a way in which institutionalized digital infrastructures could afford added pedagogical value for students and also for teachers, designers and teacher educators and consultants. The teacher can be assisted by these infrastructures in order for his practice to acquire a higher element of orchestration by means of using them to generate norms for mathematical literacy. The portal can also be used by teachers to engage in their own research and professional reflection. The design element in the teaching profession can also be enhanced in interesting ways given that these digital media can be used as expressive media for design. Teachers can create their own artefacts by remixing the ones given in the portal like Dimitris and thus act as a members of communities that discuss and share such activities and such materials.

Consequently, the artefacts now become springboards for student constructions, for design, for creation of such artefacts by the educator and for engagement in communities of educators.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 610467 - project “M C Squared”, http://mc2-project.eu. The c-book technology is based on the widely used Freudenthal Institute's DME portal and is being developed by a consortium of nine partner organisations, led by CTI&Press 'Diophantus'.

This work has been supported by the Greek National project "Digital School Platform, Interactive Books, and Learning Object Repository" (Contract No 296441/2010-2015) that is co-financed by the European Union (ESF) and National funds in the context of Operational Programme "Education and Lifelong Learning" of the Greek National Strategic Reference Framework (NSRF), and is being implemented by CTI "Diophantus".

References

Chevallard, Y. (2012). *Teaching Mathematics in Tomorrow's Society: a Case for an Oncoming Counterparadigm*. Plenary lecture at 12th International Congress on Mathematical Education, Seoul, Korea**.**

Clinton, G., Hokanson, B., 2012. Creativity in the training and practice of instructional designers: the Design/Creativity Loops model. *Educational Technology, Research and Development, 60*, 111–130. DOI:10.1007/s11423-011-9216-3

Fischer, G. (2012). Meta-Design: Empowering all Stakeholders as Co-Designers. R. Luckin, P. Goodyear, B. Grabowski, S. Puntambeker, J. Underwood, & N. Winters (Eds.), *Handbook on Design in Educational Research*. Routledge

Gash, H. (2014) Constructing Constructivism, Constructivist Foundations, Sp. Issue: Forty Years of Radical Constructivism in Educational Research, Riegler. A. and Steffe, L., P. (Eds), ISSN 1782-348X, v.9(3), 302-310.

Gero, J.S., 2010. Future directions for design creativity research. In T. Taura & Y. Nagai (Eds), *Design Creativity 2010* (pp.15-22), Springer.

Gueudet, G., and Trouche, L. (2012). Communities, documents and professional geneses: interrelated stories. In G. Gueudet, B. Pepin, & L. Trouche (Eds.), *Mathematics curriculum material and teacher documentation: from textbooks to lived resources (*pp. 305-322). New York: Springer.

Healy, L. Kynigos, C. (2010) Charting the microworld territory over time: design and construction in learning, teaching and developing mathematics *The International Journal of Mathematics Education, ZDM,* Springer Verlag, 42. 63-76.

Kynigos C. (2002). Generating Cultures for Mathematical Microworld Development in a Multi-Organisational Context. *Journal of Educational Computing Research*, Baywood Publishing Co. Inc. (1 and 2), 183-209.

Kynigos, C. (2004). A Black and White Box Approach to User Empowerment with Component Computing, *Interactive Learning Environments*, Carfax Pubs, Taylor and Francis Group, Vol. 12, Nos. 1–2, 27–71.

Kynigos, C. (2007) Half-Baked Logo Microworlds as Boundary Objects in Integrated Design, *Informatics in Education,* 2007, Vol. 6, No. 2, 1–24, Institute of Mathematics and Informatics, Vilnius.

Kynigos, C. *(*2007b*)* Half-baked Microworlds in use in Challenging Teacher Educators*'* Knowing, *international Journal of Computers for Mathematical Learning.* Kluwer Academic Publishers, Netherlands, 12 *(*2*)*, 87-111*.*

Kynigos, C. (2012). Niches for Constructionism: forging connections for practice and theory, *Proceedings of the 'Constructionism 2012' International Conference*, C. Kynigos, J. Clayson, N. Yiannoutsou (Eds.), Athens, 40-51.

Morgan, C., Kynigos*.* C. *(*2014*))* Digital artefacts as representations*:* forging connections between a constructionist and a social semiotic perspective*.* Special Issue in Digital representations in mathematics education*:* conceptualizing the role of context and networking theories**,** *Educational Studies in Mathematics,* Lagrange, J.B*.* and Kynigos, C*. (*Eds*)*, Springer Science + Business Media, Dordrecht, 85 *(*3*)*, 357-379*.*

Pepin, B., Gueudet, G., & Trouche, L. (eds.) (2013). Re-sourcing teacher work and interaction: new perspectives on resource design, use and teacher collaboration, special issue of ZDM, The International Journal on Mathematics Education, 45 (7), 929-944.

Constructionist Learning at the Group Level with Programmable Badges

Corey Brady, *cbrady@northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

David Weintrop, *dweintrop@u.northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Gabriella Anton, *gabby.anton@u.northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Uri Wilensky, *uri@northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Abstract

This paper reports on early-stage design research oriented towards engaging *groups* of learners in computationally-rich constructionist activities. The work we present here focuses on computer science (CS) instruction, but the approach is applicable across Science, Technology, Engineering and Mathematics (STEM) disciplines. To enable constructionist learning at the group level, we have created a new computational tool: a programmable and open-hardware electronic badge. In collaboration with Parallax, Inc., a leading educational robotics company, we are developing these badges as a research platform that foregrounds social and interactive dimensions of learning. In this paper, we introduce the CCL-Parallax badge, outline our design motivations, situate the badges within constructionist literature, and describe some of our early activities using them, which fall into three broad categories: embodied participatory simulations, computational systems simulations, and social and distributed maker activities.

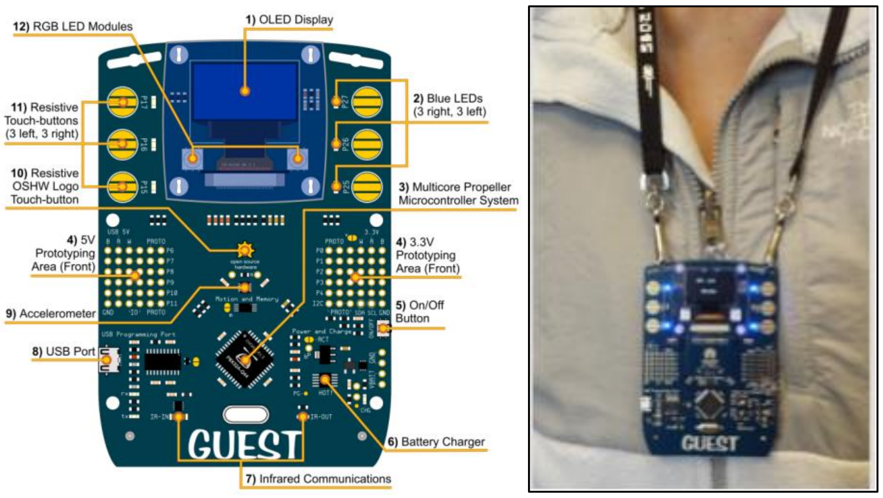

12) RGB LED Modules
1) OLED Display
11) Resistive Touch-buttons (3 left, 3 right)
2) Blue LEDs (3 right, 3 left)
10) Resistive OSHW Logo Touch-button
3) Multicore Propeller Microcontroller System
4) 5V Prototyping Area (Front)
4) 3.3V Prototyping Area (Front)
9) Accelerometer
5) On/Off Button
8) USB Port
6) Battery Charger
7) Infrared Communications
GUEST

Figure 1. The CCL-Parallax Programmable Badge.

Keywords

Participatory Simulations; Embodied Modelling; Open Hardware; Computer Science Education

Technology for Constructionist Learning at the Group Level

Can a group of learners engage in collective learning activities that are constructionist in nature? If so, what does this look like, and what technologies are needed to support this kind of activity?

We argue that this is indeed possible, and we are pursuing design research to illuminate this space of possibilities. In this paper, we describe several categories of activity that are constructionist both for individuals and for groups as collective learning entities. To support ongoing research in this area, we have developed a computing platform centered on wearable, programmable, and hardware-hackable devices: the CCL-Parallax Programmable Badge (Figure 1). These badges are tuned to support activities that make learning social, shifting images of computation away from work with self-enclosed black-boxes, toward distributed mobile devices and peer-to-peer interactions. Moreover, the badges' wearable nature encourages embodied learning, and their openness at the hardware and software levels enables a range of open-ended construction activities. Our work revives and unifies themes that have appeared throughout the history of research into using mobile computing devices for learning. One important thread emphasized distributed, embodied learning in groups, while another focused on the power of open, simple, and mobile computers to support programming and robotics. Much of this work has built on Papert's constructionist vision of computing and learning (Papert, 1980). Within this broad vision, however, researchers have diverged in the design and educational applications of mobile, personal, programmable devices. With this work, we try and bring those threads back together.

Twenty-five Years of Wearables and PartSims

Early forms of wearable computing had few affordances for learning: they were used primarily in corporate settings to increase efficiency and security, and to provide location tracking of employees (Want & Hopper, 1992; Want et al., 1992). These early badges employed an asymmetric communications model, positioning their wearers as passive objects to be *labeled*. However, a targeted set of improvements enabled a radical expansion of the affordances of wearables. The next wave of research in this area focused on supporting badge wearers in symmetric peer-to-peer and badge-to-badge communications. Making a corresponding shift in metaphor—from badge as *label* to badge as *nametag*—researchers at MIT created Thinking Tags and later, Meme Tags, which were used at conferences to augment conversations between people. These smart nametags were also programmable in a rudimentary way, opening the door to user customization and personalization (Borovoy et al., 1996; Borovoy et al., 1998).

Education research also began exploring how such activities could be used to engage learners in authentic practices of scientific inquiry. In particular, Wilensky and Resnick began exploring how group-centered, embodied activities could illuminate multi-agent computational and social systems. In the late 80's, they created a massively parallel Logo environment, *StarLogo*, which extended single-turtle Logo, to support simulations in which *many* turtles interacted to produce emergent systems behaviors. In concurrent research in *low-tech* settings, they also designed *StarPeople* activities, which allowed groups of *people* to "play turtle" in embodied simulations of complex systems (Resnick & Wilensky, 1993; Resnick & Wilensky, 1998; Wilensky & Resnick, 1995). In such participatory simulations (PartSims) a group of participants *enacts* emergent phenomena through coordinated role-play. Learners individually participate by playing the role of the system's agents that they are representing, and the group as a whole collectively experiences aggregate and emergent behaviors arising out of individual interactions.

Given the complexity and inherent parallelism of both StarLogo models and StarPeople PartSims, a design challenge arose about how to make agent-based insights into complex systems available to a wide audience, using existing technological infrastructure. (Initially, running StarLogo required some of the most advanced computers of the time.) Following one direction, which foregrounded agent-based modeling, Wilensky created NetLogo (Wilensky, 1999). NetLogo enabled both researchers and learners to create and run multi-agent complex systems models using the personal computers that were increasingly available in homes and at schools. Participatory simulations (PartSims) were then added to NetLogo by way of the HubNet module (Wilensky & Stroup, 1999a), enabling a distributed network of computers to run PartSims and making them easy to author in the NetLogo language. HubNet based PartSim research (Wilensky & Stroup, 1999b; 2000) observed significant learning outcomes and new forms and levels of participation, especially with underrepresented groups.

In parallel, a second line of research focused on carrying PartSims work forward utilizing wearable, customizable badges (Colella et al., 1998). This work identified the embodied learning benefits of wearables. For instance, in PartSims simulating the spread of a disease through a population using badges, high school students drew heavily on prior experiences, knowledge, attitudes, and interests (Colella, 2000). In spite of these successes, badge-based embodied PartSims were largely set aside in the 2000s, though some studies with Palm Pilots and other PDAs continued (e.g., Klopfer, Yoon, & Perry, 2005).

Programming and Robotics

While badge-based research focused on using portable and potentially ubiquitous computing devices for embodied PartSims, there was little attempt to involve participants in programming or designing hardware in that work. In contrast, other lines of constructionist research focused specifically on the affordances of small, open computing devices that were similar to badges in architecture and computational power as platforms for learners to build constructions involving both hardware and software. In particular, work in physical computing and robotics advanced the programmability and physical hackability of microcontroller platforms. The Cricket was created in this line of research, intended to allow learners to construct personally meaningful computational artifacts (Martin et al., 2000; Resnick et al., 2000). Such tools involved learners in authentic scientific practices and tapped into prior experiences, interests, and motivations (Martin et al., 2000). This approach remains a successful pathway into computation and scientific practice, as evidenced in the continued popularity of platforms like GoGo Boards, Arduinos, pcDuinos, and Lego Mindstorms. We argue that advances in wearable technology can support an approach that unifies this work with work on embodied social simulation, supporting not only PartSims but also scientific exploration, physical computing, and programming education. In this way, badges can serve as flexible, multifaceted tools for authentic explorations of scientific and computing practices that also foreground the social nature of computing.

Introducing the CCL-Parallax Programmable Badges

The CCL-Parallax badge was co-designed by Parallax, Inc. and the Center for Connected learning and Computer-based Modeling (CCL) at Northwestern University to enable open-ended group activities in an array of settings, from education-oriented conferences to classrooms. The badge comes equipped with an accelerometer, a small OLED display, and IR communication capabilities, meaning it is immediately possible for badges to communicate with each other. It is based on the Parallax Propeller 8-core microcontroller, which is 100% open source (hardware, firmware, and software). Additionally, multiple Integrated Development Environments (IDEs) for the badges include the Simple IDE (SIDE) and the Propeller IDE; and programming language options include the Propeller-C and Spin languages. The badges are also hardware-expandable through an array of solder pads. They are thus designed to be “hackable” at both software and hardware levels, inviting learners to tinker with, and customize, them. Across conference and educational settings, the goal is to engage wearers in computational thinking by offering a rich set of “onboard” functionality with an emphasis on social interactions through badge-to-badge communication.

All of these features support a “high ceiling” for the badges. That is, they ensure that learners will be able to pursue their interests without running into limitations of the technology. However, for the goal of broadening participation in computing, the badges are also intended to have a “low threshold” and be accessible to learners with little or no prior programming experience. For this reason, we are working to develop a blocks-based interface for creating badge programs. This will allow the badges to be programmed with elementary logic that includes access to all onboard functionality including badge-to-badge interactions and communications.

We aim to leverage the inherently social nature of the badges to engage young learners with computing and science practices in activities that position them as creators of personally meaningful constructions. Research has shown how perceptions of Computer Science and STEM fields as asocial and isolated negatively affect young learners’ dispositions to pursue these fields (American Association of University Women, 1994; Barton & Tan, 2010; Brickhouse et al., 2000;

Margolis & Fisher, 2003). Our end goal in providing a “low threshold” entry point is to enable the badges to respond to the interests and expertise of learners as they develop, and to create a novel pathway into STEM and computing learning experiences.

Designing for Constructionist Learning

In constructionist designs, the physical, social, and conceptual dimensions of learning all play important roles in structuring the experience of participants. While the constructionist tradition does not prescribe specific requirements for effective activities, it does lay out a series of guidelines and principles that collectively foster deep and effective learning. We argue that the CCL-Parallax Programmable Badge can support researchers in exploring a new constructionist design space of activities that can engage learners with powerful ideas in innovative ways.

Learning by Making

One of the central tenets of constructionism is that the development of internal knowledge structures can be facilitated through the construction of external artifacts. In *Mindstorms*, Papert uses the Logo programming language as one example of external construction in support of meaning making (Papert, 1980); and similar arguments have been made about construction with physical devices (Eisenberg, 2003). Moreover, improvements in digital fabrication and increased access to these technologies have introduced a host of tangible computing environments that enable new blends between computing and crafting (Blikstein, 2013).

To serve as an effective tool for supporting and investigating constructionist learning in a given domain, a technology must make appropriate tradeoffs between functionality, simplicity, cost, and other factors, as judged by the needs of designs in that domain (cf, Sipitakiat, Blikstein, & Cavallo. 2002; 2004). While the CCL-Parallax badge has been designed to be “protean” in the sense that they invite expansion, their onboard functionality nevertheless determines the type of activities that are their “native ground.” This built-in set of features clearly favors social and badge-to-badge interactions. Beyond the focus on IR communications, audio-visual outputs are central hardware components on the badges—from RGB and monochrome LEDs, to the high-density OLED display, to the headphone-compatible audio (and video) output port. Built-in libraries offer simple access to this hardware and facilitate uses that favor embodied social interaction. For example, there is a simple single command to vertically flip the OLED display, enabling switches for text to be read by the wearer (flipped), or by another person (public).

Social Syntonicity

Papert described the kind of learning he hoped to foster with Logo as *syntonic*. For instance, the turtle is *body syntonic* in that children interact with it in ways that are grounded in their own sense of themselves as physical beings navigating the world. It is *ego syntonic* in that their relation with the turtle is coherent with their ideas of themselves and others as beings with intentions, goals, and desires, In developing the badges, we aim to create distributed technology that supports *socially syntonic* learning. That is, we seek to develop a wearable platform for activities that encourage learners to invoke their ways-of-being as social creatures. Our goal is for badge activities to encourage learners to leverage their understandings of how, when, and why they interact with others. We see this as a particularly compelling and novel aspect of the badges, since it brings engagement with computing into the social realm, augmenting existing social practices of learners and broadening the range of contexts in which constructionist learning can occur. If successful, the badges could join other constructionist tools in serving as useful *objects-to-think-with*. Specifically, the badges can support thinking in settings that foreground distributed or group-level behaviors and phenomena. Given the importance of such phenomena to both computer science and STEM disciplines, the badges could be a significant addition to the constructionist design toolkit.

Public and Sharable Artifacts

Early Logo research found that the social and communicative dimensions of the construction process were integral to learning. Harel and Papert (1990) reported that Logo "facilitated the ongoing *personal engagement* and gradual evolution of different kinds of knowledge; and at the same time, it also facilitated the *sharing* of that knowledge with other members of the community, which in turn encouraged the learners to continue and build upon their own and other people's ideas" (p. 33). The integration of public sharing mechanisms has also played an important role in the success of some of today's most widely-used constructionist software (Fields et al., 2014). In making constructed artifacts public, the shared artifacts come to serve as public reflections of the creator's ideas. Furthermore, these artifacts can serve as occasions for conversations - conversations in which the learner is the expert, with valuable knowledge to share. Such discussions can serve as powerful opportunities to disseminate ideas, to gain and give feedback, and to reflect collaboratively with peers. Some of our early activity designs with the badges integrate new opportunities for public sharing, which are enabled by the underlying distributed computing model. In activities that engage learners in creating badge constructions that *communicate*, the iterative programming and debugging cycle also intrinsically involves sharing. Students need to work in pairs or groups even to test their programs; and these social interactions provide opportunities for sharing at all levels.

Powerful Ideas of Computing Made Accessible

Papert argues that an idea is *powerful* if it is seen to be immediately useful to the learner, if it connects to many other productive ideas, and if it is rooted in the learner's intuitive understanding about the world (1980, 2000). Constructionist research in computational thinking attends to the significance of various powerful ideas, be they *practices*—such as debugging and abstraction—or *concepts* and *patterns* that provide explanatory purchase in a variety of domains—such as emergence and feedback. Badges have the potential to expand access to powerful ideas in computing because they can operate not only individually as mobile computers, but also collectively as a distributed computing system

Activity Structures Supported by the Badges

In this section we briefly describe and categorize some of the activity types that we have explored with the badges. To date, our designs have fallen into three categories.

- Embodied PartSims of social and scientific phenomena
- Simulations of systems significant to computer science
- Social and distributed *maker* activities, emphasizing programming, wearable physical computing, or hardware hacking and expansion

In the first category, we see the badges as a means to carry forward the tradition of embodied PartSims research previously described. As an example, we have done field studies of two versions of a PartSim involving networks and disease transmission with three groups of learners, experimenting with the interactions between the badges, variations in the participant group, and different badge-mediated interaction designs. Two of these studies are described in Brady et al. (2015). Our fieldwork has included teachers in a professional development event; a group of 32 fourth graders in a museum setting; and a smaller group of 11 high school students, also in a museum. The disease activity has two phases. In the first phase, participants interact with each other as well as with badges that are affixed to inanimate objects in the "scenery" of the simulation environment. In the second phase, interaction data stored on the badges is relayed to a central computer, which provides a dynamic, manipulable visualization of these interactions. The group investigates the visualization, to gain an analytic perspective on the shared embodied simulation experience. In this category of work, the badges act as supports for the roles that participants take on in the simulation. Thus, these activities extend the metaphor of "badge as nametag" discussed previously, to a more general role that we call "badge as *costume*."

In a second category of activities, we have designed PartSims that address specific topics and phenomena in computer science. These activities differ from activities of the first category because the badges are foregrounded as computational devices, which are simulating the behavior of components in (other) computational systems. An example is the Wearing the Web activity (Brady et al., in review) in which learners play the role of network endpoints, data packets and routers in simulating a working instant message application. While activities in this category are also PartSims, the computational nature of participants' roles focuses their attention on the mechanisms by which the badges achieve their functions. These activities thus act as a bridge between research in the PartSim tradition and research that is oriented toward supporting learners in programming and physical computing.

In the third category of activities, the badges' computational nature takes center stage. Their behaviors become the explicit focus of collective attention, and the group collaboration is centered on constructing these behaviors. We provide an early instance of a software-focused multimedia activity in this category in the next section: Talking Badges.

Talking Badges

This is an introductory activity that can be used early in students' work with the badges and perhaps as their first experience with programming them. It foregrounds the audio-output functionality of the badges, along with a text-to-speech library that converts specially-formatted phoneme strings into intonated speech. For instance, the following string produces the first seven letters of the familiar alphabet song: "#1aybee+7seedee++ee ef-jee".

The Talking Badges activity provides an open social computing environment in which learners collectively develop computational artifacts of two kinds. First, at the level of *content* they construct and iteratively improve phoneme strings that yield recognizable utterances when run through the text-to-speech engine. Second, at the level of *code* the learners create programs that allow them to render and share the phoneme strings they are building. The "code" goal is pursued as a means of achieving the "content" goal, and both proceed in a distributed, social manner, leveraging the emerging collective knowledge and thinking of the group.

At the start of the activity, students are introduced to the blocks-based programming environment *and* to the phonemic conventions of the text-to-speech engine. They do this by examining a rudimentary initial program that enables them to (a) play pre-defined phoneme strings, (b) send these strings to another badge, and (c) store strings received from other badges into long-term memory, which can be uploaded to a computer when the badge is connected. After a simple demonstration of how to load and run programs onto the badges, how the initial program works, and how to upload strings collected in long-term memory, learners are invited to tinker with the program, changing it in any way they like and using their constructions to make their badges communicate. They are also encouraged to make use of an online class "Gallery" which allows them to post and download digital artifacts (e.g., programs or text strings), enabling sharing, remixing, and so forth.

It is important to note in this activity, while the user is programming, they are not writing programs from scratch. Instead they are modifying and extending existing programs, thus making the activities accessible to those without prior programming experience. And although the blocks-based environment is still being developed, early tests of the Talking Badges with graduate students suggest that text-to-speech is engaging to a variety of learners with a range of prior programming experience. Moreover, "coding" a phoneme string seems to provide a motivating entry point for tinkering with computer code.

Hybrid Activities that Blend these Categories

Because the badge platform is still in its infancy, our initial studies have involved PartSims in the first two categories of activity described above: these activities do not require a fully-developed low-threshold programming environment. However, we feel that a great deal of the innovation potential for the platform resides in the third activity category and also in an entirely new family of activities that *blend* PartSims and Making. For example, consider a PartSim in which participants can contribute to the simulation by modifying the code or hardware of their badges. Extending our existing work on disease spread, we have imagined a family of such activities in which a “viruses” or other pseudo-biological entities are represented a character strings that can be transmitted from badge to badge. Error-prone IR communications can simulate mutations in transmission. Students can be given challenges to create viruses with given properties, to generate “drugs” or simulate immune responses that attack “genetic” sequences, and so forth.

Conclusion

The CCL-Parallax Programmable Badge is designed to make powerful ideas accessible to learners through a variety of group-centered activity types that unfold in a social space and integrate virtual and physical constructions. We believe that badge-based activity designs can bridge between the PartSims and Programming themes within the constructionist research literature, as well as supporting broader scientific exploration and physical computing. This not only allows for activity sequences that incorporate both traditions, but it also opens up a new design space of hybrid activities—including PartSims involving distributed programming and networking scenarios. As we develop a robust learning platform around the badges and iterate the hardware design, we hope to share insights into new social dimensions of learning in computer science and the STEM disciplines.

References

American Association of University Women. (1994). Shortchanging Girls, Shortchanging America. Washington, DC: AAUW Educational Foundation.

Barton, A. C., & Tan, E. (2010). We be burnin’! Agency, identity, and science learning. The Journal of the Learning Sciences, 19(2), 187–229.

Blikstein, P. (2013). Digital fabrication and ‘making’in education: The democratization of invention. *FabLabs: Of machines, makers and inventors*, 1-21.

Borovoy, R. et al. (1996). Things that blink: Computationally augmented name tags. *IBM Systems Journal*. 35 (*3.4*), 88-95.

Borovoy, R. et al. (1998). Meme tags and community mirrors: moving from conferences to collaboration. *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, 159–168

Brady, C., Weintrop, D., Anton, G., Gracey, K., & Wilensky, U. (2015) Learning at the Intersection of Personal Expression, Social Computing, and Wearable Design with Programmable Badges. *Proceedings of SIGITE / RIIT conference.*

Brady, C., Weintrop, D., Anton, G., Orton, K., Rodriguez, S., & Wilensky, U. (In review) All Roads Lead to Computing: Making, Participatory Simulations, and Social Computing as Pathways to Computer Science. *IEEE Transactions on Education*

Brickhouse, N. W., Lowery, P., & Schultz, K. (2000). What kind of a girl does science? The construction of school science identities. *Journal of Research in Science Teaching*, 37(5), 441–458.

Colella, V. (2000). Participatory simulations: Building collaborative understanding through immersive dynamic modeling. *The Journal of the Learning Sciences*. *9*(4), 71-99.

Colella, V. et al. (1998). Participatory simulations: using computational objects to learn about dynamic systems. *Proceedings of SIGCHI 98,* New York: NY, 9–10

Conway, J. (1970). The game of life. *Scientific American*, 223(4), 4.

Eisenberg, M. (2003). Mindstuff Educational Technology Beyond the Computer. *Convergence: International Journal of Research into New Media Technologies*. *9*(2), 29-53.

Fields, D., Giang, M., & Kafai, Y. (2014). Programming in the wild: trends in youth computational participation in the online scratch community. In Proceedings of the 9th Workshop in Primary and Secondary Computing Education (pp. 2–11). ACM Press.

Papert, S., & Harel, I. (1991). Situating constructionism. In I. Harel & S. Papert (Eds.) *Constructionism*, pp. 1-11.

Klopfer, E., Yoon, S. & Perry, J. (2005). Using palm technology in participatory simulations of complex systems: A new take on ubiquitous and accessible mobile computing. *Journal of Science Education and Technology*. *14*(3), 285-297.

Margolis, J., & Fisher, A. (2003). *Unlocking the clubhouse: Women in computing*. MIT Press.

Martin, F. et al. (2000). To Mindstorms and Beyond. *Robots for kids: exploring new technologies for learning.*

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic books

Papert, S. (2000). What's the big idea? Toward a pedagogy of idea power. *IBM Systems Journal*, *39*(3.4), 720-729.

Resnick, M. et al. (2000). Beyond black boxes: Bringing transparency and aesthetics back to scientific investigation. *The Journal of the Learning Sciences*. *9*(1), 7-30.

Resnick, M. & Wilensky, U. (1993). Beyond the deterministic, centralized mindsets: New thinking for new sciences. Paper presented at the AEAR Annual Meeting*, Atlanta.*

Resnick, M. & Wilensky, U. *(*1998*).* Diving Into Complexity*:* Developing Probabilistic Decentralized Thinking Through Role-Playing Activities*. Journal of the Learning Sciences. 7(*2*)*, 153-172*.*

Sipitakiat, A., Blikstein, P., & Cavallo, D. (2002). The GoGo Board: Moving towards highly available computational tools in learning environments. In *Proceedings of Interactive Computer Aided Learning International Workshop*.

Sipitakiat, A., Blikstein, P., & Cavallo, D. P. (2004, June). GoGo board: augmenting programmable bricks for economically challenged audiences. In *Proceedings of the 6th international conference on Learning sciences* (pp. 481-488). ISLS.

Want, R. et al. (1992). The active badge location system. *ACM Transactions on Information Systems*. *10*(1), 91-102.

Want, R. & Hopper, A. (1992). Active badges and personal interactive computing objects. *IEEE Transactions on Consumer Electronics*. *38*(1), 10-20.

Wilensky, U. (1999). *NetLogo*. http://ccl.northwestern.edu/netlogo Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL

Wilensky, U. & Resnick, M. (1995). New thinking for new sciences: Constructionist approaches for exploring complexity. Paper presented at the AERA Annual Meeting, San Francisco.

Wilensky, U. & Stroup, W. (1999a). HubNet. http://ccl.northwestern.edu/netlogo/hubnet.html. Center for Connected Learning and Computer-Based Modeling, Northwestern University.

Wilensky, U. & Stroup, W. (1999b). Learning through participatory simulations: network-based design for systems learning in classrooms. Proceedings of Computer Supported Collaborative Learning (CSCL'99). Stanford, CA.

Wilensky, U. & Stroup, W. (2000). Networked gridlock: Students enacting complex dynamic phenomena with the HubNet architecture. In B. Fishman & S. O'Connor-Divelbiss (Eds.), Proceedings of the Fourth Annual International Conference for the Learning Sciences (pp. 282-289). Mahwah, NJ: Erlbaum.

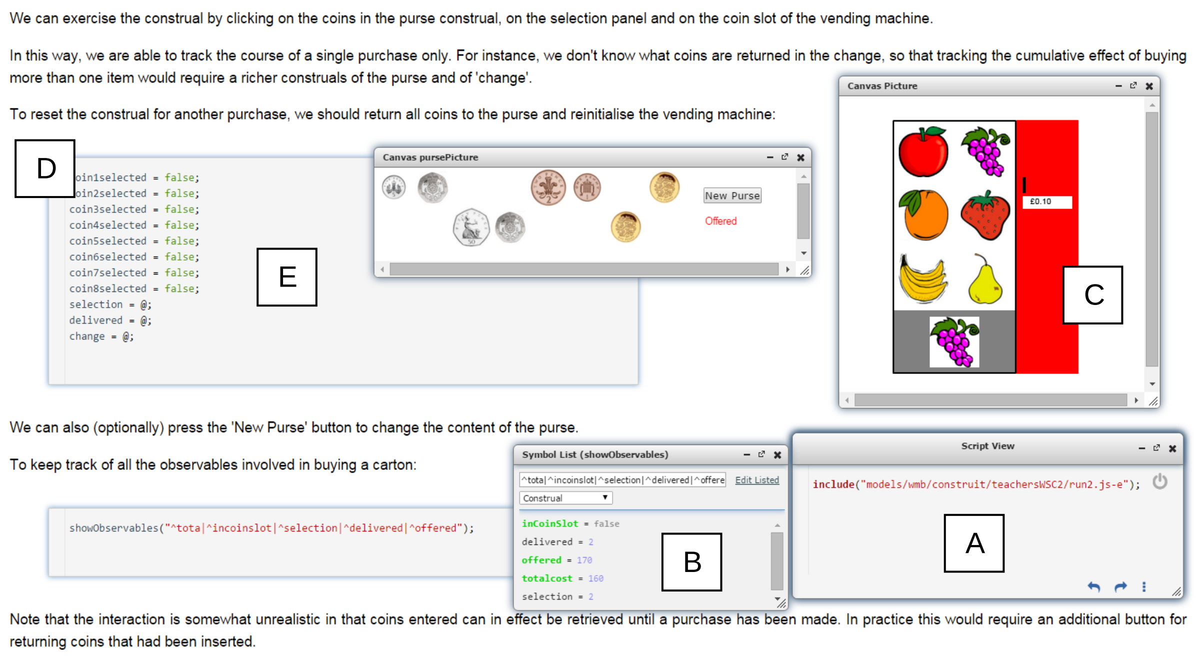
Construing and Computing: Learning through Exploring and Exploiting Agency

Meurig Beynon, *wmb@dcs.warwick.ac.uk*
Dept of Computer Science, University of Warwick, Coventry CV4 7AL

Jonathan Foss, Antony Harfield, Elizabeth Hudnott, Nick Pope
Dept of Computer Science, University of Warwick, Coventry CV4 7AL

Abstract

Constructionism is a practice that has developed alongside computing. In addressing a conference of educators in the 1980s, Papert himself recognised that “the new technologies are very, very rich in providing new things for children to do”, that a child who used LOGO to make a picture would be unlikely to say “I’m programming a computer”, and that “nobody knows how computers will be used in 10 or 20 or 30 year's time” (Papert, c1980). Thirty years later, despite many generations of development in LOGO, and the advent of other programming languages such as Alice and Scratch designed with constructionist aspirations in mind, there is scant conceptual support for computing that is not in essence based on a paradigm of ‘programming the computer’. In this paper, we outline an alternative constructionist practice that is based on exploiting computing technology in a way that cannot be accounted for merely in programming terms. A crucial ingredient in this new practice (“making construals”) is an epistemological stance that is constructivist in character and derives from the radical empiricism of William James. Adopting this stance enables us to regard the contention by Ben-Ari – fundamental to the idea of *programming* – that the computer is ‘an accessible ontological reality’, as itself a construction.


We can exercise the construal by clicking on the coins in the purse construal, on the selection panel and on the coin slot of the vending machine.

In this way, we are able to track the course of a single purchase only. For instance, we don't know what coins are returned in the change, so that tracking the cumulative effect of buying more than one item would require a richer construals of the purse and of 'change'.

To reset the construal for another purchase, we should return all coins to the purse and reinitialise the vending machine:

```
oin1selected = false;
oin2selected = false;
coin3selected = false;
coin4selected = false;
coin5selected = false;
coin6selected = false;
coin7selected = false;
coin8selected = false;
selection = @;
delivered = @;
change = @;
```

We can also (optionally) press the 'New Purse' button to change the content of the purse.

To keep track of all the observables involved in buying a carton:

```
showObservables("^tota|^incoinslot|^selection|^delivered|^offered");
```

```
^tota|^incoinslot|^selection|^delivered|^offere
inCoinSlot = false
delivered = 2
offered = 170
totalcost = 160
selection = 2
```

```
include("models/wmb/construit/teachersWSC2/run2.js-e");
```

Note that the interaction is somewhat unrealistic in that coins entered can in effect be retrieved until a purchase has been made. In practice this would require an additional button for returning coins that had been inserted.

Figure 1. Making a construal of a purse and vending machine

Keywords

constructionism; programming; making construals; radical empiricism

Introduction

Computing and construing are two concepts linked by the question: *What agencies are at work in the world?*. When computing, we exploit contexts in which we can be confident of the answer. When making a construal, we are exploring the myriad possible answers. Computing is typically concerned with how we can deploy the agencies at work in the world to achieve a clearly defined goal. Construing is typically an open-ended activity, speculative in nature, that is often invoked when we are uncertain or confused.

By tradition, computing is squarely rooted in an objective reality. As Ben-Ari explains (Ben-Ari, 2001), the novice programmer must come to terms with the computer as an ‘accessible ontological reality’, where there is no room for negotiation about the interpretation of an instruction. With reference to Latour’s essay on the nature of ‘construction’ (Latour, 2006), the established view of computing is most readily allied with a world view that espouses *naturalism*.

By contrast, construing is a personal and subjective matter. In some contexts, as in writing science fiction, for instance, it is quite appropriate for a construal to be fantastical. With reference to Latour (Latour, 2006), associating ‘making construals’ with ‘construction’ is potentially problematic because a construal can be so idiosyncratic and arbitrary. Where there is controversy, it is not uncommon for one person to say of another: “How can they possibly construe things that way?”. In this respect, construing has affinities with *deconstruction*.

Naturalism and deconstruction represent polarised views about the fundamental nature of agency. Latour argues that in order to sustain a robust notion of ‘construction’ it is necessary to resist both – yet to reconcile computing and construing within a constructivist setting is to bring together activities with the flavour of both (Beynon and Harfield, 2007). The almost paradoxical nature of this conjunction is reflected in the guarantees Latour sets out in his characterisation of authentic *construction*. Consider, for instance, his **First guarantee**: “once there, and no matter how it came about, discussion about X should stop for good. This is an essential assurance against endless controversies, heckling, superfluous doubts, excessive deconstruction.” and contrast this with his **Second guarantee:** “In spite of the indisputability insured by the former, a revision process should be maintained, an appeal of some sort, to make sure that new claimants—which the former established order had not been able to take into account—will be able to have their voices heard—and ‘voice’, of course, is not limited to humans.”

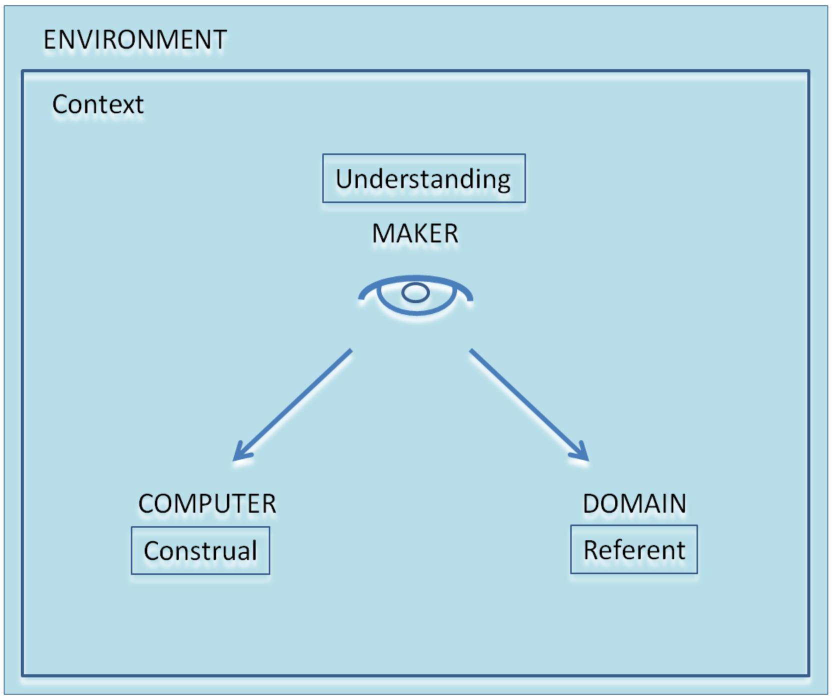


Figure 2: Making a construal by using a computer

In the approach to computing to be outlined and illustrated in this paper, the computer serves as an instrument to support the maker of construals (cf. Figure 2). The key idea is that the maker creates an interactive artefact that embodies their personal, provisional and evolving answer to the question 'what agency is at work in the domain of interest?' and auxiliary questions such as 'how is this agency mediated?'. The term 'construal' reflects the role of this interactive artefact in helping the maker to *construe* agency in the environment viz. to think about how agency in the environment of interest 'works'. Through interacting with the construal and in the environment, the maker crafts a computer-based experience that mirrors what is observed in interacting with the environment. This observation is typically oriented towards a specific 'referent' in the environment, such as an object or phenomenon to be understood, and involves creating an appropriate context for interaction (as in experimental practices in science). The directness of the connection between the construal and the referent, as experienced by the maker, is aptly captured by the 'mirror' metaphor. A movement of an object in the world and the associated movement of its image in the mirror are experienced as if they were one. Note that, even in this context, there is an element of 'construction' – if the mirror were sufficiently far away, then there might be some discernible delay between the movement of the object and that of its image, but the maker may still construe the one movement to be indivisibly associated with the other.

From a learning perspective, the connection between a construal and its referent is of the same kind that a native speaker experiences between words and their meaning. No conscious element of formal translation is involved – the words are heard or spoken and what they signify comes to mind in the very act of hearing or speaking. Much experience of a language and of life is required to acquire this degree of fluency, and there is no point at which the understanding of the meaning of a word or phrase can be deemed absolute. It is quite appropriate that the term 'construing' is conventionally used to refer to what we do when faced with a situation in which the fluent connection between a phrase and its meaning is for some reason perturbed or undermined. Construal in this sense involves playing around with words and exploring what comes to mind so that the fluency of the connection can be restored. Though the computer is not involved in this kind of construal, essential ingredients of such activity are represented in Figure 2. As a computer-based example of a similar kind of 'connection in experience' to which Figure 2 applies, consider the way in which, without conscious effort, an examiner connects the rows and columns of a spreadsheet with students and subjects of study.

The philosophical outlook that is best matched to 'making construals' is the *radical empiricism* of William James (James, 1912). James's central thesis is that *all* knowing is rooted in connections between experiences that are themselves given-in-experience ('conjunctive relations'). James contends that those aspects of experience that we regard as objective or formal are derived from primitive experiences of connection: "the "truth" of our mental operations must always be an intra-experiential affair" (James, 1912, p202), "... subjectivity and objectivity are affairs not of what an experience is aboriginally made of, but of its classification" (James, 1912, p141), "knowledge of sensible realities ... comes to life inside the tissue of experience" (James, 1912, p56). The importance of adopting a Jamesian outlook for construction is that it addresses the issue to which Latour draws attention: the need to resist the claims of naturalism and deconstruction. Where naturalism argues for absolute knowledge of agency, radical empiricism interprets all attribution of agency as pragmatic in nature, calling into question the significance and practical value of believing that "we could know what causation really and transcendentally is in itself" (James, 1912, p185). And where (as Latour maintains) deconstruction "has been able to turn into dust all the claims to solidity, autonomy, durability and necessity", radical empiricism traces the root of all such claims to connections in personal experience whose authenticity is pragmatically attested and cannot be refuted: what proof do we need that a native speaker makes fluent connections between words and their meanings but to hear them speak? This Jamesian perspective on naturalism and deconstruction is summarised in the Principle of Pure Experience: "Everything real must be experienceable somewhere, and every kind of thing experienced must somewhere be real" (James, 1912, p159-60).

This paper outlines and illustrates an approach for exploiting computer-based technology to make construals. This is the theme of an ongoing EU Erasmus+ project, entitled CONSTRUIT!, aimed at disseminating online resources for making construals. The three following sections respectively discuss: the basic principles of making 'digital' construals; how this gives support to making connections in experience of the kind described by William James; and what merits this has where computer support for constructionism is concerned. More details appear in a tutorial paper (Beynon et al, 2015) and in online versions of the construals shown in Figures 1, 3 and 4.

Making construals by using the computer

In making a construal by using the computer the focus is on exploiting the rich interactive experiences that computing technology affords. In keeping with James's view that all knowing is rooted in connections between one aspect of our experience and another that are themselves experienced, the goal is to develop ways of invoking and evoking connections in experience.

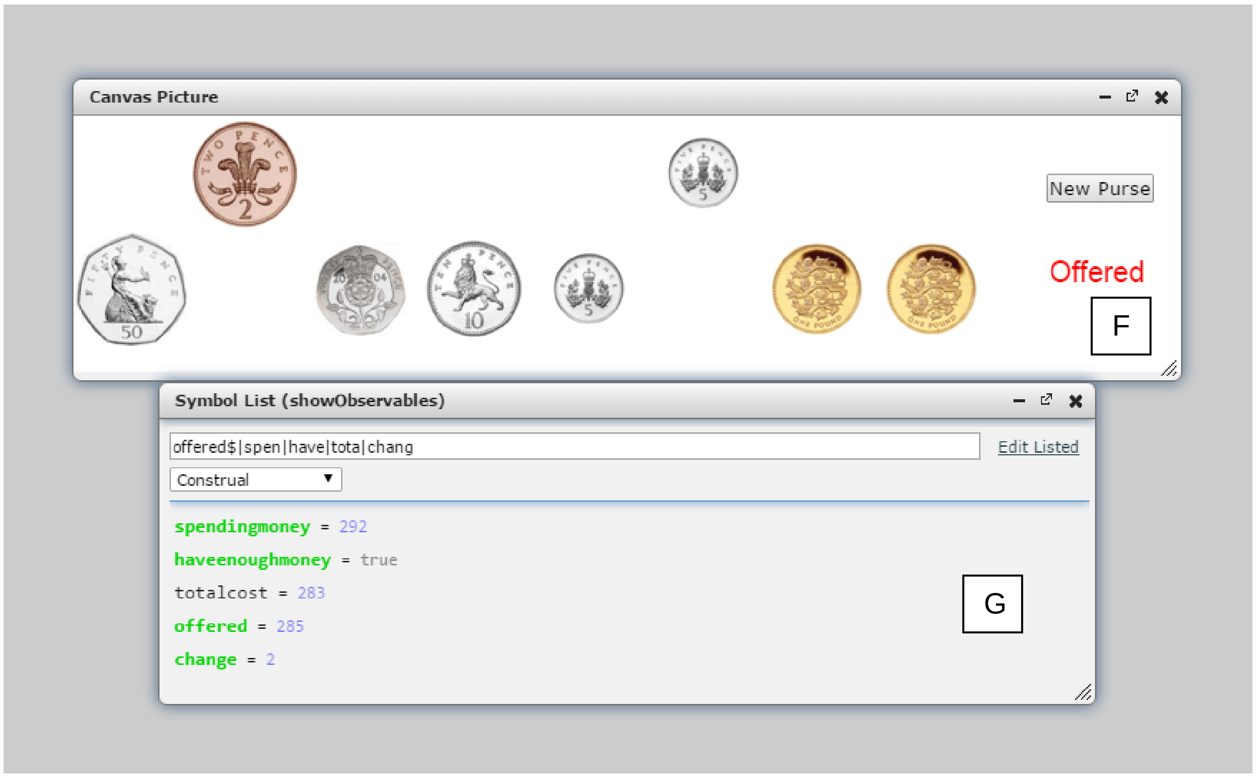
Figure 2 depicts the key ingredients in making a construal. The maker, represented by the eye icon, observes the construal on the computer (e.g. the symbols in the cells of the spreadsheet grid), in parallel observes an independent phenomenon (e.g. the exam performance of a class of students) and experiences a connection between what is observed on the computer and what is observed in the world (e.g. reading the value of a spreadsheet cell as 'Tim's mark in Greek'). The term 'referent' is used to refer to "what is observed in the world" (cf, Figure 2).

What makes it possible for the maker to experience such a connection? Though the experience is always in the present, it is informed – indeed constructed – by past and ongoing experience of interaction with the construal and its referent. Of central importance is the fact that the changes to the referent that might be expected to occur have direct counterparts in the construal (cf. revising Tim's mark in Greek, removing a row to signify that Tim has withdrawn from Greek, scaling the numbers in a column as the Greek exam was too hard). In Figure 2, the term 'understanding' refers to the memories of prior interactions and the expectations these arouse in the maker's mind in this fashion. If the construal has reached a mature state of development, the maker's experience of the connection between the construal and its referent will be very reliable and unremarkable. The crafting of the construal aims to establish and maintain a connection that can be experienced seamlessly. In this respect, the context for the interaction (cf. Figure 2) is significant as it influences what is to be expected (cf. the absence of a mark in a spreadsheet is normal whilst an exam is still being marked, but concerning at an exam board).

The connection in experience between a construal and its referent is enabled by a correspondence between 'observables' in the construal and in the referent. An observable is an entity to which a current value or status can be attributed. Observables in the construal may or may not have a visual representation – the term 'observable' is used in a broad sense, as in science, where an observable value may not be directly perceptible. Observables are linked by dependencies, whereby changing the value of one observable affects the value of others in a predictable way. As the spreadsheet illustrates, recognising observables in the construal and the referent and experiencing a connection between them relies upon perceiving dependencies.

The practice of making a construal will be introduced via a simple illustrative example. The maker interacts with a special-purpose *environment for making construals* ("the MCE"). Within the MCE, the maker can develop a network of observables and dependencies in an empirical incremental fashion. Each dependency is expressed by giving a definition for the value of an observable similar to the definition of a spreadsheet cell. The current state of the construal is then recorded as a set of definitions or 'script'. Within the MCE, the definitions in the script can be freely modified and inspected. By way of illustration, the screenshot in Figure 1 depicts a construal of a purse and vending machine: it features a window (A) through which the script can be modified, a Symbol List (B) from which current values and definitions of observables can be obtained, together with visual representations of observables where appropriate (C). Unlike a conventional program, a construal does not have a clearly prescribed 'user-interface' and 'user-manual'. It supports a wide range of open-ended interactions and interpretations, many of which are not preconceived. The maker's ‘understanding’ is one of many possible ways to interact with the construal. This understanding can be partially documented by embedding script fragments into a narrative, as in the backdrop to Figure 1, where what is involved in construing a purchase (D) and the definitions needed to initialise the purse and vending machine are displayed (E).

More insight into the role of the MCE can be gained from looking more closely at the purse construal that features in Figure 1. Panels extracted from a screenshot of the purse construal are shown in Figure 3.



```
offered$|spen|have|tota|chang

spendingmoney = 292
haveenoughmoney = true
totalcost = 283
offered = 285
change = 2
```

Figure 3. A construal of a purse

In making the purse construal, the scenario the maker has in mind as a referent is the old-fashioned shop, where the customer takes money out of their purse and places it on the counter when they wish to buy something. This is modelled in the construal by clicking on the coins so that they are moved out of the purse, where they are being offered in purchase (F).

To expose the construction that underlies this transaction, it is useful to contrast the customer’s adult view with that of their accompanying child. What the child sees is simply the placing of the coins as tokens on the shop counter. It is an activity devoid of explicit meaning. In fact, the activity embodies many invisible observables, dependencies and a pattern of agency that is only appreciated by the adult. The Symbol List (G) in Figure 3 shows how these are taken into account in the construal. The adult interprets the coins they have to hand as a derived observable – the money available to spend (“spendingmoney”). They have in mind the total cost of the items they wish to buy (“totalcost”). They can observe how much money they have offered (“offered”) and know how much change to expect (“change”).

The observables that are defined by dependency are shown in green. For instance, the observable ‘change’ has the definition:

```
change is offered - totalcost if (offered >= totalcost) else @;
```

The role of the construal is to mimic the environment for interaction and agency that is experienced in its referent. Dependency is crucial in making it possible to capture the many possible concurrent interpretations of a simple atomic action. For instance, in the purse construal, 'choosing to offer a coin in payment', to this end 'placing the coin on the counter' and 'increasing the amount of money being offered' are expressed as one and the same action

Modelling the current state of the referent as a network of observables and dependencies that can be revised at any moment in an open-ended fashion also makes it possible to take account of many different agent perspectives in ways that cannot be comprehensively preconceived. For instance, suitable supplementary definitions turn the coins over for the benefit of someone who is totally unfamiliar with the currency but can read the numerical values displayed on the other side of the coin. A simple transposition of definitions can reflect a misconception about the value of coins, such as mistaking 2p coins for 5p coins and vice versa on account of their relative size. Another form of agency is expressed by the 'New Purse' button, which randomly generates new contents for the purse, as is appropriate in a context where what the customer has in their purse is a matter of chance. Changing the definition of 'totalcost' to reflect the need to save the bus fare home illustrates an alternative way in which agency may be adapted to context.

Making construals as making connections in experience

The aspiration in making construals in the MCE is to give explicit concrete expression to the Jamesian notion that all knowing is rooted in making connections in our experience.

The development of the construal of a purse and vending machine depicted in Figure 1 was contrived to demonstrate how 'making connections in the experience' pervades every aspect of the maker's interaction with the MCE. Indeed, given sufficient expert knowledge of its referent, learning how to exploit the MCE as an instrument for making connections in experience is in principle all that is required to be able to develop a construal from scratch.

With reference to Figure 1, the skeletal visualisation of the vending machine comprises four rectangles: a compartment to hold the items for sale, a slot to put coins in, the delivery tray and a box for the change. The definition of the observable to represent each rectangle has the form:

```
testrectangle is Rectangle(10,10,100,50);
```

To construct the visualisation, the maker must first define the observables for the component rectangles and then establish the appropriate geometric connections between them. This construction can be carried out by repeating a standard process for 'making connections in experience' that is quite characteristic of the MCE. This process involves disposing appropriate panels, one or more of which is a script fragment that can be interactively edited, in such a way that they can be viewed simultaneously, then interacting in an exploratory fashion by editing the script and observing the result.

A simple application of this technique juxtaposes the above definition of 'testrectangle' with its visualisation on the canvas. Experimental redefinition of the four parameters on the right-hand side of the definition readily identifies them as referring to the x and y coordinates of the top left corner of the rectangle, the width and the height of the rectangle respectively. Note that being able to associate the symbolic names x, y, width, height is not a prerequisite for being able to manipulate the rectangles or indeed to elaborating the construal. The connection between the visualisation and the definition can be experienced whether or the learner can attach objective names to the parameters. But for the learner who speaks English, the connection is more vivid.

Though this empirical approach to such a simple task as constructing rectangles at first sight appears trivial, this is misleading. Making connections in experience is indeed a primitive learning activity but – in keeping with James's thesis about knowing – its cumulative potential to express rich and subtle understanding is immense. As illustrated by the allusion to English-speaking makers above, a connection in experience does not have the binary limitations of semantic relation based on logic. Neither should we forget that the abstract concept of a rectangle is far from trivial, when we consider that no true rectangle can be physically drawn.

For the non-specialist, the exploratory process outlined above may seem more appropriate when considering the more technically challenging task of specifying the fruit images in Figure 1, as in:

```
apple is HTMLImage(0.1*s, 0.1*s, 0.8*s, 0.8*s, imagelocation // "a.png");
```

The same process generalises to other connections between the rectangle and image constituents of the vending machine that relate to its real-world semantics (cf. Figure 1). These include the connections that give integrity to the parts by defining their location relative to a single reference point and those that keep them in the same proportion (cf. the scale factor 's' in the formula above). It is also used to set up dependencies to ensure: that, for the purpose of submitting coins to the vending machine, an observable 'inCoinSlot' attached to the coin slot responds to mouseover; that, when appropriate, the change due will be displayed in the change tray; and that the selected choice of fruit juice will be displayed in the delivery tray.

The benefit of making construals where making connections in experience is concerned is illustrated by the way in which (as shown in Figure 1) the construals of the purse and vending machine can be blended within the MCE to form a new construal. This makes it possible to apply the exploratory process described above to make connections between the state of the purse and that of the vending machine. For instance, when the money offered is sufficient to buy the item selected, the item is delivered.

The above discussion suggests that making the construal in Figure 1 is a matter of exploiting the support for exploratory interaction in the MCE to express the connections in experience that are in the mind of a person who has real-world experience of using a purse and a vending machine. This is not without its elements of technical challenge, but several distinctive qualities of the environment can assist in establishing these connections:

- the immediate / present moment nature of the model of current state
- support for direct interaction that can disclose latent patterns of state-change
- means to display aspects of state and dispose them so that they can be viewed side-by-side
- means to distinguish different kinds of agency and modes of state-change.

Computer support for constructionism

Construing has an important role in teaching and learning. The learner construes what the teacher presents. The teacher construes the responses of the learners. Being able to apprehend connections in experience has a crucial role in learning. Terry Pratchett, the distinguished author who suffered from a rare form of Alzheimer's disease at the end of his life, describes how his illness made it difficult for him to put his pants on the right way round (Pratchett, 2010). He had lost the ability to interpret what he was looking at to such an extent that he could not recognise the orientation of his pants without first pulling them up, and then – if necessary – taking them down, walking around them, and pulling them up again. This is evidence that our capacity to make connections is implicit even in what appear to be the simplest tasks. It also shows that being able to model connections in experience is a necessary element in giving personalised support for learning. What is more, it suggests that our reliance on being able to make connections is hard both to recognise and to analyse.

The need to understand learning with reference to modelling connections in experience that may take subtle and unexpected forms is a reason for thinking that making construals is a better foundation for constructionist learning than conventional programming. This may seem paradoxical when we consider that 'developing the MCE' could be regarded as 'writing a program'. The fundamental distinction is that the priorities in giving support for making construals are quite different from those involved in writing a computer program. The primary emphasis in exploiting computer-based technology as a rich source of experience is not on implementing an algorithm and crafting a user-interface to realise a specific functional objective. Of paramount importance are the characteristics of the peripheral devices that make state perceptible, pragmatic issues concerning the instrumental qualities of the construal as matched to the skills and expertise of the maker, and considerations concerning real-time response and how agency is mediated that are in the normal way regarded only as epiphenomena.

The basic procedural specification of a rectangle in LOGO is ill-suited to expressing meaningful relationships associated with rectangles such as were discussed above: there is a gross mismatch between a formal recipe to describe a behaviour in a highly engineered and precisely formulated machine-like environment and an informal and open-ended perception of state with plausible agency that is yet to explored. Alternative programming paradigms, whether declarative or object-oriented, resolve this problem only to the extent that more useful abstract relationships can be preconceived. The idea of making connections in an open-ended manner transcends the formal semantic boundaries that such abstractions impose.

Though educational software is not in general conceived with explicit computer programming concepts in mind, the influence of computational thinking is still apparent. A typical approach to developing a microworld involves conceiving a rich set of affordances for the learner to enable them to explore the states that arise. These affordances reflect what a teacher might consider to be appropriate primitive steps out of which interesting and instructive patterns of behaviour can be fabricated. Certainly, the concept of 'use' no longer applies to the learner's interaction in such environments: through exploration, the learner may encounter situations that were never in the teacher-developer's mind. The danger is that in such situations the connection in experience that can associate meaning with state has been lost, thereby removing the ground from the learner's efforts at interpretation. In contrast, making construals is an activity in which the experience of making connections is critical – the construal, its referent, the maker's understanding and the context for interpretation can evolve freely, but if the correspondence between the construal and its referent is no longer the subject of a connection in experience, the exercise is futile.

In the educational context, the significance of maintaining dependencies as a way of establishing a cognitive link between software and the domain to which it refers is well-recognised (Beynon, 2007; Roe and Beynon, 2007) . Dependency features prominently in dynamic geometry and in the application of spreadsheets in education (Baker and Sugden, 2003). The Scratch programming environment also has built-in features to enable the computing novice to make connections between program variables and media devices. Satisfactory support for construction demands more than adding dependency to the repertoire of features available in a programming language: the computing environment in which the maker-learner interacts cannot be abstracted from the learning domain if it is to be open to live specification of dependency.

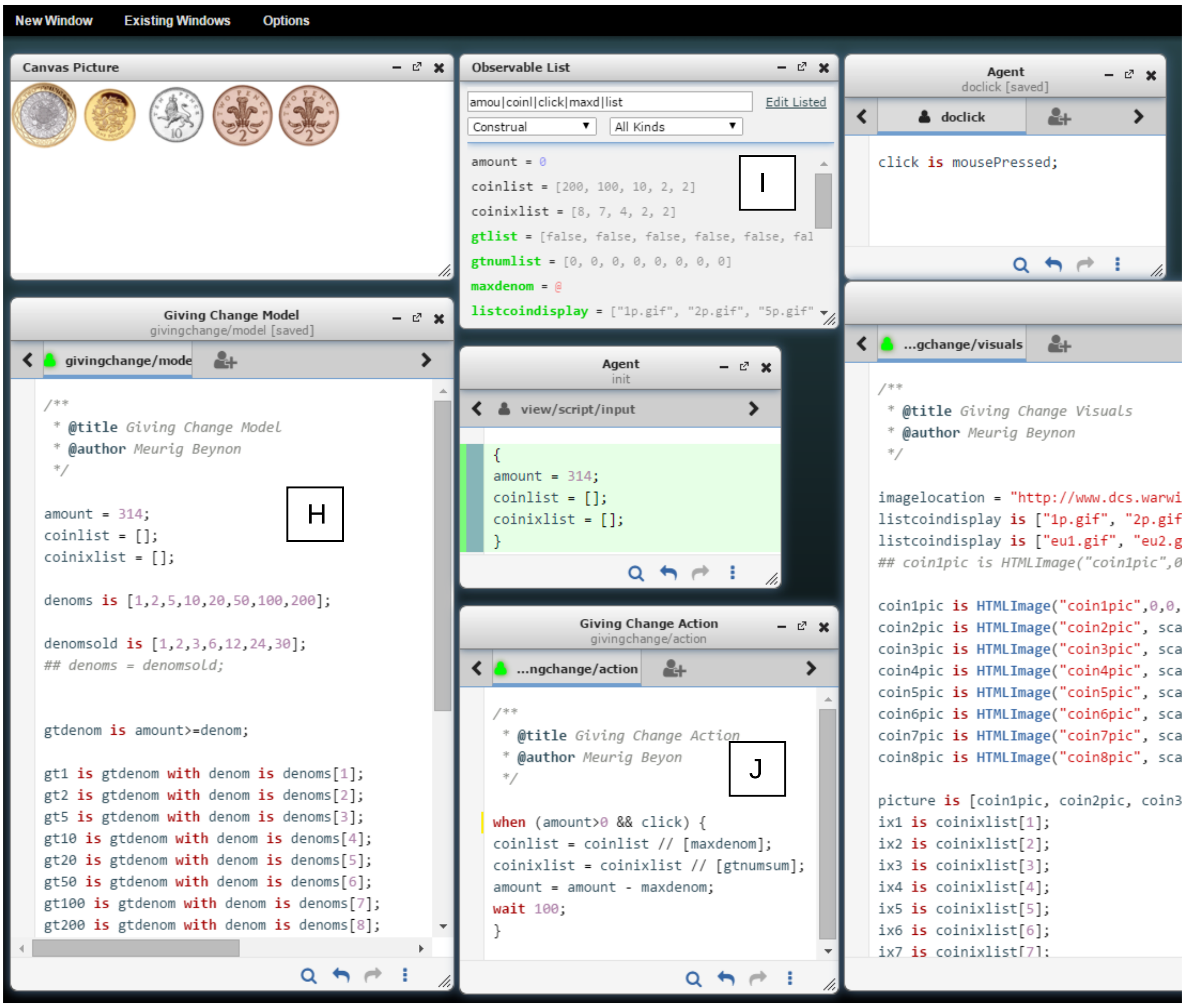
Making construals makes it possible to specify program-like activities without compromising the openness to live interaction and revision (cf. Beynon, 2009). This is illustrated by the construal of 'giving change' depicted in Figure 4 below. The learning resource on which this construal has been modelled is a Scratch program entitled 'Coins' from Phil Bagge's book on teaching primary programming (Bagge, 2015, p107).

As described by Bagge, the Coins program is conceived as 'a machine that chooses the largest coins possible to make from the pence inputted by the user'. This machine is constructed by making use of several abstract procedural constructs represented by Scratch blocks.

In making a construal, the primary focus of attention is on what sequence of states might be observed when one person gives change to another. Panel (H) in Figure 4 shows some key observables and dependencies viz. whether the amount of change to be given in pennies exceeds each of the denominations of coins available (gt1, gt2, gt5 etc.). These truth values can be disposed in a list that is ordered from the smallest to the largest denomination (see 'gtlist' in panel (I)) and reinterpreted as 1s and 0s via a simple dependencies (see 'gtnumlist' in (I)) . In the list of denominations ('denoms' in (H)), the index of the maximum denomination smaller than 'amount' is defined as the sum of the elements in 'gtnumlist'. This maximum denomination can be defined by dependency via this index (see 'maxdenom' in (I)).

The core action in specifying the change in coins (as recorded in 'coinlist' in panel (I)) appends a coin to represent 'maxdenom' to 'coinlist' and reduces 'amount' by 'maxdenom' (see panel J). When this action is performed, the values of the observables in panel (H) are updated by dependency, and the next coin to be given in the change is identified. In this approach, there is a direct correspondence between acting out the process of giving change and executing it automatically. This makes it possible to interleave manual and automatic action – so that for instance keeping back a 2 pence coin and in place using two penny coins can be acted out.

In the Coins program, the denominations are built into the structure of the code. In contrast, the construal of giving change illustrates the separation of agency (J) from the specification of context (H). This gives potential for easy and flexible adaptation of the context (e.g. a single redefinition can change the denominations available or convert from UK to Euro currency) and of the algorithm (e.g. giving change so that the smallest denominations are presented first). The construal can also be plugged into the construal of the purse and vending machine in Figure 1.



```
amount = 0
coinlist = [200, 100, 10, 2, 2]
coinixlist = [8, 7, 4, 2, 2]
gtlist = [false, false, false, false, false, fal
gtnumlist = [0, 0, 0, 0, 0, 0, 0, 0]
maxdenom = @
listcoindisplay = ["1p.gif", "2p.gif", "5p.gif"
```

```
click is mousePressed;
```

```
/**
 * @title Giving Change Model
 * @author Meurig Beynon
 */

amount = 314;
coinlist = [];
coinixlist = [];

denoms is [1,2,5,10,20,50,100,200];

denomsold is [1,2,3,6,12,24,30];
## denoms = denomsold;


gtdenom is amount>=denom;

gt1 is gtdenom with denom is denoms[1];
gt2 is gtdenom with denom is denoms[2];
gt5 is gtdenom with denom is denoms[3];
gt10 is gtdenom with denom is denoms[4];
gt20 is gtdenom with denom is denoms[5];
gt50 is gtdenom with denom is denoms[6];
gt100 is gtdenom with denom is denoms[7];
gt200 is gtdenom with denom is denoms[8];
```

```
{
amount = 314;
coinlist = [];
coinixlist = [];
}
```

```
/**
 * @title Giving Change Action
 * @author Meurig Beyon
 */

when (amount>0 && click) {
coinlist = coinlist // [maxdenom];
coinixlist = coinixlist // [gtnumsum];
amount = amount - maxdenom;
wait 100;
}
```

```
/**
 * @title Giving Change Visuals
 * @author Meurig Beynon
 */

imagelocation = "http://www.dcs.warwi
listcoindisplay is ["1p.gif", "2p.gif
listcoindisplay is ["eu1.gif", "eu2.g
## coin1pic is HTMLImage("coin1pic",0

coin1pic is HTMLImage("coin1pic",0,0,
coin2pic is HTMLImage("coin2pic", sca
coin3pic is HTMLImage("coin3pic", sca
coin4pic is HTMLImage("coin4pic", sca
coin5pic is HTMLImage("coin5pic", sca
coin6pic is HTMLImage("coin6pic", sca
coin7pic is HTMLImage("coin7pic", sca
coin8pic is HTMLImage("coin8pic", sca

picture is [coin1pic, coin2pic, coin3
ix1 is coinixlist[1];
ix2 is coinixlist[2];
ix3 is coinixlist[3];
ix4 is coinixlist[4];
ix5 is coinixlist[5];
ix6 is coinixlist[6];
ix7 is coinixlist[7];
```

Figure 4. A construal of giving change

Concluding remarks

The aspiration in CONSTRUIT! is to develop open online resources to teach making construals, and so encourage its adoption in school education. Support for 'digital' construction has potential benefits not only for developing general educational resources but also for computing and computer science education. The synergy between James's radical empiricism and making construals may also help in communicating both. As James (1912, p90) remarks of his 'philosophy of pure experience', so crucial to interpreting the notion of construction being advocated in this paper: "it is almost as difficult to state as it is to think it out clearly".

The interested reader can find more background on both the practical and conceptual aspects of this paper. The illustrative examples were developed in association with training activities held under the auspices of the CONSTRUIT! project, and can be accessed via the link to the MCE cited below (The CONSTRUIT! environment, 2015). For in-depth discussions of the role of dependency in educational technology, and of how the concept of construal can be related to model-building in support of software construction, see (Beynon, 2007) and (Beynon 2012).

Acknowledgements

We are grateful to Mike Joy for acting as the Coordinator for CONSTRUIT! and to other project team members for their stimulating input. This project has been funded with support from the European Commission under the Erasmus+ programme (2014-1-UK01-KA200-001818). This publication reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

References

Bagge, P. (2015) *How to Teach Primary Programming Using Scratch*, University of Buckingham Press. Online at http://code-it.co.uk/wp-content/uploads/2015/05/coins_planning.pdf

Baker, J. E., and Sugden, S. J. (2003) *Spreadsheets in Education - the First 25 Years*. Spreadsheets in Education e-journal, 1 (1), 18-43.

Ben Ari, M. (2001) *Constructivism in Computer Science Education*. Journal of Computers in Mathematics and Science Teaching 20(1), 45-73.

Beynon, M. (2007) *Computing technology for learning - in need of a radical new conception*. In Journal of Educational Technology & Society, 10 (1), 94-106.

Beynon, M. (2009) *Constructivist Computer Science Education Reconstructed*. HEA-ICS ITALICS e-Journal, Volume 8 Issue 2, June 2009, 73-90.

Beynon, M. and Harfield, A. (2007) *Lifelong Learning, Empirical Modelling and the Promises of Constructivism*. Journal of Computers, Volume 2, Issue 3, May 2007, 43-55.

Beynon, M. (2012) *Modelling with experience: construal and construction for software*. Chapter 9 in Ways of Thinking, Ways of Seeing (ed. Chris Bissell and Chris Dillon), Automation, Collaboration, & E-Services Series 1, Springer-Verlag, January 2012, 197-228

Beynon, M. et al (2015) *Making construals as a new digital skill: dissolving the program - and the programmer - interface*, Proc. International Conference on Interactive Technologies and Games, 22-23 October 2015, Nottingham, UK, 9-16.

James, W. (1912) *Essays in Radical Empiricism*, Longmans, Green and Co., New York.

Latour, B. (2006) *The promises of constructivism*, in D. Idhe (ed.) Chasing Technology: Matrix of Materiality, Indiana Series for the Philosophy of Science, Indiana University Press, 27-46.

Papert, S. (c1980) *Constructionism vs Instructionism* – a speech delivered to a conference of educators in Japan. Online at http://www.papert.org/articles/const_inst/const_inst1.html

Pratchett, T. (2010) *The Importance of Being Amazed about Absolutely Everything*. Inaugural Lecture, Trinity College Dublin. Online at https://www.youtube.com/watch?v=n2FZ_0d3yEI

Roe, C. and Beynon, M. (2007) *Dependency by definition in Imagine-d Logo: applications and implications*. In Ivan Kalaš (ed.) Proc. of the 11th European Logo Conference, 19-24 August 2007, Bratislava, Slovakia.

The CONSTRUIT! environment for making construals: “the MCE”, (2015) Online together with versions of the construals in Figures 1, 3 and 4 at http://jseden.dcs.warwick.ac.uk/construit.c6/ For instructions for accessing these, refer to go.warwick.ac.uk/em/publications/papers/132.

Costa Rica's Omar Dengo Foundation Program:29 years later

Leda Munoz, leda.munoz@fod.ac.cr
Executive Director, Omar Dengo Foundation

María Eugenia Bujanda, maria. bujanda@fod.ac.cr
Former Educational Director, National Program of Educational Informatics MEP-FOD

Abstract

Twenty-nine years ago, Costa Rica decided to introduce computers into the educational system. From the beginning, it was clear that a solid theoretical and pedagogical basis should guide the design of the project and that it ought to be focused on the use of computers as tools to develop high level thinking skills and on training the teachers in the methodology to be used rather than on the computer hardware and software. This understanding was a key factor to select the pedagogical proposal of Seymour Papert, whose influence in the design and initial implementation of the project was strategic.

With time, the project became a sustainable, innovative and dynamic National Program, implemented through a public-private partnership between the Ministry of Public Education and the Omar Dengo Foundation. This paper reviews the evolution of the Program, the results achieved, lessons learned, and challenges still ahead. The financial cost of this type of programs is substantial, particularly to developing countries that still have educational systems demanding resources to cover even the basic needs of infrastructure, educational materials and of sufficient and well-trained teachers. The international cooperation and financing agencies are interested in understanding under what circumstances such investments generate significant results, both educationally and socially, and allow for a sustainable scheme. As more and more countries around the world have engaged in the design and implementation of variants of these projects, it becomes more important to examine the results that have been obtained, and share the knowledge that accumulates around.

One of the key issues is understanding the role of teachers and of the curricula in this type of programs. The profound transformations that new technologies are generating in all areas of society, including the knowledge about our brain and its specific learning processes, call for a deep revision of our educational systems. The twenty-first century demands creative and collaborative citizens capable of contributing to solve local and global problems and provide new and inspiring ideas. How to create the adequate conditions to foster these conditions is a challenge, and our experience suggests that Constructionism does provide a valuable alternative.

The initiation of an idea

- An idea surges: to bring computers to students at public schools
- The educational model: PBL and programming using Logo as the core language and learning tool
- The organizational model: a Foundation
- Project´s launch in 1988: first steps in implementation, 57 elementary schools, two lessons a weak, a tutor in each computer Lab
- The governance model: Board of Directors with the Minister of Public Education as Board´s President

In 1987, a group of academics and entrepreneurs were called by the recently elected President to shape a project to provide computers to schools in Costa Rica. This was quite a visionary and courageous idea if we consider the fact that computers were at that time a luxury item that was difficult to see even in the most advanced companies in the country and when initiatives to make computers available at schools were rare, especially in developing countries.

This early decision to invest in digital technologies and enrich children's learning opportunities has to be pondered against the background of the country's historical determination of making education a pillar of its development strategy since the declaration in the country´s 1847 Constitution –well ahead to more developed countries in the region – that education ought to be universal, free and state-financed.

The founding group started by asking some fundamental questions: Computers in school, for what? How to do it? 29 years later the ideas that inspired their answers to those questions are still sound and valid, if not more. One of the founding group´s key insights was that the introduction of technologies in schools should lay the foundation to prepare for the country's transition into a modern economy by providing its citizens with the creativity, flexibility and intellectual skills needed for this shift. Technologies were also seen as a powerful tool for innovation and regeneration of the national educational system, at that time considered to be in a deep crisis. To these motivations, we must also add the concern to close the emerging technological gap, not only in its international manifestations, i.e., between developing countries and industrialized countries, but also in its national manifestations, i.e., between the different sectors of the country and among generations (Fonseca, 1991).

So, from its beginnings the initiative showed a broad understanding of what the purposes of technology in education could be, expressed in the following four domains (Fallas & Zúñiga, 2010):

- In the individual domain: to develop students' creativity and intelligence.
- In the educational domain: to make a contribution to improving the quality of the education system and foster its technological modernization.
- In the social domain: to strengthen social cohesion inside the country and so reduce existing socioeconomic, geographical, technological and educational gaps.
- In the economic domain: to help achieve a wider incorporation of people into the national economy and the international dynamics.

From the onset, it was clear that a solid theoretical and pedagogical basis should guide the design of the project and that it ought to be focused on the use of computers as tools to develop high-level thinking skills and on training teachers in the methodology to be used rather than on the computer hardware and software. This understanding was a key factor to select the pedagogical proposal of Seymour Papert, whose influence in the design and initial implementation of the project was strategic.

So, the decision was made to start introducing computers at the elementary school level --contrary to most initiatives of that time that worked at the secondary level-- and to create *a place for children to build their knowledge through exploration and ludic activities,* as Papert proposed. Logo was the ideal tool for that purpose, as it created what Papert called the conditions or micro worlds for students to root intellectual models. In fact, Logo delivered both a powerful instrument and a learning proposal for promoting mental processes and for solving problems, as Papert noted. So programming at elementary schools with Logo, and a project-based learning approach constituted the core of the soon to be launched project.

The project's continuous focus on children´s use of programming to develop their creativity and thinking skills has distinguished it from other projects in the region, and it proves to be still in force as more and more countries are launching initiatives to introduce these types of learning opportunities in their curricula.

Concerning the need to answer the ***how*** of the project in terms of its organizational structure and dimensions, it was clear to the founding group that the model should be able to overcome bureaucratic obstacles and political cycles. Thus, a non-for-profit foundation was established, the Omar Dengo Foundation (FOD), to take on the development and implementation of the program in partnership with the Ministry of Public Education (MEP). In order to guarantee the link between the two organizations, the Minister of Education was initially appointed head of FOD´s Board of Directors. This public-private arrangement provided a good combination that included the needed efficiency and relevance conditions.

In 1988, the Project began establishing *Educational Informatics Laboratory* (name the computer labs received) in 57 elementary schools. Two lessons a week (90 minutes) were devoted for students to work at these computer labs, and a *tutor* for each lab was appointed among the schools' teachers. At that time, FOD assumed all the implementation components of the project: financial resources, training of teachers, selection and purchasing of computers, hardware and software installation and configuration processes. Schools and their local communities were required to facilitate and provide conditions for setting up a classroom to house the school´s computer laboratory.

And thus gently but firmly, the project became a reality.

First 10 years: growth, sustainability and diversification

- The educational proposal
- *Multigrade*, one teacher school: an important variation
- The tutors
- The funding

After the initial launching to the project at schools, it became evident that the educational proposal was difficult to implement, although the initial enthusiasm of students, teachers and principals to the novelty of the proposal counterbalanced the difficulties. The educational proposal began to be enriched complementing the original concept through the introduction of robotics, telematics and others. Extracurricular clubs were created in some cases to facilitate the incorporation of these new options, where students would voluntarily participate in a non-formal complementary learning opportunity.

Eventually, new teaching and learning scenarios emerged, especially the so called *Multigrade* schools. These are very small rural schools, with a total of less than 30 students, usually headed by one teacher in charge of all grades (from 1st to 6th).

However, to facilitate the growth and sustainability of the Project, other things were needed in addition to a solid pedagogical approach. FOD then built an articulated and efficient program architecture around the following main pillars:

- A continuous training, support and follow-up system for teachers: The system comprised the provision of an initial training program for incoming teachers, annual mandatory training workshops and continuous pedagogical face-to-face support from a team of advisors who visited the schools with regularity. A specialized computer-lab teacher position was lobbied and eventually created by the MEP; and education departments of the main public universities, designed and began offering the new training programs, initially with the guideline and support from ODF.
- A sustainable and constant financing source: After ten years of launching, the program managed to growth and be available in 181 public elementary schools benefiting 148,000 students nationwide. In 1988, due to the Program's efficient performance, the Costa Rican government decided to strengthen its investment

in it, and in order to provide appropriate financial support, the MEP would annually transfer resources to it from the national budget. ODF continued bringing into the project funding to incubate new ideas, to support research and evaluation, etc. but from this moment on the main funding has come from the MEP.

- Logistic, administrative and technical services were designed: evolving from a small contained initial effort, to a large scalable initiative.
- A research, monitoring and evaluation process to provide feedback to the educational component of the program and to contribute to its enrichment and advancement.

The second decade: a national program

- 2002: Secondary schools are incorporated
- 2007: Educational Informatics goes to Technical High Schools
- The renovation of the didactic proposal

By the year 2002, the Ministry of Public Education and the Country´s Superior Council of Education decided to transfer to the Omar Dengo Foundation the effort that MEP had initially implemented to bring computers into the secondary-level schools. This decision was based on an evaluation that showed evidence of limited results of the MEP´s project at secondary-level schools and on the need to have a more articulated national effort to undertake educational informatics initiatives in the public school system. So a National Program of Educational Informatics was created as an integrated initiative from grades K through 9.

In 2007 a complementary educational proposal emerged for the final years of the educational system, that is, grades 10 to 12, in technical high schools, with the specific aim to develop entrepreneurship skills in students using digital technologies and virtual environments to simulate real world experiences.

In 2008, ODF started a review process with the intent to generate a more precise, coherent and integrated educational proposal for all levels, with appropriate scaffolding to guide the teachers' work in the Computer Labs. The goal was to clearly define the minimum learning outcomes expected to occur at each level, and based on that, provide didactic guidelines to the EI Lab teachers as references and support. The initial training of teachers by the universities was not providing the required profiles, and the needed changes were difficult to promote, given important structural limitations such as the minimum requirements faced by students wishing to follow a career in Education, and the usually low demanding training being offered to them. In addition to this limited initial training in a notable proportion of teachers, the years had also allowed a better understanding of the scaffolding needed by teachers, in general, to implement an ambitious and demanding proposal as PRONIE's, and to be able to lead their students along the path to deep and effective learning.

By the end of this period (2009), 974 schools and 453.826 students were participating in the Program.

The last five years: evolution and permanency

- Mobility to support the curricula
- The equity perspective
- An enriched and integrated educational proposal
- Professional development and eLearning

The evolution of the Program in the last five years has been marked by:

- Irruption of mobile digital technologies, the broader possibilities to use them in regular classrooms and the widely available practical knowledge on how to use them to promote the new pedagogies needed in the 21st century.
- Increasing international calls to have every child learning how to program and develop their own software, what has been recently named as *computational thinking* (Wing, 2006, 2010; Grover and Pea, 2013), and that Papert usually would refer to as computational or computer culture (1976, 1980, 1996, 1999).
- Development of applications and knowledge based on digital learning platforms and their potential to enhance learning opportunities both for students and for teachers.

Let us see next how the program has responded to these new possibilities and demands:

1. In the last five years, the Program has made significant efforts to promote the use of digital technologies in regular classrooms and not only in computer labs, thus aiming to enrich content learning in key subject areas and develop the strategic skills that students need today. At this moment, students and their teachers in over 1000 primary and secondary schools have been provided with laptops to be used in their classrooms –and outside of them-, and FOD´s goal is to reach by 2017-18 all the country´s schools not yet in the program.

Two main pedagogical schemes have been explored: (a) one-to-one laptop programs, aimed at facilitating ubiquitous access to technology to primary and secondary students in rural areas, thus allowing students in such areas to experience the potential of the computer as a personal tool for learning, production and creation; and (b) laptop cart programs, installed at both primary and secondary schools, with a focus in reinforcing research, reasoning and creative skills in curriculum subjects such as language, mathematics and science. These approaches have been designed to promote equity in a society like the Costa Rican, which has been easily and rapidly adopting new technologies, with a majority of homes nowadays having a computer and internet access.

One-to-one laptop programs have thus been intended for those rural schools where the digital gap is deeper. The pedagogical vision behind these initiatives is to take seize of the potential of technology to amplify and deepen our students' capacity to build knowledge individually and collectively and to apply it. The vision also seeks making classroom learning more personally meaningful, dynamic and attractive for students. The Program sees digital technologies as powerful tools serving the kind of learning most needed for today's and tomorrow's world: ubiquitous, connected and self-managed learning.

2. As a second complementary educational strand, the program has also been working in the consolidation of its educational proposal for the computer-lab educational informatics lessons by enriching and deepening it with new conceptual approaches and experiences that have evolved from the constructionist framework. The program is visualizing an enriched computer-lab proposal, no longer constricted to a physical facility (the computer lab) seeking to benefit all students in the country. Drawing on its long experience in teaching programming and the use of technologies to promote students' high-order capacities, the program is now looking into new and exciting perspectives such as physical computing, while strengthening its learning approach to problem solving, computational concepts and computational practices.

3. As a third educational strand, the program has been designing and producing a new learning extracurricular proposal for students to expand their learning possibilities according to their individual interests and talents and reinforce their self-learning capacities. Most of these new learning opportunities are delivered on-line, through a digital learning platform.

These three strands are unified under an integrated model, a well-defined strategy to use technologies as thinking tools, to stimulate the development of specific student's capabilities and competencies that are highly valued in the modern society.

The configuration of this educational offer has required the design of a new model of professional development for teachers which relays more in virtual platforms, and it is more strategically oriented to accompany teachers progressive evolution towards higher levels of ownership and innovation (see www.upe.ac.cr).

Conclusions and perspectives

- Relevant results
- The public- private partnership: its strengths and weaknesses
- Lessons learned and the challenges ahead

Throughout the years, the program has continually monitored and evaluated its educational proposals although a large-scale evaluation of students learning outcomes has been hard to organize – among other reasons, due to the difficulty of having access or developing the appropriate assessment instruments capable of measuring the kind of dispositions and deep thinking skills that were aimed at.

However, increased efforts have been recently directed to measure the Program’s outcomes in terms of the student’s abilities (the individual domain). Last year, close to 10,000 students

that had just finished elementary school (6th grade) were incorporated in a study to evaluate their learning levels, based on the program´s didactic guidelines and its learning outcomes framework. The evaluation results show a statistically significant and positive accumulative effect in students participating during several years in the program, improving their performance in dimensions such as problem solving.

With regard to the other dimensions where effects of the Program were also expected (in the economic, the social and the educational domain), the results are varied. In the economic domain, the program’s impact on the national productive structure and economy has been regarded to be clearly positive. Though hard to be rigorously assessed, the fact of having a whole new generation of children who got familiarized with technology much sooner than children from other Latin American countries, who learnt programming at an early age and had the experience of being in control of the machine and not in a position of receiving orders from it, is commonly recognized in the country as one of the key factors explaining the exponential surge of its high technology industry sector.

Other very visible impact of the program has been its contribution to reducing the digital gap inside the country. The slowly increasing technology access rate, especially in rural areas, has been compensated by the possibility for many children and young people of having access to technology in the school – and, furthermore, accessing it in the context of learning experiences designed to foster their creativity and thinking.

On the contrary, the renovation of the school system has proven to be a difficult goal to achieve. To begin with, it was perhaps a too optimistic goal to pursue. Notwithstanding the powerful message broadcasted from the educational informatics Lab, a synergistic and multidimensional plan is needed to cause the system to change.

The public-private partnership established with the Ministry of Public Education from the beginning, has proven to provide strengths and weaknesses, with an overall positive balance. Among the positive elements, evidently, is the sustainability of the program, with

29 years of continued work, and its articulation with the country’s education policy. Notwithstanding several political administrations that have taken office during almost three decades, the program has grown constantly in terms of coverage, and its educational proposals have evolved dynamically, in respond to lessons learned, accumulated experience, and new and inspiring technical possibilities that had arisen.

Nevertheless, the political cycle has proven to be a challenging process, with new authorities coming every four years, and an intense transitional route that needs to be taken of every time. This includes explaining the program, its objectives and results, mutually understanding the role of each partner, synchronizing the program with the political priorities of each administration, and yet maintaining the fidelity of the program´s objectives and its epistemological framework which are critical elements of that process.

The economic limitations of the country have resulted in a slower progress of the program through time than most of us would have liked, and 100% coverage is still a goal to reach,

which would be hopefully accomplished very soon. In addition, new requirements have emerged, such as access to broad band and good quality internet services, which in Costa Rica are under the responsibility of MEP and the Ministry of Science, Technology and Telecommunications.

Improving the training of teachers --both initial and continuous training--, is probably the most determining factor in any educational proposal, and certainly it is crucial in our program. The complementary efforts that FOD has developed in this field are important, but they do not substitute a sound training process along the four or five years of formal university education. On the other hand, we need to better understand how to use new technologies to support teachers training process. FOD´s new proposal is aimed at reaching this goal.

References

Fallas, I and Zúñiga, M. (2010*). Las tecnologías digitales de la información y la comunicación en la educación costarricense*. In: State of the Nation Third Report. San José: Programa Estado de la Nación. (http://www. estadonacion.or.cr/files/biblioteca _virtual/educacion/003/ Fallas_Zuniga_2010_TIC_Educacion.pdf)

Fonseca, C. (1991). *Computadoras en la escuela pública costarricense: la puesta en marcha de una decisión.* San José: Fundación Omar Dengo.

Grover, S. and Pea, R. (2013). *Computational Thinking in K–12: A Review of the State of the Field.* Educational Researcher, 42 (1), pp. 38–43.

Papert, S. (1976). *An Evaluative Study of Modern Technology in Education.* MIT Artificial Intelligence Laboratory Memo No. 371. Available at: http://www.papert.org/articles/AnEvaluativeStudyofModernTechnology.html

Papert, S. (1980). *Mindstorms*. New York, NY: Basic Books.

Papert, S. (1996). *An Exploration in the Space of Mathematics Educations*. International Journal of Computers for Mathematical Learning, 1 (1), pp. 95-123. Availalble at: http://www.papert.org/articles/AnExplorationintheSpaceofMathematicsEducations.html

Papert, S. (1999). *What is Logo? Who Needs It?* In: Almeida et al. *Logo Philosophy and Implementation* (pp. IV-XVII). Logo Computer Systems. Available at: http://www.microworlds.com/company/philosophy.pdf

Wing, J. (2006). *Computational thinking*. Communications of the ACM, 49(3), pp. 33–36. Available at: https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf

Wing, J. (2010). *Computational Thinking: What and Why?* Available at: http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Keywords
Educational informatics, thinking tools, thinking skills, programming, pedagogical proposal

Designing Interfaces for Special Needs

Michael Weigend, *mw@creative-informatics.de*
Holzkamp Gesamtschule Witten, Willy-Brandt-Str. 2, 58453 Witten, Germany

Abstract

This contribution presents a junior high school workshop on designing physical interfaces with Scratch, the PicoBoard and Sensors. At the beginning the students worked on given starter projects including a car race simulation and a simple version of the game “Pong” using step-by-step instructions. These first projects did not involve the PicoBoard. Sprites were controlled with keys. But it was rather obvious that the keyboard could be replaced by external sensors. In the next phase the students got a PicoBoard and tried out different types of external sensors, including force-sensors, flex-sensors and prototype switches made of everyday material like paper, plastic bottles, wire, sticky tape and aluminium foil. Having gained some experience with this technology, the students were now supposed to develop their own Scratch project, which could be (but did not need to be) inspired by one of the five given starter projects. The only condition was that the PicoBoard had to be involved. Among the student’s projects were a lunar landing simulation with a self-made tilt sensor (made of a plastic bottle) and a foot switch and several versions of the classic computer game “Pong” using external switches and audio feedback. Before starting the development, the students (n = 17) answered a questionnaire on personal motives and general design decisions. Why did they pick a certain project? Which aspects raised their curiosity? According to this survey, the most interesting aspect of sensor technology is the possibility to make a simulation project more realistic. The average rating of this design goal on a scale from 0 (uninteresting) to 3 (very interesting) was 2.12). The design goal “Make a digital application accessible to handicapped persons” got lower scores (average: 1.76) but was still quite relevant. In the context of this workshop, sensor technology did not seem to be a primary motive for programming. Students tended to develop a Scratch project, because they had specific ideas about desired functionality and not because they wanted to work with certain sensors.

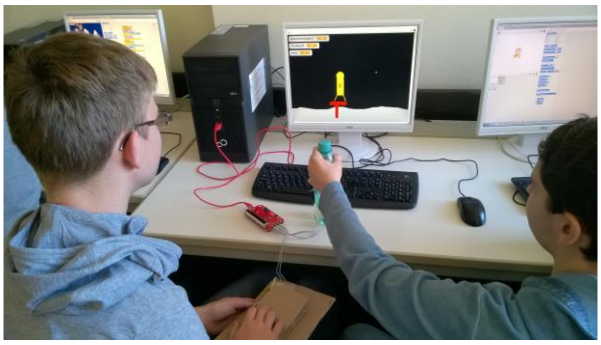


Figure 1: A “Lunar Lander” game with a tilt sensor made of a plastic bottle.

Keywords

Scratch, PicoBoard, physical computing, sensor

Cyborgs and Digital Prostheses

A cyborg (acronym for *cyb*ernetic *or*ganism) is a being with both organic and artificial parts. In 1960 Manfred E. Clynes and Nathan S. Kline introduced the term for human beings with incorporated devices which make them suitable to survive under special conditions like long space missions. “The Cyborg deliberately incorporates exogenous components extending the self-regulatory control function of the organism in order to adapt it to new environments.” The German science journalist Ranga Yogeshwar suggests that we are all on the way to become “machine-men” (WDR, 2014) and he sees Nomophobia as an indicator for this. Modern mobile devices are explicitly designed to be extensions of individual’s minds. The software running on a smartphone is personalized. It learns and remembers individual preferences and it adapts automatically to location, time and movements.

Digital technology can replace missing body functions to some extent. Consider the famous physicist Steven Hawking suffering from amyotrophic lateral sclerosis (ALS) that has gradually paralysed him. Having lost his speech completely in 1985 he started to use a computer for communication in the year 1986. Using one hand and since 2005 his cheek muscles, Steven Hawking manipulates an input device that enables him to select or to spell words. In this case the technical challenge is to create a special input device using force sensors or special switches. Data from force sensors are also used for microprocessor controlled prosthetics like the “C-leg” (Carroll & Edelstein 2006).

There are 39 million blind people on the planet (World Health Organisation, 2012). One approach of vision sensory substitution is to represent images by sound. For example, EyeMusic adopts a sweep-line technique where each image is processed column-by-column from left to right constructing a “soundscape”. It uses Instruments for different colors (for example strings for yellow and brass for blue) and a pentatonic music scale to make the soundscape a music-like pleasant experience (Abboud et al. 2014). Among the blind community general vision-to-audio sensory substitution is not much used regarding the general perception of images. But when it comes to more specific problems there exist usable digital tools. For instance, there is digital support for clothing matching for blind people (Tian & Yuan 2010). This illustrates that there is much room for creativity in the field of medical computer science which makes it an inspiring context for classroom projects (Strecker, 2013).

Workshop on Designing Special User Interfaces with Scratch

In this section I am going to describe a workshop, in which students (grade 10) develop a Scratch application with a special user interface using the PicoBoard and external sensors. The basic idea is to combine the fun of creating colourful games with the more serious intention to better the quality of life. The workshop consists of five parts.

1. Introduction. The students watch sequences from a movie about cyborgs and the usage of sensors and computers in medical technology for creating prostheses (WDR, 2014). The general idea of the workshop is presented: The students are supposed to develop their own Scratch with some kind of self-made physical interface.

2. Starter projects. The students study a collection of five starter projects. They are supposed to implement a car race simulation and then may implement as many further projects as they want. For each of the starter projects there is a small manual (two pages) with step-by-step instructions and explanations. In order to reduce the complexity, these projects do not involve the Picoboard. Sprites are controlled with keys. But they all have the obvious potential to be extended using external sensors instead of the keyboard.

3. Trying out sensors. After a short introduction, all students get a PicoBoard and get the opportunity to try out different types of external sensors:

- A force sensor (force sensitive resistor), that changes its resistance when force is applied on it.

- A flex-sensor, which changes its resistance when it is bent.
- A prototype tilt sensor consisting of a plastic bottle partly filled with water and four needles.
- A prototype switch made of paper and aluminium foil.

.

4. Reflection. The students answer a questionnaire and reflect experience, personal motives, general design decisions etc. Then they decide which project they would like to develop and write a description of their project idea.

5. Development. Individual students or teams of two students develop their project. If they want they can present it to the public at the open-door day.

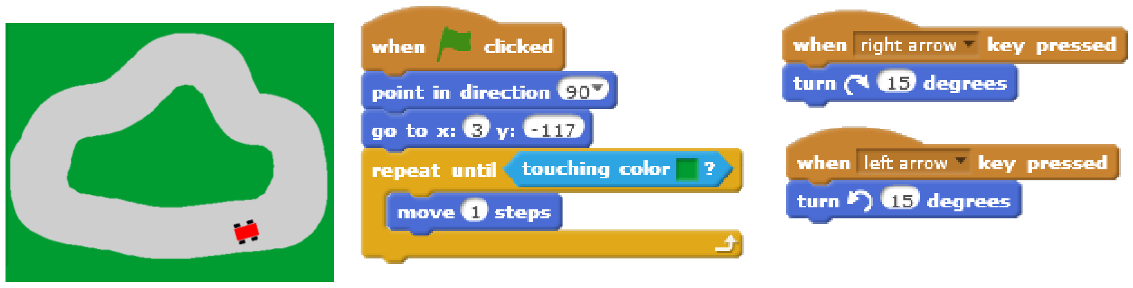
Starter Projects

The students are asked to implement at least one starter project. The first project ("Formula One") is a simple simulation of a car race. All students start with the same project to make helping each other easier and to keep the start smooth. In contrast to the remix-idea of Scratch, the starter projects are not imported from the scratch repository. The students implement them from scratch using a manual. The manual of the first project is more elaborated than the others and it covers relevant details of the Scratch user interface. After that the participants choose further projects from a small selection at will.

Project 1: Formula One

The first project ("Formula One") is a simple simulation of a car race. The user steers a car with the arrow keys. It has to stay on the grey track. When it leaves the road and touches the green, it stops.

This project adopts basic programming concepts, like loops, conditions and output via speech bubbles, but no variables. Relevant mathematical and physical concepts are Cartesian coordinates, angles, movement and speed. Although the scripts are short and simple, they are not trivial for students in grade 10.



```
when ⚑ clicked
point in direction 90
go to x: 3 y: -117
repeat until touching color 🟩 ?
    move 1 steps

when right arrow key pressed
turn ↻ 15 degrees

when left arrow key pressed
turn ↺ 15 degrees
```

Figure 2: "Formula One" – Controlling a car with constant speed.

Project 2: Lunar Lander

In this classic of computer simulations, the user has to land a rocket on the moon with a limited amount of fuel. The space ship approaches the surface attracted by gravity. The pilot can switch on the thrust by hitting the space key. The space ship decelerates and fuel is consumed. In this project variables are introduced. This project introduces variables and moving a sprite by changing the y-coordinate of the position. Relevant physical concepts are speed and acceleration.

Project 3: Pong

Pong is one of the first computer games. It was published 1972 by Atari. In this project a simple version for one player is implemented. Specific programming concepts are bouncing, random numbers, changing the direction of motion.

Project 4: Talking Machine

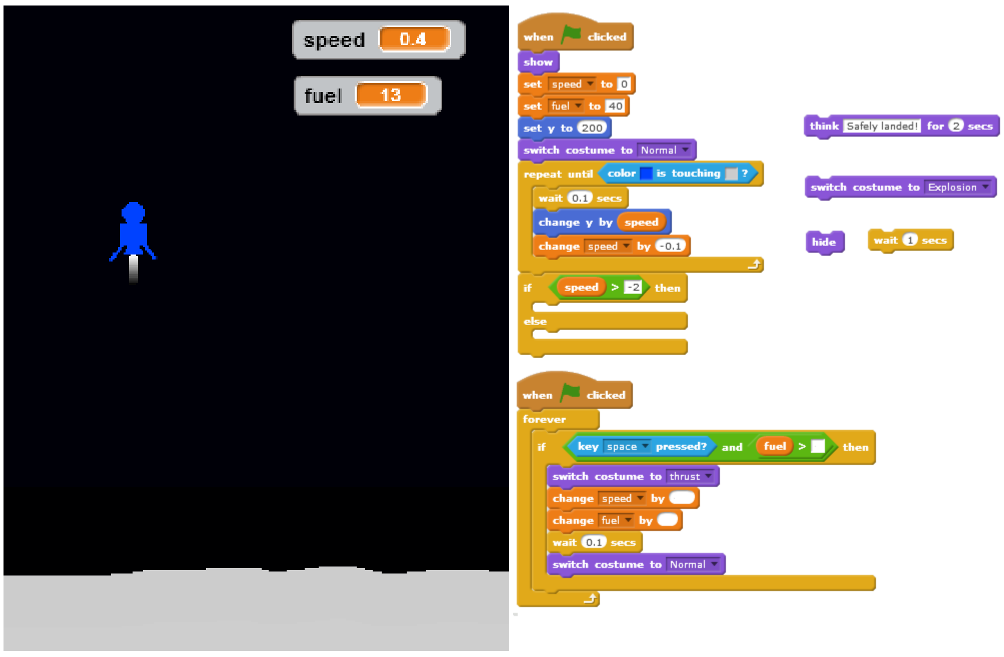
This is a non-game application taking care of the communication problem of Steven Hawking. The prototype has just one sprite, depicting a hand, which can be moved with the up-arrow key and the right-arrow key. On the backdrop of the stage there are different words in different colours. When the hand touches a word, the application says the word (audio) und the hand jumps back to its starting position in the lower left corner. This project introduces recording and playing sound.

Project 5: Sprinter

The Scratch cat runs from start to goal. The user drives the cat by hitting two keys (say A and B) alternately. When the user hits key A, the cat moves 10 steps. Then the key is locked. When the user hits it again, nothing happens. She or he has to hit key B next. Then the cat moves another 10 steps, key B is locked and key A can be used again, and so on. When the cat reaches the goal, it says the time that has been needed for the sprint like "You needed 5 seconds." For locking and unlocking key a variable check is used. The students are encouraged to follow the instructions in the manuals (instead of getting explained everything by the teacher) but they are also encouraged to experiment, to try changes, implement their own ideas and to create their individual versions. An important feature of each starter project is that it not just represents a project idea but also explains some programming technique. If different teams implement different starter projects introducing different programming techniques they can share expertise and the class as a whole quickly gets a high level of competence.

Manuals

The students implement projects from scratch using given manuals. A manual includes step-by-step instructions with images and little text. They may contain the complete code or a "code puzzle".



```
when [flag] clicked
show
set speed to 0
set fuel to 40
set y to 200
switch costume to Normal
repeat until color [ ] is touching [ ] ?
  wait 0.1 secs
  change y by speed
  change speed by -0.1
if speed > -2 then

else

```

```
think Safely landed! for 2 secs

switch costume to Explosion

hide      wait 1 secs
```

```
when [flag] clicked
forever
  if key space pressed? and fuel > [ ] then
    switch costume to thrust
    change speed by ( )
    change fuel by ( )
    wait 0.1 secs
    switch costume to Normal
```

Figure 3. Screenshot from Lunar Lander and code puzzle.

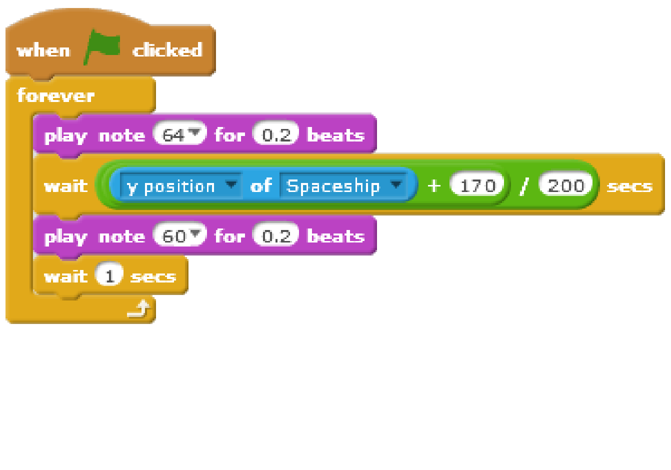
A code puzzle consists of code fragments, which have to be put together properly. The image depicts an easy code puzzle for the Lunar Lander project. The first script controls the movement of the landing rocket. The second script takes care of changing the speed. The difficulty of a puzzle can easily be adjusted by the amount of coherent code that is already given. In this example, all the students have to do is add the blocks at the right hand side to the first script and add appropriate numbers in the second.

Extending the Starter Projects – Designing User Interfaces for Special Needs

The Starter projects do not require the PicoBoard. All input is done via the keyboard. The students were asked to pick a project idea and develop an application for a person with a special need. The students had to take care of input, output and computational extensions. For the input they should use the Picoboard. Regarding the output they had to add sound features to make visual information perceivable to those who cannot use their eyes for some reason. Computational extensions are often necessary to make the application "smarter" and easier to use. In this section I will sketch some ways how to adapt the software.

Output

Extension 1: To make a game suitable for a blind player, relevant visual information must be represented in a non-visual way, for example by sound. There are many ways do this. Numbers can be represented by pitch, volume or the time between two sound signals ("Radar signals"). Figure 4 depicts a solution for the "Lunar Lander". These two scripts can just be added to the already existing scripts of the rocket-sprite. The left hand script produces "Radar signals" like "ding ...dong ....ding...dong". The time between "ding" and "dong" corresponds to the height of the rocket. The remaining amount of fuel is told through speech that has been recorded before (right hand script). For example, when the content of variable fuel goes beyond 30 the system plays a message like "Attention, you have less than 30 units fuel."

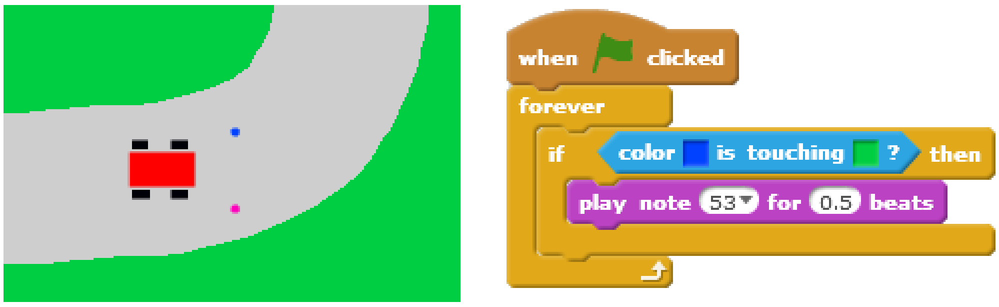


```
when clicked
forever
    play note 64 for 0.2 beats
    wait (y position of Spaceship + 170) / 200 secs
    play note 60 for 0.2 beats
    wait 1 secs
```

```
when clicked
wait until Fuel < 30
play sound Fuel 30
wait until Fuel < 20
play sound Fuel 20
wait until Fuel < 10
play sound Fuel 10
wait until Fuel < 5
play sound Attention
```

Figure 4. Two additional sound producing scripts

The car of the starter project "Formula One" gets "sensors" (in figure 5 two dots of different colours in front of the car) that trigger a sound, when the car is too close to the right edge and a different sound, when it is too close to the left edge. This way the car can be steered by a blind person. The PicoBoard is not required, since it is just an input device.



```
when 🏁 clicked
forever
    if < color 🟦 is touching 🟩 ? > then
        play note (53▾) for (0.5) beats
```

Figure 5. A car with "sensors" triggering sounds, when it gets too close to an edge of the road.

Input

The car of the starter project "Formula One" can be steered with an external sensor. This could be a steering wheel with a potentiometer, one or two force sensors, which are connected to the resistance inputs of the PicoBoard. Using a flexible force sensing resistor the car can be steered with a single finger. A tilt-sensor can easily be built from a plastic bottle and four needles: Press needles through the plastic into the bottle. At each end a pair of needles. Connect the needles pair wise with resistance inputs a and b. Put water into the bottle. When the bottle is tilt, one pair of needles gets galvanic connected by the water. When you put this device on your back, you can steer the car by bending your body to left and right. In each project the input can be designed in a way that a video game becomes a physical exercise. In case of the project "Sprinter" the two keys that have to be hit one after another can be replaced by self-made switches or force-sensitive resistors, which are connected to the resistance inputs of the PicoBoard. The two switches can be placed a few meters away from each other, which will cause quite some effort to use them. For this type of user interface a command like "when a-key is pressed" is simply replaced by "when resistance-A < 50". Another approach is to use the exact sensor value. For example, the speed of the moving cat in the "Sprinter"-project can be set using the light sensor. The user operates a dynamo lamp as hard as possible and shines light on the PicoBoard. In this way the cat's speed is determined by the power the user invests in operating the dynamo lamp.

Computation

Using sensors instead of direct vision to drive a car is difficult and dangerous. If the goal is to enable blind persons to drive a car, it would more realistic to adopt autonomous driving. The car finds its way along the road automatically (using sensors). The driver can start and stop and choose the direction where to go. For example, when the blind user turns to the right the car follows the road and turns to the right at the next junction.

Motives for Project Development and Design Goals

In the evaluation phase, just before the students had to decide, which project they wanted to develop for the open-door day, they were asked to reflect their programming experience and motives for further development. 17 students rated the following general goals on a scale from 0 (completely uninteresting) to 3 (very interesting).

- Make a simulation project more realistic. For example a car simulation is more realistic, when the car on the screen is controlled by a stirring wheel and pedals with force sensors.
- Support physical training. Using a tilt sensor made of a plastic bottle and needle pins, a car on the screen can be steered by bending the body. A virtual bicycle could be ridden by hitting two switches on the floor alternating with the left and right foot. It requires real physical effort to make the bicycle move on screen.
- Improve the richness of a fantasy activity and make it more attractive. Digital applications may be more fun, when unusual input tools are involved. This might also imply sounds and visual effects.
- Make a digital application accessible to handicapped persons. For instance, make it usable for blind persons or persons, who cannot use their hands.

General goal	Avg. score
Make a simulation more realistic	2.12
Make a project supporting physical exercise	1,94
Make the user interface richer, add sound	1.88
Make a project usable for handicapped persons	1.76

Table 1: Interest in goals of development (3 = very interesting, 0 = completely uninteresting, n = 17)

The design goal ”making a simulation more realistic” got the highest scores. However, the idea to help handicapped persons was quite well accepted too. Another section of the questionnaire covered more specific design goals related to the starter projects. Since everybody had made the driving simulation, most questions were related to this. It turned out that projects which required audio representations for blind people got a rather low score.

Specific goal	Avg. score
Driving simulation: Autonomous driving, car stays on the road.	2.53
Pong for more than one player and scoring	2.53
Driving simulation: Different levels of difficulty	2.25
Driving simulation: Steering wheel and accelerator	2.18
A car race as a physical exercise	2.18
Driving simulation: "Sensors" indicate low distance to the border by sounds	1.88
Driving simulation: Steering by body movements	1.53
Lunar lander for blind persons.	1.53
Pong for blind persons	1.19

Table 2: Interest in specific goals of development
(3 = very interesting, 0 = completely uninteresting, n = 17)

No.	I choose a certain project, because...	Avg. level of agreement
1	… I already have a great idea for it.	2.71
2	… it gives me the opportunity to learn something new.	2.13
3	… I can use things I already know for it	2.12
4	… I think it is something useful.	2.06
5	… it is not too difficult and I can certainly finish it.	1.94
6	… I can use a certain sensor.	1.53
7	… I can finish it most quickly.	1.24

Table 3: Level of agreement to statements about personal motives for project choice
(3 = I do completely agree, 0 = I do completely not agree, n = 17)

Finally, the students answered questions about personal motives that have influenced their choice of project. The results in table 3 show that the students want to implement their own ideas and use the knowledge they already have. But they also want to learn. “Usefulness” gets only medium score and the specific sensors seem to have not too much influence on the decision which project to choose. Although the intention of the workshop was to inspire students to develop special interfaces for handicapped persons, the participants did not put too much effort on this design goal. A student working on a version of "Pong" with audio feedback indicating the position of the ball said to me that he had serious doubts that a blind person would like to play his game. This statement points to a problem of this workshop. Developing digital devices for persons with special needs remains a "sand box game", when there is no collaborating with real persons who could try out and use these artefacts.

Conclusions

With Scratch plus the PicoBoard and external sensors students can create a big variety of digital artefacts. Projects become more serious. "Toy projects" that are easy enough to be implemented by beginners can model digital technology that is highly relevant in present and future life. In this context the sensors as such do not seem to be very inspiring. The starter projects can be implemented with the keyboard instead of external devices, which have to be built first. This seems to be enough for many students. The use of external sensors is an additional troublesome difficulty they have to cope with. Sensors cause poorer performance of the Scratch scripts, which makes it necessary to tune and optimize scripts.

However the mere possibility that the input could be customized using external sensors might increase the value and seriousness of a project. While constructing devices and trying out the classmates' products, students think about human aspects of digital technology. Topics like digital prostheses, cyborgs, autonomous driving, inclusion through technology, can be discovered in a constructionist way.

References

Abboud, S., Hanassy, S., Levy-Tzedek, S., Maidenbaum, S., & Amedi, A. (2014). EyeMusic: Introducing a "visual" colorful experience for the blind using auditory sensory substitution. issues, 12(13), 14.

Carroll, K & Edelstein, J. E. (2006). Prosthetics and Patient Management: A Comprehensive Clinical Approach. Slack Incorporated.

Clynes, M. E. & Kline, N. S. (1960) Cyborgs in Space, Astronautics, September 1960, 29-33. Online available: http://de.scribd.com/doc/2962194/Cyborgs-and-Space-Clynes-Kline
de Lange, C.: The man who saves Stephen Hawking's voice. New Scientist 2011-12-30. URL: https://www.newscientist.com/article/dn21323-the-man-who-saves-stephen-hawkings-voice.

JB, B., Mathew, P., Thulasi, P. C., & Philip, J. (2013). Nomophobia-Do we really need to worry about?. Reviews of Progress, 1(1).

Strecker, K., & Weg, F. (2013). Medizinische Informatik in der Sek. I. [Medical Computer Science in Lower Secondary Schools] INFOS 2013 proceedings, Springer, 25-34. Online available: http://cs.emis.de/LNI/Proceedings/Proceedings219/P-219.pdf

Tian, Y., & Yuan, S. (2010). Clothes matching for blind and color blind people. In Computers Helping People with Special Needs (pp. 324-331). Springer Berlin Heidelberg.

Westdeutscher Rundfunk (2014): Quarks & Co: Mensch 2.0 Wie wir zum Maschinenmenschen werden [How we become Machine Men], 45 minutes, broadcasted on 2014-02-11. URL: http://www1.wdr.de/mediathek/video/sendungen/quarks_und_co/videosensibletechnik102_size-L.html

Eliciting Engineering Expertise from Novices

Tamar Fuhrmann, *tamarrf@gmail.com*
Transformative Learning Technologies lab, Stanford University, CERAS 232, CA, US

Marcelo Worsley, *worsley@usc.edu*
University of Southern California, 3470 Trousdale Parkway, WPH-600A, Los Angeles, CA 90089

Paulo Blikstein, *paulob@stanford.edu*
Transformative Learning Technologies lab, Stanford University, CERAS 232, CA, US

Abstract

Principle-based reasoning refers to a problem solving strategy that is based on principles. Literature on principle-based reasoning suggests that it is more employed by experts than novices and that it provides a means for advancing one's designs. However, effectively leveraging principle-based reasoning has historically been challenging. Nonetheless, we argue that constructionist learning, especially in the specific context of engineering design, can be a fertile space for promoting principle-based reasoning and object closeness, even among novices. In this paper, we propose a new model to advocate engineering expertise in novice students by encouraging them to engage principles. Our report includes results from two studies both with novice students: One study (N=22) was implemented in a classroom with first grade students. The second study (N=20) was conducted with high school and undergraduate students. Across these two studies we used a very similar model for engaging students in the opportunity to leverage engineering principles in approaching open-ended engineering design tasks. Two particular interests of this study were to: 1. Examine ways to effectively evoke principle-based reasoning from novices, 2. Examine if the principle-based reasoning strategy has utility among novice students. Results show that the model employed enabled novice students to demonstrate expert-like practices; novice engineers, first grade as well as high school students, can effectively make use of principle-based reasoning.

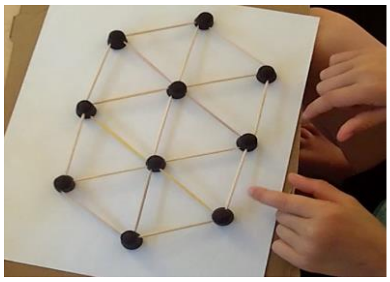


Figure 1. Designing the foundation of the structure with principle-based-reasoning strategy

Keywords

Principle-based-reasoning; engineering curriculum; expert and novice; engineering design cognition

Introduction

With the Next Generation Science Standards (NGSS) comes a renewed interest in promoting engineering education among K-12 students. According to NGSS the performance expectations for both students and teachers regarding engineering practice are high. Nevertheless, the standards provide very little in the way of practical level knowledge about how to support students learning in STEM. In this study we attempted to examine engineering expertise in a classroom setting. More specifically, we introduced a new model to advocate engineering expertise in novice students by encouraging them to engage in engineering principle-based reasoning. Principle-based reasoning is a strategy used in the process of solving new problems based on principles. When using principle-based reasoning, students start with engineering principles and work their way down to their solution. The use of principle-based reasoning is often associated with high levels of expertise (Anderson & Greeno, 1981; Chi et al., 1982; Moss et al, 2006). Moss et al (2006) is a prime example of such findings. Moss and colleagues compared freshman and senior undergraduate students and noted that when faced with an engineering design task, the seniors used a more principle-based approach for recalling the design. The authors inferred that the seniors were able to draw from principles that were not yet salient to the younger students. Other work would also lead one to the same conclusion that principle-based reasoning is more closely associated with experts than with novices (Chi et al., 1982; VanLehn, 1996). However, one should not necessarily interpret principle-based reasoning as being a privileging of the abstract. On the contrary, one way for considering principle-based reasoning is as an example of students getting "closer" to the objects being created (Turkle & Papert 1992). In essence this means having students more deeply connect with how the object moves and behaves. This can be seen as a contrast to establishing a surface level (Chi & VanLehn, 2012) connection with a given object. Even so, principle-based reasoning can be quite challenging. For one, if students aren't experts in a given domain, they can't be expected to properly understand the relevant domain knowledge, nor how that knowledge relates to the other pieces of information from that discipline (Nokes, Schunn, & Chi, 2010). Similarly, work by Hammer (2004b) and Wilson et al. (2006) both find that students have difficulty using causal, or mechanistic, reasoning in the domains of biology and physics. Specifically, Hammer (2004b) shows many instances where students' causal reasoning quickly deteriorates, and where students fall into the trap of relying on formulas, instead of building on and considering the principles associated with those formulas. Another common challenge within principle-based reasoning is that students sometimes dwell on the wrong details for a given problem (Chue & Lee 2013). If this is the case, even if students start to think like an expert, their efforts will be in vain. Given the assorted complications that exist for effectively promoting principle-based reasoning, this paper proposes a new model to encourage principle-based reasoning in an engineering design context. Our report includes results from two studies both with novice students: One study (N=22) was implemented in a classroom with first grade students. The second study (N= 20) was conducted in a classroom with high school students and undergraduates. Two particular interests of this study were to: 1. Examine ways to effectively evoke principle-based reasoning from novices, 2. Examine if the principle-based reasoning strategy has utility among novice students.

Methods

Participants and settings

Study 1: This study includes 22 first grade students. It lasted for a total of 5 lessons in a science class. The class took place at a K-8 school, which serves the residents of one unified school district in the San Francisco Bay Area. Parents' participation is an important part of the school philosophy. The flexibility of the curriculum, the child-centered environment together with the fact that one of the authors of this paper is a parent at the school, enabled us to provide this set of activities as part of the science curriculum.

Study 2: This study includes 20 high school students and early undergraduate students. The study was conducted at a private research-one, West Coast University. The high school students were participating in a two-month long intersession period, while the undergraduate students were recruited to participate in the study as paid research participants.

Instructional sequence

Study 1: Students designed and built a model of a stable three-dimensional structure using gumdrops and toothpicks. They scaled up the model and built a “small village” of geodesic dome structures made from paper rods and finally from PVC (Polyvinyl chloride). To help facilitate the learning experience, students completed the following tasks: 1) introduction to robust, yet familiar structures, 2) designing a model, 3) scale up from model to real structure- preparation, 4) scale up from model to real structure- design, and 5) design with different material.

Study 2: Dyads of students worked to complete an engineering design challenge. Students were asked to build a structure that could support a 0.5 lb. weight as high above a table as possible. The students were given basic household materials: one paper plate, four straws, five wooden sticks, and garden wire. To help facilitate the learning experience, students completed the following tasks: 1) introduction to robust, yet familiar structures, 2) reflecting on principles, 3) idea generation about engineering principles and mechanisms, 4) collaborative designing and building, and 5) student reflection.

Data sources and analysis

Study 1: For the scope of this paper we focused on data collected a month after the activity was over. We used the task as a transfer-task to determine if students were able to transfer insights they learned to a new situation. Five students were given an individual task; they were asked to design a cup holder that will hold a cup filled with hot chocolate above the table (no portion of the cup was allowed to be in contact with the table). Students were given an endless supply of toothpicks and gumdrops, they also had unlimited time to complete the activity. They were asked to “make their thinking visible” by chatting and explaining while constructing. Video data was used to record students comments and actions. Furthermore we videotaped an “expert” (a graduated mechanical engineer from a university in California) performing the same task. The idea was to reveal the engineer’s design process, and compare it with the novice engineers’ design process. We refer to the expert design strategy as an ideal way to complete this type of assignment. Final artifacts from both students and expert were collected as well. Findings were analyzed qualitatively in two levels: 1. Design process-based analysis, 2. Final-artifacts analysis.

1. Design process-based analysis: The video recordings of the design process of both expert and students were transcribed. We identified design activities explicitly showing similarities between expert and novices’ strategy (for instance: planning, measuring, constructing, etc.). Applicable engineering principles that novice students used when constructing the engineering design challenge were listed.
2. Final artifacts analysis: In order to assess expert and students’ final artifacts we developed a rubric. The initial version of the rubric was developed by open coding and further refined by looking at the artifacts, which were assessed by two researchers. It is important to note that the rubric was developed to evaluate the quality of the designed artifact. The rubric includes 4

main categories. Each category ranked from 0 as the lowest (entire cup on the table, not stable etc.) to 2 the highest. The highest grade an artifact could get was 8 (table 1).

Table 1: Artifacts' assessment rubric

	Category	***Score***
1.	*Functionality #1.* Examine if it is capable to do what it meant to do: being above the table.	*Cup above the table.* Entire cup on table (0), a portion of the cup contacts with the table (1), cup not touching the table (2).
2.	*Functionality #2.* Examine if it is capable to do what it meant to do: hold the weight of chocolate milk.	*Cup is stable with chocolate milk.* Immediately falls (0), stays stable for at least 10 seconds (1), stays stable for 1 min or more. (2).
3.	*Originality and creativity.* Examines whether the model represents original idea of a cup holder or does it look similar to a traditional holder.	Not original (0), some ideas are new (1), creative; does not look like any cup holder (2).
4.	*The use of engineer principles*. Examine the consumption of engineer's principles in the final artifact.	No principles used (0), one principle is shown (1), two or more principles, are used (symmetry, use of triangles, the foundation are wider etc.) (2).

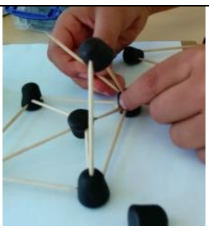
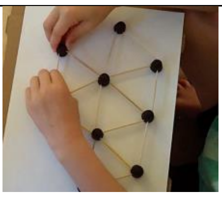
Study 2: Data for study 2 includes pre- and post-test measures along with a process-based artifact analysis. The pre- and post-test measured the breadth of applicable engineering principles that students consider when troubleshooting a structural engineering design challenge. The process-based analysis of student artifacts followed the approach of Category 3 in study 1, in which we closely examined how students incorporate engineering principles into the design of their structures. However, we are expanding on the approach in Study 1, by chronicling the addition of principle-inspired modifications to a given design. Making this determination was based on qualitative coding of the current state of a student's design any time they tested their design for structural stability. The specific engineering principles that we looked for include (1) adding a base, (2) adding reinforcement, (3) adding triangles, (4) making strong connections and (5) adding symmetry.

Results

In this section we describe and discuss the findings. In both studies we demonstrate students' approach to design following the interventions and emphasize on similarities between expert and novice reasoning strategy.

Study 1: Following are two types of findings: 1. Students' principle-based reasoning throughout the design process 2. Final artifacts and the similarities between expert and novices.
1. Results extracted from the expert's design process show a strong propensity to design based on principles. Interestingly novice students operated in a similar way and constructed their cup holder based on principles. The following table (table 2) includes a few examples of students' principle-based reasoning.

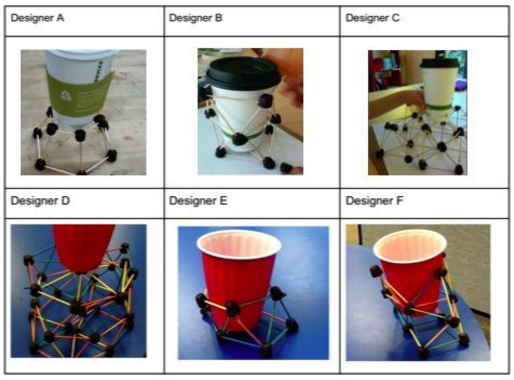
Table 2: Novice principle-based reasoning examples

Principles	Context and Citation	Screenshot example
Triangle is stronger than a square: adding triangles	C. started to design a square and transformed it to a triangle. He explained while moving the triangle to show me how strong it is. *"If it's a square its wobbly and if it's a triangle it is stronger"*	
Adding symmetry	M. built another layer, on top of the foundation to make it higher. In his words: "the same in each side". He wanted it to be symmetrical and said that if it will not be symmetrical it will not be stable. *"It has to be the same at each side...if it's different it is going to be a little less stable...".*	
Adding a base, and adding reinforcement: The base holds everything together. It is the most important thing.	A. started with a triangle shape that turn into a platform made out of diamonds. He said that he is working on the base and that the foundation must be stable. *"This is the most important part of the whole thing basically", "It hold the whole thing together".*	

In summary, results imply that novice students, 6 years old, are going through similar design process as an expert engineer while constructing a cup holder. It is indicated that they can use an akin design strategy as expert engineers and following these set of activities have a lot in common with an expert engineer in constructing this specific assignment. 5 out of 5 students showed a strong propensity to design based on principles. None of the students used material-based reasoning or example-based reasoning. Students referred to principles that were used by them while working in the class: adding triangle, using symmetrical shapes, wide base etc.

2. The final artifacts for five participants are shown in Table 3. Data implied that it is hard to distinguish the expert from the novice designs. In order to characterize the final artifacts we graded each product using the categories in the developed rubric (table 1). The final artifact ranks illustrate, that there is no clear distinction between expert and novice (figure 3). "Designer A", the expert designer (in red), was ranked high in three out of four categories: in both functionality categories (cup above the table, cup stability) he received a high grade. In addition he used more than two engineering principles, which were shown in his final artifact (Symmetry and the use of a wide foundation etc.). However, he did not receive any grade in originality of the product. His model did not represent original idea of a cup holder and looks similar to a traditional holder. 3 out of 5 students got high grade in functionality (both categories) and 2 out of 5 (C, D) got the highest grade in the originality. It was interesting to see that all students were using the geodesic dome principles as a reference to their final design.

Table 3: Expert and students final artifact (designer A is the expert)



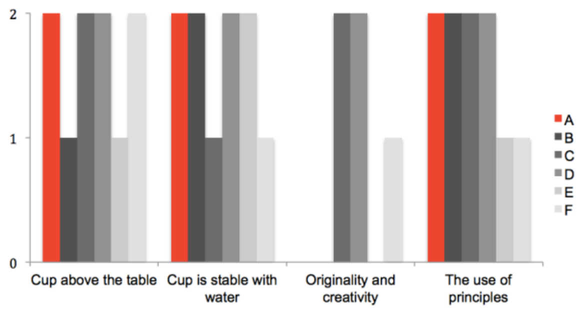


Figure 2: Final artifacts' grades according to the rubric

Study 2: We paid particular attention to how students made principle-based modifications to their designs. Table 4 walks the reader through the principles added to a given pairs' design over the course of the design challenge.

Table 4. Intermediate designs and description of principle-based modifications

Principles	Context and Citation	Screenshot example
Triangle is stronger than a square Symmetry	The students begin using triangles, strong connections and symmetry.	
Add reinforcement	Students add reinforcement with a popsicle sticks and re-add plate.	

This pair of students used four principles across five design iterations (2 of which are pictured for the reader). These modifications appear within the first and fourth step in their design process and include the addition of triangles, symmetry, a strong base and reinforcement. Hence, these students were able to take principles that they considered during the intervention task and incorporate them into the hands-on activity that they completed.

As we consider the relative importance of principle-based additions more broadly, we examined the relationships between student post-test scores and the rate that they made a principle-based addition to their design. More specifically, the rate of principle-based additions is computed by taking the total number of principle-based modifications, divided by the number of times the students testing their structure. From the figure (figure 3), we see a strong correlation between the rate of principle-based additions and student post-test scores. In particular, principle-based item inclusion, significantly predicted a pair of students' combined post-test score ($\beta=0.21$, $t(4)=6.54$, $p < 0.01$). Principle-based item inclusion also explained a large portion of the variance in learning scores ($R^2=0.93$, $F(1,3)=42.83$, $p<0.01$).

Discussion and Conclusions

Results from those two independent studies suggested that novice could effectively make use of principle-based reasoning. In the first study, first grade students used principles both in the design process and their final artifact. Principles were drawn from their class experience and were strongly impacted by the geodesic dome they had previously constructed. In the second study, we showed that high school students were able to effectively use principles, even without receiving any formal instruction about those principles. Furthermore, we saw a clear correlation between the ways that students incorporated principles into their design, and their scores on the post-test. These two studies could be used to suggest that non-experts can effectively make use of certain expert strategies. Despite the fact that expertise is typically defined as requiring both expert knowledge and expert organization of that knowledge (e.g. Bedard & Chi, 1992) and even though it is unreasonable to expect non-experts to behave like experts, it seems that the specific instructional sequences, better prepared students to reason based on principles. The instructional sequence, in both studies, was designed with common elements and was transferred to the students in a similar order, it includes: 1. Concrete examples: which were familiar and relevant to students (a house for the first grade students, and bridge, ladder for the high school students). 2. Identification/instantiation of principles: In the first study, students designed a model of a house for a Lego man; while in the second study they made a sketch of the design using principles. 3. Hands-on, designing and building in collaboration (design a scale up geodesic dome, design a structure). We believe that the instructional sequence that was designed in both studies had similar common elements, which were crucial to effectively making use of principle-based reasoning by novice students. For example, we would argue that the hands-on experience, or physicality component was one of those essential elements. By physicality, we mean the actual and active design with concrete and common materials (Zacharia et al., 2012). This type of learning is consistent with the concrete-to-abstract nature of cognitive development, it is also based on the idea that cognitive processing of information coming from multiple modalities (i.e., a lecture, drawings, and hands-on) forms a stronger representation for a learner than information coming from only one of these modalities (Han & Black, 2011). Even though the students in these studies did not have expert-level knowledge or a deep understanding of the scientific theories involved, they were able to generate principles based on conceptual intuitions and by using everyday language. We also think that, in the particular case of physical constructions and design, powerful forms of expertise can be translated into elementary principles (such as using triangles, symmetry etc.) that are easy to learn. We would also like to note that the proposed model was instituted over a relatively short time frame. Hence supporting effective use of principle-based reasoning simply requires the use of a three step instructional sequence that could be easily adapted and be tailored to different populations of learners. Finally, perhaps because many of the students had not yet been introduced to mathematical formulas, there was less of a propensity for them to fall back on formulas.

In presenting the findings from the two studies reported in this paper, we have endeavored to contribute to a body of literature that will help advance engineering education and human cognition. Although principle-based scientific reasoning is an essential skill for scientific literacy, it is rarely learned (Treagust et al. 2002, Wilson et al. 2006) and future research will be needed to make it teachable.

References

Anderson, J. R., Greeno, J. G., Kline, P. J., & Neves, D. M. (1981). Acquisition of problem-solving skill. Cognitive skills and their acquisition, 191-230.

Bédard, J., & Chi, M. T. (1992). Expertise. Current directions in psychological science, 135-139.

Chi, M. T. H., Glaser, R., & Rees, E. (1982). Expertise in problem solving. IN R. Sternberg (Ed.), Advances in Psychology of Human Intelligence 1: 7-76. Hillsdale, NJ: Erlbaum.

Chi, M. T. H.,& VanLehn, K. (2012). Seeing deep structure from the interactions of surface features.

Chue, S., & Lee, Y.-J. (2013). The Proof of the Pudding?: A Case Study of an “At-Risk” Design-Based Inquiry Science Curriculum. Research in Science Education, 43(6), 2431–2454. doi:10.1007/s11165-013-9366-x

Han, I., & Black, J. B. (2011). Incorporating haptic feedback in simulation for learning physics. Computers & Education, 57(4), 2281-2290.

Hammer, D. (2004b). The variability of student reasoning, lecture 2: Transitions. In Proceedings International School of Physics Enrico Fermi (Vol. 156, pp. 301-320). IOS Press; Ohmsha; 1999.

Moss, J., Kotovsky, K., & Cagan, J. (2006). The role of functionality in the mental representations of engineering students: Some differences in the early stages of expertise. Cognitive Science, 30, 65–93.

NGSS Lead States. 2013. Next Generation Science Standards: For states, by states. Washington, DC: National Academies Press.

Nokes T. J., Schunn C. D. and Chi M. T. H. (2010), Problem solving and human expertise. In: Penelope Peterson, Eva Baker, Barry McGaw, (Editors), International Encyclopedia of Education. volume 5, pp. 265-272. Oxford: Elsevier.

Treagust, D. F., Chittleborough, G., Mamiala, T. (2002), Students' understanding of the role of scientific models in learning science. International Journal of Science Education 24: 24–357.

Turkle, S., & Papert, S. (1992). Epistemological Pluralism and the Revaluation of the Concrete. Journal of Mathematical Behavior, 11(1), 1–30.

VanLehn, K. (1996). Cognitive skill acquisition. Annual Review of Psychology, 47, 513–39. doi:10.1146/annurev.psych.47.1.513

Wilson, C. D., Anderson, C. W., Heidemann, M., Merrill, J. E., Merritt, B. W., Richmond, G., et al. (2006). Assessing students' ability to trace matter in dynamic systems in cell biology. CBE Life Sciences Eduction, 5(4), 323–31. doi:10.1187/cbe.06-02-0142

Zacharia, Z. C., Loizou, E., & Papaevripidou, M. (2012). Is physicality an important aspect of learning through science experimentation among kindergarten students?. Early Childhood Research Quarterly, 27(3), 447-457.

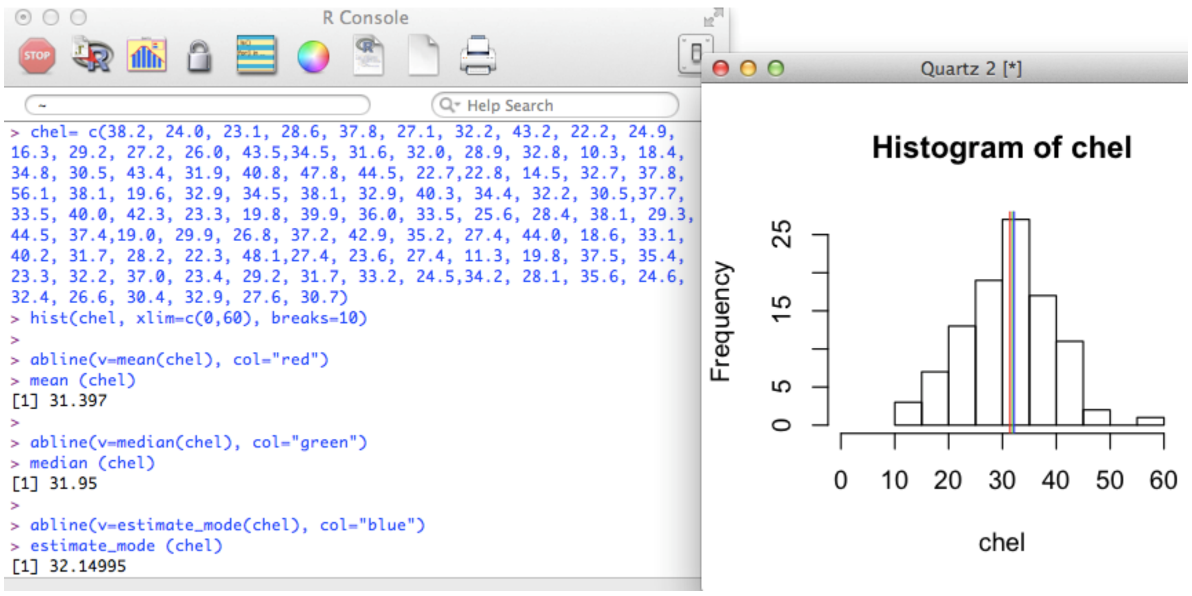
Exploring randomness and variability in statistics through R based programming tasks

Maite Mascaró, *mmm@ciencias.unam.mx*
UMDI-Sisal, National Autonomous University of Mexico (UNAM)

Ana Isabel Sacristán, *asacrist@cinvestav.mx*
Dept of Mathematics Education, Centre for Research and Advanced Studies (Cinvestav-IPN)

Abstract

In this paper we present a computational programming activity using R code (http://www.r-project.org/) –see Fig. 1–, for exploring randomness and variability in statistics, as carried out by two pairs of 2nd-year university environmental sciences students in Mexico. This activity is part of a larger set of over 30 R-based constructionist and collaborative programming activities, developed over the last 5 years, for the teaching and learning of experimental analysis and statistics, which we presented at the 2014 Constructionism conference (see Mascaró, Sacristán & Rufino, 2014), and that have so far been implemented in 13 university courses in three institutions in Mexico and Portugal, with successful results.



```
> chel= c(38.2, 24.0, 23.1, 28.6, 37.8, 27.1, 32.2, 43.2, 22.2, 24.9,
16.3, 29.2, 27.2, 26.0, 43.5,34.5, 31.6, 32.0, 28.9, 32.8, 10.3, 18.4,
34.8, 30.5, 43.4, 31.9, 40.8, 47.8, 44.5, 22.7,22.8, 14.5, 32.7, 37.8,
56.1, 38.1, 19.6, 32.9, 34.5, 38.1, 32.9, 40.3, 34.4, 32.2, 30.5,37.7,
33.5, 40.0, 42.3, 23.3, 19.8, 39.9, 36.0, 33.5, 25.6, 28.4, 38.1, 29.3,
44.5, 37.4,19.0, 29.9, 26.8, 37.2, 42.9, 35.2, 27.4, 44.0, 18.6, 33.1,
40.2, 31.7, 28.2, 22.3, 48.1,27.4, 23.6, 27.4, 11.3, 19.8, 37.5, 35.4,
23.3, 32.2, 37.0, 23.4, 29.2, 31.7, 33.2, 24.5,34.2, 28.1, 35.6, 24.6,
32.4, 26.6, 30.4, 32.9, 27.6, 30.7)
> hist(chel, xlim=c(0,60), breaks=10)
>
> abline(v=mean(chel), col="red")
> mean (chel)
[1] 31.397
>
> abline(v=median(chel), col="green")
> median (chel)
[1] 31.95
>
> abline(v=estimate_mode(chel), col="blue")
> estimate_mode (chel)
[1] 32.14995
```

Figure 1. Using R to define a data vector 'chel' for the lengths of a sample of a hundred seahorses, representing the data by generating a histogram and locating on the graph the mean, median and mode.

Through these activities, the students carried out explorations in R that helped develop meanings for the concepts of randomness and variability, as well as introducing them to statistical functions, tools and representations, in a "functional" way that they can eventually appropriate for their future research activities.

Keywords

Statistics education, randomness, variability, programming, R code, constructionism, case studies

Introduction

As explained in Mascaró et al. (2014), our concern is that environmental scientists require probability and statistical knowledge for experimental data analysis, decision-making, evidence to support theory, and communication of scientific results, so it is important for future researchers to have adequate understanding of these topics. However, learners in university programs in biology and related sciences tend to have difficulties in understanding and/or applying the concepts, and even researchers have a poor understanding and use incorrectly many statistics concepts (Batanero, 2001). Many teaching programs have had unsuccessful results (Bishop & Talbot, 2001) and there tends to be an aversion (including general 'mathophobia' – Papert, 1980) towards learning statistics (Mascaró et al., 2014).

In order to overcome these difficulties, through an iterative design research (Cobb et al, 2003), we developed a series of meaningful computer programming activities, using as framework the constructionist philosophy (Papert & Harel, 1991), which suggests that learning can be facilitated if students engage in exploring ideas and concepts through construction, e.g. programming. We also incorporated the following recommendations drawn from research in statistical education: (i) Contextualising concepts using concrete examples with data from real research situations; (ii) Emphasising the use of diagrams, since graphic representations are essential in data organisation, statistical reasoning and analysis (Wild & Pfannkuch, 1999), and on changing from one representational register to another. (iii) Shifting the role of students from passive to active, with emphasis on reasoning, rather than an algorithmical application of statistical tests.

In our approach, as explained in Mascaró et al. (2014), all of our activities for teaching statistical reasoning are to be carried out using the R programming language; and are even presented through R-code "worksheets" with instructions, examples, programming tasks, questions for reflection, and comments. We chose R, because it is a transparent and powerful expression "Logo-like" language commonly used by statisticians, but simple to use (using a set of intuitive and relatively simple commands): R uses a consistent syntax with inputs to functions written between parentheses, in the common mathematical function style; it includes primitive functions and libraries of pre-defined commands**.** New objects in R can be variables, arrays of numbers, character strings, functions, or more general structures built from such components**.** Values can be assigned to objects by using the **'<-'** operator or, alternatively, the **'='** operator**.** Using the R console, one can directly run commands, or scripts **(**programs**)** created in an editor.

In accordance with the constructionist philosophy, we believe that the understanding of statistical models is facilitated by creating objects in R, to represent them. Furthermore, students will develop familiarity with R and its libraries. Thus, the benefits of using R programming are two-fold: 1.- Through R-programming tasks, students can apply and explore statistical functions and concepts, and develop meanings for them. 2.- Students can develop competency in a statistical analysis tool that they can use in their future research activities. The activities are designed to be carried out in teams of 2-3 students and moderated by an instructor. They are of two types: those for introducing the basics of R functioning and use; and those on different topics of statistics courses (e.g. activities that deal with frequency distributions; activities on binomial, Poisson, and normal distributions; activities on ANOVA and comparisons between means; activities on linear regression; etc.). In this paper, we focus on an activity for exploring randomness and variability.

The activity for exploring variability

This activity is the second one during the undergraduate course **"**Probability and Statistics**"** and is called Activity 2: Frequency Distributions **(**Continuous Variables**)**. Students are given the following context of the problem, a set with the total lengths of a hundred seahorses and then asked to carry out the sequence of tasks further below.

> In a study investigating the morphometrics of seahorses, a random sample of *Hippocampus erectus* was collected from Chelem, Yucatán, a tropical coastal lagoon with submerged vegetation in different densities, and that is subject to artisanal fisheries. For each of the individual seahorses collected with

a dragnet, the total length was recorded (TL mm)

1.- Save the results of the lengths as a vector in R, naming it *'chel'*. For this, they would define the vector *chel* by typing chel = c*(<values of the sample>*) in the R console (as in Fig. 1). They could also type the same replacing the equal sign '=' with an arrow: chel <- c(…)

2.- Analyse, without any arithmetical operations on the data, the vector *chel* corresponding to the Chelem sample and describe observations on the total length (TL) of the seahorses.

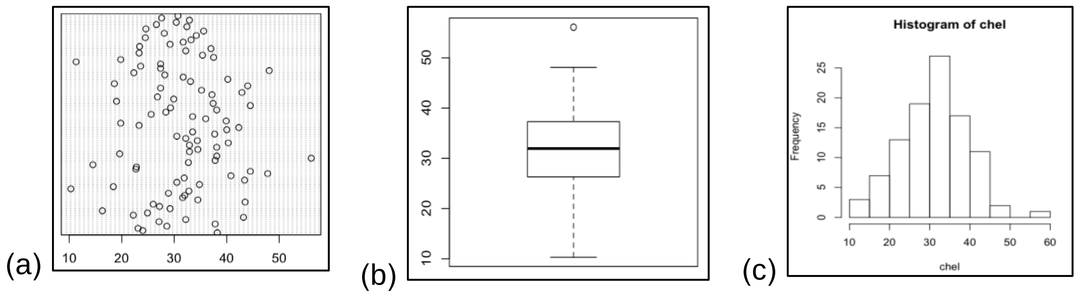


Figure 2. Representing the data contained in the vector 'chel' by generating graphs through the following R commands: for (a), dotchart(chel)*; for (b),* boxplot(chel)*; for (c),* hist(chel)*.*

3.- Create plots and graphs of the Chelem data, by typing commands such as dotchart(chel), boxplot(chel), hist(chel) –with which they had become familiar in previous activities– and observe and compare the graphs (see Fig. 2). (As explained by Zuur et al., 2007, dotcharts are a means to identify extreme values for the response and explanatory variables in one-dimensional data.) A hint is given that for understanding the graphical elements represented in the graphs, they can use the help documentation (e.g. typing ?boxplot.stats for the box-and-whisker plot.) Then, they are asked to describe again the TL by taking into account the values on the axes of the dots, bars and boxes in each graph. Finally, they are asked to indicate which of the characteristics described before could be identified in the graphs; and select the graph that serves best to represent the data from the Chelem sample, by discussing with their partners.

4.- Use their previous knowledge on central tendency and dispersion parameters, in order to explore the statistical nature of the Chelem sample. (It is suggested that they can use arithmetic operators or specific functions, such as: 'length', 'sum', 'var', 'max', 'sqrt', or search the help documentation for other functions). They are then asked to explain which of the parameters they would choose to describe the sample numerically.

5. Build a new histogram of the Chelem sample using the following commands:

```
hist(chel, xlim=c(0,60), breaks=10); abline(v=mean(chel), col="red")
```

This produces a histogram with a vertical red line (representing the mean value), which students need to interpret, and then build different coloured lines over the value of the median and mode of the distribution. They are asked if they could represent the range and standard deviation of the sample by using similar graphic procedures; if they could calculate the size of the sample by analysing the graph; and to discuss this with their partners. (For the latter it is worth noting that if they generate lines using 'abline' with the values of the range or standard deviation –as some students did, as shown below–, such lines would *not* represent these parameters on the graph, since they quantify differences or distance between points –therefore involving two numbers– and are meaningless if located at a specific x-value.)

6. Students are told that the data for the vector representing the Chelem sample that was used in the first part of the activity, was generated using the following command:

```
popul = round(rnorm(10000, 30, 8),1)); sample(popul, 100, replace=T)
```

(This command line will take a random sample of 100 elements from a randomly generated sequence of the normal distribution of 10000 observations (called 'popul'), with mean equal to 30 and standard deviation equal to 8, rounded to 1 decimal point). They then are asked how they could verify that the sample was randomly collected, and to show which argument can be used to change the size of the sample.

7. Explore the relationship between the size of the sample and the parameters used to statistically describe the sample, pick some parameters and predict how their value will change depending on whether the sample gets bigger or smaller, and then verifying their predictions –by changing the size arguments in the code line above: `sample(popul,…` ,computing the parameters that they think appropriate, and filling out a table with the information.

8. Students are introduced to the 'plot' function, which is a generic function for plotting, and have to build a graph to show how chosen parameters change with the size of the sample.

9. Given the following two formulas:

```
sum((chel-mean(chel))^2)/length(chel)
sum((chel-mean(chel))^2)/(length(chel)-1)
```

test and analyse these in order to explain the difference between them; if they represent the same; and which one would they choose for the Chelem sample. The aim is for students to reflect on the difference between the population variance (σ^2: where N is the population size) and the sample variance (s^2: where n is the size of the sample). By definition, the population variance (divided by N) will always be less than the sample variance (divided by n-1), even in the unlikely case where the sample equals the complete population (N=n). This is relevant to understand the relation of size and quality of the information, with randomness, and independence, as traits necessary in the definition of a sample in this context.

10. Calculate the difference in magnitude of the variance using the previous formulas and describe what will happen to that difference as the size of the sample decreases/increases. (Students are reminded that it is assumed that the population is known and that σ^2 has a constant value). The purpose of this activity is to develop the concept of randomness and relate it to variability, and to be able to understand that when a process is random, values change in a manner that is not predictable. We want students to understand that randomness is part of natural phenomena, and can be an ally when it is adequately treated as "background noise". We also want students to develop meanings for the concepts of mean, it´s precision and accuracy, and to understand the degree to which sample size can be controlled in order to improve the estimation of population parameters.

Simultaneously, the activity helps us to introduce students to the statistical tools and commands found in R –and through them to the statistical concepts, as such–, analyse them and apply them in "functional" manner in the programming tasks. This should help give them both an understanding of the concepts, as well as some knowledge of the tool (the R programming language) that can be very useful for them in their future research activities.

Some results from case studies

Although we have implemented similar activities in previous courses in whole-classroom situations, we wanted to be able to observe more precisely how students carry them out, in order to document –according to the ideas presented by Noss and Hoyles (1996)– their thinking-in-change, and their understandings and difficulties, using these tasks as windows into their developing meanings for statistical concepts (as computational statistical commands, in their application within the situation of the tasks, and as more abstract concepts). Therefore, we tried out the previous activity, over the course of two sessions, in case studies with some 2nd-year students, studying for the degree of Sustainable Coastal Zone Management at the National Autonomous University of Mexico, and enrolled in the semester-long introductory course "Probability and Statistics". Below, we present some results on the case studies of two teams of students: Alexis and Hiram, and Isabel and Marcos. These students had never studied statistics before university, though they had recently taken a very brief (28 hour) introduction to frequency distributions and their descriptors, during which they used R for the first time. From previous classes we detected that Alexis and Hiram were above average students, whereas Isabel and Marcos were average students.

Task 2: This task asks students to give observations on the total length (TL) of the seahorses without arithmetic operations on the data. Alexis and Hiram read the problem out loud, silently analysed the data given in the vector *chel*, then discussed using some R functions for ordering the vector and for finding the maximum and minimum values more easily. Thus, they used the following R commands for this analysis: min(chel) and max(chel) in order to find, respectively, the minimum and maximum lengths recorded in the sample. They then estimated the number of elements in the sample, by approximation, by estimating how many elements were in each line, then confirming this by looking at the display of the vector, where the number of the first item in each line is given in square brackets. This was re-confirmed by typing the command length(chel). They then wrote in their worksheets that “the smallest seahorse measures 10.3mm”; “the largest seahorse measures 56.1mm”; “the total size of the sample is of 100 individuals”; and “the mean is 33.2mm, which was obtained by mentally adding the minimum and the maximum values and dividing the result by two”. It is interesting to see that they consider the mean and median as central tendency values, but it is not clear whether they realise that the arithmetic mean and the median only coincide in certain specific distributions (e.g. symmetrical or uniform ones), but not all.

Isabel and Marcos, at first didn’t know what to do with the information, with Marcos not understanding what each element of the vector represented and asking what TL (total length of the seahorses) meant. Then they only did a visual analysis of the sample, writing that the sample showed “seahorses with a minimum TL of 10.3mm and a maximum of 56.1mm, with more seahorses with a length 30-40mm”. Without calculations and other statistical tools, these values only give minimal information, but at this stage students may not be aware of this.

Task 3: This is the task where students were asked to build different plots of the data (Fig. 2), analyse them, and determine which they considered gave a better representation of the data. Alexis and Hiram focused on the boxplot. After using the hint given to help them understand what that the box-and-whiskers represents, they wrote this description:

> The first whisker shows the minimum value of the sample; the line from the first whisker to the box groups 25% of the data; the box groups 50% of the data; and the central line represents the median. The line from the end of the box (third quartile) at the end of the graph, groups another 25% of the data. There is a point outside the diagram; this is a case of an outlier (a data point with an extraordinary value) that doesn’t follow the characteristics for belonging to the graph.

With the latter comment there is no definition as to what it means “not to belong to the graph”. Therein lies precisely the central aspect of the concept of variability: how different must a value be from the rest, in order to be able to say that it doesn’t belong to the same group? Is it just the size of the difference that one should take into account, or is also the relative frequency of occurrence of such value? Related to that, what information do the histogram and boxplot present that the dotchart doesn’t?

Alexis felt that what was being calculated with the boxplot, was the variance and tried to remember the formula and tried to write what he remembered on a paper to show to his partner and seek his help. Because they were unsure they tried calculating the variation coefficient to see if it coincided with what was represented in the boxplot, and Alexis realised that it didn’t. Alexis and Hiram finally explain that the graphs represent “the dispersion of the data, where the data concentrate, the extremes, and with the boxplot the outlier”. Hiram then felt that the dotchart with “the dispersion of points explained better the distribution [of the lengths] of the seahorses” and showed all 100 points, which allowed him to understand better what the graph represented. Alexis said he preferred the boxplot to the dotchart, which was less abstract; and to the histogram of frequencies, because it helped “identify immediately any [outlier]”. Unfortunately, the answers are descriptive rather than having a deeper basis for their decision.

Isabel and Marcos, on their part, began by exploring the dotchart and then comparing the different graphs in a same window (using an R script from a previous lesson) and discussing amongst themselves. They then described correctly and thoroughly the graphs:

> In each graph one can observe a bigger grouping in the 30-40mm interval. The clearest graph for observing such grouping is the frequency histogram where the biggest frequency is 30-35mm. In the box-and-whisker plot, one can see an outlier that exceeds 50mm. Likewise, in the dotchart one can see an accumulation of data in the centre, between the 30-40 interval, with few infrequent values in the extremes (less than 20 and larger than 50).

However, they didn't choose one, possibly because they couldn't yet identify the important traits that differentiate the graphs. One such trait is the variability, but they couldn't yet identify which format best represented that variability.

Task 4: This task asked which functions could help describe numerically the Chelem sample. Alexis and Hiram tried out length(chel); mean(chel); var(chel) (using 'help(var)' for understanding the variance command); the standard deviation sd(chel); the coefficient of variation (sd(chel)/mean(chel)); the median(chel). They said that the ones that best represented the variability would be "the mean, the median, the standard deviation, the size of the sample and the variance" but did not explain why. Although some of these are redundant, they seem to understand them and how to calculate them, although they don't seem to be able yet to identify the advantages or disadvantages in terms of the information that each provides.

Isabel and Marcos, on their part, answered that they would "use 'length' to see the size of the sample, 'max' to know the maximum value in the sample, 'sum' divided by 'length' to get the mean, 'var' to get the variance, and 'sqrt(var(chel))' to get the standard. deviation". Although they focus more on the operative aspects, instead that on the information provided by each function, it is interesting that they define ways to calculate the mean and standard deviation using other functions, rather than relying on the direct functions for each; the latter shows at least some knowledge of the definitions of those two concepts, which they use in the R code.

Task 5: In this task, both teams of students correctly identified the red line in the histogram (Figs. 1 & 4) produced by the given commands, as representing the mean (as "*media*" in Spanish, even though the R commands are in English). Isabel and Marcos then added the median as a blue line (Fig. 4), by typing abline(v=median(chel), col="blue") –none drew the mode line, since there is no direct command for the mode. They then typed the following, without reflection:

abline(v=range(chel), col="yellow")
abline(v=sqrt(var(chel)), col="green")

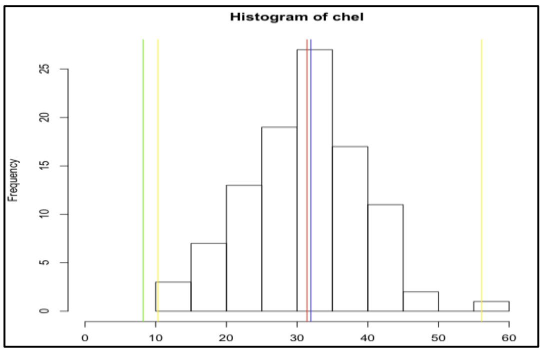


Figure 4. Isabel and Marcos histogram in Task 5, with meaningless yellow and green lines using the values of the range and standard deviation.

They did not realise that these lines (the yellow and the green) are meaningless in terms of the graph since, as explained above, the range and standard deviation quantify differences or distance between points, thus are meaningless if located at a specific x-value. Isabel and Marcos also said that "you can't" find out the size of the sample from the graph. This indicates that they haven't yet understood what the bars in a histogram represent.

Alexis and Hiram, on their part, rather than explaining how they could identify the range and standard deviation from the graph, simply said that they would "subtract the largest value with the smallest one and the S.D. [standard deviation]; although their language is not completely appropriate, at least they are referring to the difference or distance between the highest value and the lowest, which is the definition of range. They then answer that the size of the sample can be calculated from the graph, because "the frequency is [given] in clear units". In order to clarify the nature of a continuous variable and randomness for generating values, the instructor (first author of this paper) asked them with respect to the frequency in which a continuous variable occurs in a sample (such as the total length variable of seahorses), and if the exact value would repeat itself. Hiram just shook his head, and Alexis explained that it wasn't necessary for the exact value to repeat itself or knowing its frequency, because that knowledge is already in the histogram. Related to the graphical representation of variability, they felt that range was easier to represent than standard deviation. When questioned how to represent either one, Hiram opened his arms indicating a distance. If it is a distance, how would it be represented with a single value? They cannot answer; Alexis only mentions that it will be done "using the size of the bars in the histogram" and they look this up in the internet. Finally they arrive at the conclusion that "it is meaningless; [range and standard deviation] are of a second dimension [*sic*] and cannot be represented graphically in the histogram"

Tasks 6 & 7: Here, students need to reflect on how the sample is produced randomly, what changes the size of the sample, and the relationship between the size of the sample and the arguments used. At first, Alexis and Hiram answer by simply analysing (rather than trying out) the code and what the 'sample' command does: "output a sequence of random data" (Hiram) with an argument for having 100 elements (Alexis). After a long time, Hiram finally suggests experimenting by trying out different values for the arguments, confirming that by changing 100 with 150 in sample(popul, 150, replace=T) the size of the sample increases to 150. In later lessons, this student would autonomously generate samples using the 'rnorm' function.

Likewise, Isabel and Marcos write that if the argument is changed, the size of the sample changes. Isabel and Marcos focus on the mean argument 30; by generating different samples they find that these have a mean close to 30, but not exactly the same. Also, by experimenting Marcos concludes that "the difference between max. and min. is the range" of the sample.

For Task 7, Alexis and Hiram identify that in the given formula, the population mean is μ=30 and the value of the size of the sample is n=100, have good understanding of these concepts and play with the code easily, using modularity. They explore again sample(popul, 100, replace=T), changing the value of 100 repeatedly up and down (e.g. to 900, 20, 9000, etc.) and in that way create different samples (sample1, sample2, etc.) for which they compute the mean (i.e. mean(sample1), etc.) and standard deviation (i.e. sd(sample1), etc.). Hiram thinks that "the parameters get closer to the values that they should be, as n gets bigger" but Alexis doesn't yet quite understand until they test them with very small and big sample sizes (as above). They are struck when n=2 and get a sample with mean of 29.7, and think they made a mistake, that it can't be possible to get such a precise value with such small n! Hiram, disconcerted, says "the smaller the n, the farther it should be from the population value…" Prompted by the teacher, they search for a single data point with value 30, to see the possibility of getting a "mean" of 30 with n=1, and to their surprise they find it. There is a Eureka moment with Hiram saying: "Ah! It is possible… we can have mean values of 30 with a really small n… because it is random! It may have little probability, but it *can* happen. But it is true that the bigger the sample, the closer". The teacher reaffirms this, but emphasizes the randomness, referring to it as "background noise".

Isabel and Marcos, on their part, concluded that "As the size of the sample increases, the range increases, and as [it] decreases, the mean deviates more from the true mean (30)" after a very long exploration. Because they had difficulties visualising the pattern and graphing in R, at the teacher's suggestion they built a table and a paper-and-pencil graph, which helped them.

Tasks 8-10: Due to time constraints neither team explored tasks 8 and 10, and only Alexis and Hiram began answering Task 9, saying that sum((chel-mean(chel))^2)/length(chel) referred to the variance of the population, and sum((chel-mean(chel))^2)/(length(chel)-1) to the variance of the sample, but they didn't really test it.

Final remarks

The constructionist approach taken, where students are active in using and expressing statistical ideas and concepts through the R programming tasks, includes several important elements: among those, the expressive tasks and generation of multiple representations and application of statistical concepts through formal descriptions (in the code), is a very important one. As diSessa (2001) explains, programming can turn analysis into experience and allow "a connection between analytic forms and their experiential implications that algebra and even calculus can't touch" (p. 34). Furthermore, programming is engaging, as was the case with these students, helping them become more familiar with (and overcome some of their fears of) both statistical concepts and the computer programming. The collaborative work also served as another means of expression and clarification of the concepts under study. The students may not have fully explored in each task what we had set to achieve, and may not have yet developed proficiency fully in the construction of graphs with R (with Isabel and Marcos, in particular, having difficulty in this regard), but they did seem to learn how much a graph synthesises information, and the enormous importance to be able to construct graphs quickly for understanding the data. More importantly, they achieved our purpose of exploring and developing some understanding of the concepts of randomness and variability, and will continue with a dozen more such activities to develop more meaningful knowledge of the statistical concepts and also R.

Acknowledgements

This work was financed in part by the PAPIME PE204614 grant from DGAPA-UNAM. We also wish to thank our colleague Marta Rufino, of the Portuguese Institute for the Sea and Atmosphere (IPMA), who is the co-author of the activities we have been designing and trying out in the past 5 years; as well as the four students discussed in this paper.

References

Batanero, C. (2001) Main research problems in the training of researchers. *Training Researchers in the Use of Statistics* (C. Batanero ed.). Granada, Spain: International Association for Statistical Education and International Statistical Institute, pp. 385-396.

Bishop, G. & Talbot, M. (2001). Statistical thinking for novice researchers in the biological sciences. In C. Batanero (Ed.), *Training Researchers in the Use of Statistics* (pp. 215-226). Granada, Spain: Intl. Association for Statistical Education.

Cobb, P., Confrey, J., diSessa, A., Lehrer, R., & Schauble, L. (2003). Design experiments in education research. *Educational Researcher*, 32(1), 9–13

diSessa, A. (2001). *Changing minds: computers, learning and literacy*. Cambridge: MIT Press.

Mascaró, M., Sacristán, A. I. & Rufino M. (2014). Teaching and learning statistics and experimental analysis for environmental science students, through programming activities in R. In G. Futschek & C. Kynigos (Eds.), *Constructionism and Creativity - Proceedings 3rd Intl. Constructionism Conf. 2014* (pp. 407-416). Vienna, Austria: OCG.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: learning cultures and computers*. Dordrecht: Kluwer.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas.* New York: Basic Books.

Papert, S. & Harel, I. (1991). Situating Constructionism. In I. Harel & S. Papert (Eds.), *Constructionism.* Norwood., NJ: Ablex Publishing Corporation. Retrieved from http://www.papert.org/articles/SituatingConstructionism.htm

Wild, C. & Pfannkuch, M. (1999). Statistical thinking in empirical enquiry. *International Statistical Review,* **67** (3), 223-265.

Zuur, A. F., Leno, E. N., and Smith, G. M. (2007). *Analysing Ecological Data Series: Statistics for Biology and Health.* Springer, New York

Game Design with Pocket Code: Providing a Constructionist Environment for Girls in the School Context

Anja Petri, *anja.petri@ist.tugraz.at*
Institute of Software Technology, Graz University of Technology, Austria

Christian Schindler, *christian.schindler@ist.tugraz.at*
Institute of Software Technology, Graz University of Technology, Austria

Wolfgang Slany, *wolfgang.slany@tugraz.at*
Institute of Software Technology, Graz University of Technology, Austria

Bernadette Spieler, *bernadette.spieler@ist.tugraz.at*
Institute of Software Technology, Graz University of Technology, Austria

Abstract

The widespread use of mobile phones is changing how learning takes place in many disciplines and contexts. As a scenario in a constructionist learning environment, students are given powerful tools to create games using their own ideas. In the “No One Left Behind” (NOLB) project we will study through experimental cycles whether the use of mobile game design has an impact on learning, understanding, and retention of knowledge for students at risk of social exclusion. We will use the mobile learning app Pocket Code with partner schools in three countries: Austria, Spain, and the UK. This paper focuses on the Austrian pilot, which is exploring gender inclusion in game creation within an educational environment. We first study differences in game creation between girls and boys. This study that started in September 2015, will help teachers to integrate Pocket Code effectively into their courses. For future studies an enhanced school version of Pocket Code will be designed using the results and insights gathered at schools with pupils and teachers.

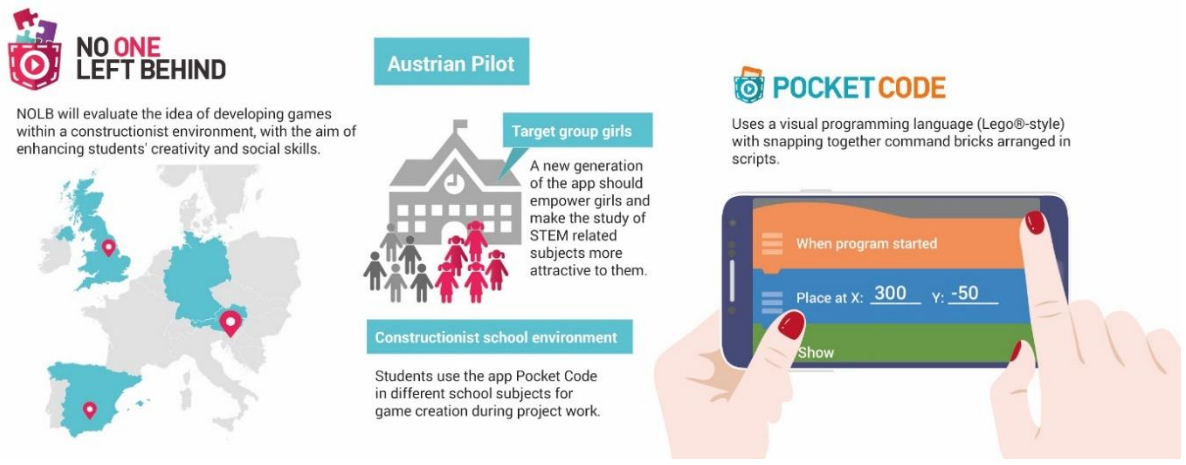

NO ONE LEFT BEHIND
NOLB will evaluate the idea of developing games within a constructionist environment, with the aim of enhancing students' creativity and social skills.
Austrian Pilot
Target group girls
A new generation of the app should empower girls and make the study of STEM related subjects more attractive to them.
Constructionist school environment
Students use the app Pocket Code in different school subjects for game creation during project work.
POCKET CODE
Uses a visual programming language (Lego®-style) with snapping together command bricks arranged in scripts.
When program started
Place at X: 300 Y: -50
Show

Figure 1: **The NOLB project, the Austrians pilot and Pocket Code**.

Keywords

Pocket Code, Social Game Experience, Game Design, Tools for Game Creation, Social Inclusion, Programming, Mobile Learning, Constructionism, Constructivism, Gender Inclusion, Girls

Introduction

Constructionism motivates learning through creating one’s own games (Parmaxi and Zaphiris, 2014) but requires appropriate hardware infrastructure, which is often outdated and/or insufficiently available in schools. Modern smartphones are increasingly owned by students all over the world and thus could help solve the hardware problem in and outside of schools. Playing mobile games is a popular leisure activity for young people (Schippers and Mak, 2015), but creating smartphone apps has until now been difficult. To ensure a constructionist learning environment, Parmaxi and Zaphiris recommended the use of appropriate mobile tools that can address the cognitive and social aspects of learning. Therefore we will support students creating their own mobile games in a visual, Lego®-style way to improve their learning experience at schools. With the mobile learning app Pocket Code students can easily program games directly on their smartphones, without requiring any additional hardware. Within the EU project “No One Left Behind" (NOLB), we will develop a constructionist approach that integrates Pocket Code into school curricula and that motivates students to become independent thinkers.

In previous work, the authors described reasons why girls and women are underrepresented in Science, Technology, Engineering, and Mathematics (STEM) related subjects (Beltrán *et al*., 2015). In addition, Weibert *et a*l. (2012) defined personal attachment as a determining factor to motivate girls to attend computer science subjects. Therefore, the challenge of a constructionist learning environment is to create situations in which girls feel comfortable expressing their own interests. Within NOLB, groups of girls and boys will solve curriculum related problems while using the Pocket Code app. When working on these projects, students can gather information from a variety of sources, analyse this information, engage with it intensely, and derive knowledge from the process of constructing a game (Chandrasekaran, 2012). This connection with real world problems enhances their learning and makes it more valuable. Thus game design methods utilise game and project based learning and foster collaboration for students at risk of social exclusion.

This paper starts by describing gender differences in game design, followed by background information about constructivism and constructionism. The next section provides details about the research methods used within the NOLB project, the Pocket Code app, the Austrian pilot, and goals of the first feasibility study. The next sections illustrates the integration of Pocket Code in the school contexts and describes workshops with students and teachers in Austria. The intent is to connect students’ interests with coding while at the same time fostering a constructionism approach through collaboration and engagement to make learning enjoyable. Clearly it is important to support students and teachers with material and to provide resources such as media assets and frameworks. Finally, we describe possible future directions for research.

Related work

In this section we describe studies about gender aspects in game play and design. We also present settings for a constructivist and constructionist school environment.

Supporting gender unbiased game design

In 1996, Greenfield and Cocking discovered that there are significant gender differences between girls and boys regarding performance, interests, and experience in game play. Kafai (2008) proposes to emphasize different content, mechanics, and characters in order to make games more appealing to girls. Craig *et al.* (2013) observed that the participation of girls in computer classes is not the same as those of boys: girls tend to spend more time on visual customization while boys spent more time on solving logical puzzles, and the authors thus point out that it is essential to consider gender differences in logical and computational skills. According to this study it may thus be more effective to get girls interested in technology by asking them to design games rather than to focus on the learning of specific programming skills.

Constructivism and constructionism

Piaget's constructivism provides a framework for optimizing the learning progress at different levels of children's development (Kafai and Resnick, 1996). Younger children create their own subjective reality, depending on their own experiences, which is suited to their current needs and possibilities. Children enhance their capability of abstract thinking and start to philosophize about probabilities, associations, and analogies by the age of 11.

Papert's constructionism is the practical realization of the constructivism theory. Papert (1980) noted that individual learning occurs more effectively when students understand the world around them by creating something that is meaningful to themselves. These can be artefacts such as a sand castle or a computer program (Parmaxi and Zaphiris, 2014). Programming tools such as Scratch[6] make programming accessible to a large number of people and teach new skills and abilities, such as engineering, design, and coding (Blikstein and Krannich, 2013).

Research setting

The NOLB project plans to validate its approach by conducting three pilot studies from January 2015 to June 2017 in Austria, the UK, and Spain. In each of these pilots participate approximately 200 students (including the control groups) between the ages of 9 and 18 years to study different social exclusion problems. The framework for this study refers to the theory of constructionism, which emphasizes design and sharing of artefacts (Parmaxi and Zaphiris, 2014). This section describes the NOLB project, the Pocket Code app, the research setting of the Austrians pilot, and the feasibility study.

The "No One Left Behind" project

The "No One Left Behind" (NOLB) project[7] aims at unlocking game creation by students who are socially excluded. Students will use Pocket Code to develop games on mobile devices, with the goal of enhancing their abilities across all academic subjects as well as their computational proficiency, creativity, and social skills. The pilots address three inclusion challenges: gender exclusion, disabilities, and immigration. We plan to use a constructivism approach by learning through Pocket Code. This app will support students in constructing their own games in the context of their regular courses, by embedding academic elements in their games and reflecting their understanding of what they need to accomplish. Moreover, students can socialize with their peers during the game making process. The results will include designed sharable artefacts that reflect the students' different styles of thinking and learning (Ramnarine-Rieks, 2012).

Pocket Code

Pocket Code is an application for mobile devices and tablets. It is currently available for Android (a version for iOS and Windows Phone is in development).This app provides a visual programming language which allows students to create their own games, animations, interactive music videos, and many types of other apps, directly on their smartphones or tablets (Catrobat, 2015). Similar to Scratch, programs in Pocket Code are created by visually composing Lego®-style bricks arranged in "scripts" which can run in parallel, thereby allowing concurrent execution.

Setup in Austria: Empowering girls' computational skills

The Austrian pilots are conducted together with three high schools situated in and around Graz: "Akademisches Gymnasium Graz", "Graz International Bilingual School" (GIBS), and "BORG Birkfeld". 190 students from 11 to 16 years, of which 122 are girls, take part in the study. The intent is to find out how to develop Pocket Code in ways that specifically empower girls by engaging with them and thus making the study of STEM related subjects more attractive to them. Additionally,

[6] https://scratch.mit.edu/

[7] http://no1leftbehind.eu

the differences in the game creation process will be analysed. Aspects of self-identity and stereotyped gender categories will be taken into consideration.

Feasibility Study: Integrating Pocket Code in school lessons

Figure 2 outlines the different stages of the feasibility study. Pocket Code is used in several academic subjects, thus the purpose of this study is twofold. First, the feasibility pilot allows students and teachers to familiarize themselves with Pocket Code through applying game design elements in selected curricula areas. By this we will identify students' and teachers' needs regarding the app. Second, the study gives students an initial positive experience with the app. Pocket Code will allow them to engage with their subjects in a playful way.

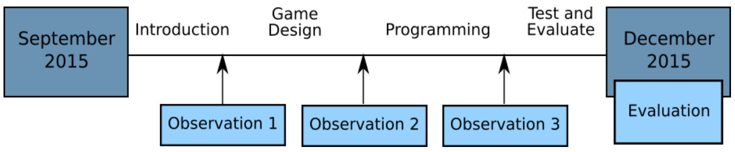


Figure 2. Schedule of the feasibility study.

The results of the study will shape the new version of Pocket Code through the feedback from students and teachers. At the end of the study the following evaluations are planned:

- Analysis of data collected (pre- and post-surveys, observations during the study).
- Defining the range of capabilities measured through gaming analytics and the progression of the learners.
- Learning about the barriers of using Pocket Code in schools.
- Measuring the learning objectives against the learning outcomes.
- Documenting findings to improve future studies and developments.
- Reviewing school capabilities and issues.
- Exploring the role of the teachers (teachers as facilitators, coaches).

Additionally, the following aspects are specific to the Austrian study:

- Differences in game creation and design between girls and boys.
- Suggested changes that are needed to make Pocket Code more attractive to girls.

Preparation phase and feasibility study in Austria

This section describes the preparation phase in Austrian pilot schools and provides a summary of the workshops with students as well as an overview of the teacher training sessions and prepared material. We further describe how findings from the preparation phase have influenced the ensuing feasibility study and the students' lessons with Pocket Code.

Pocket Code workshops with students

At the end of the school semester in July 2015, we organised workshops in two of our partner schools, with a focus on introducing Pocket Code to students. The theme for the workshop was "150 years of Alice in Wonderland" in reference to the 150th anniversary of Lewis Carroll's book celebrated in 2015. This topic seemed interesting to the target audience: students 14-17 of both genders. For the workshop itself, various materials were prepared, including media assets such as graphics, sounds, and tutorial cards for important functions (e.g., set size of an object). One NOLB partner, the National Videogame Arcade (NVA) in the UK, has developed a taxonomy called "The Shape of a Game" ceremony. It consists of a title screen, an instruction screen, a game screen, and an end screen. This framework was preinstalled on the mobile phones the students used for the workshop. With this framework the students' game design processes were scaffold, allowing them to focus on the game development itself. At GIBS 19 and at Borg Birkfeld 35 students attended the four hour workshop.

The introduction session was held in front of the whole class to show what could be achieved within Pocket Code, with the team presenting example games and the user interface. Afterwards the students formed groups of two or three and pitched game ideas. These ideas were shared with the class and the students got direct feedback. Next, the students created storyboards which helped them to get a clearer image of the gameplay and characters. Without giving them further guidance in programming, they started to code their games. The team took the role of coaches, and students could consult them when they needed some input. This learning by doing approach supports the constructionist theory, and most importantly students had the opportunity to add their creativity to this session and explore the various functionalities of Pocket Code on their own. Figure 3 illustrates different impressions of the workshop. At the end they uploaded their games to our web share page[8] and presented them in front of their peers.

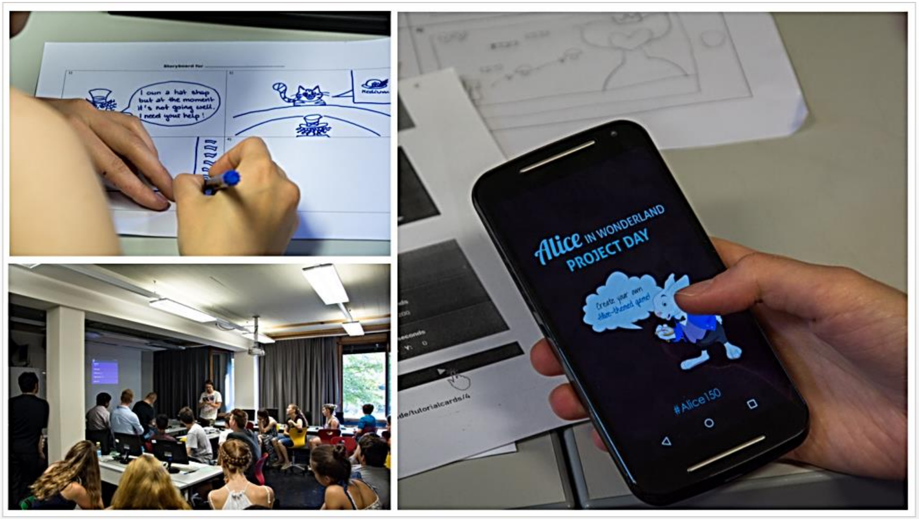


Figure 3. Pictures of the workshop: storyboard, instruction session and framework "Shape of a Game".

In both workshops many different games were created, e.g., mazes, skill games, jump and run games, quizzes, adventure games, or racing games. We conducted a survey at the end of the workshops. The results showed that 98% of the students described the workshop as very good or good, 90% were satisfied with their results, and 89% liked to work in teams. While observing the students use of Pocket Code, challenges could be identified, e.g., working in groups or difficult parts while programming. The students had especially problems in creating the following functions: to detect a collision, to move the background, and to use variables. Furthermore, the games created have been analysed in order to define what kind of game design elements were used most often and to generate therefore tutorials cards, tutorial videos and frameworks with Pocket Code. Additionally, the workshop provided a good opportunity to gain more experience in applying game oriented methods within the curricula.

Pocket Code teacher training sessions

Before finalising the planning phase, the teachers needed to be involved as well. The success of the project depends on their participation and cooperation. Influencing factors include the subjects in which teachers will use Pocket Code and the amount of units available. Preliminary work included several meetings with teachers in spring 2015 and conducting an online questionnaire to establish the teachers' digital skills and abilities. Initial results were used to develop the classroom resources that teachers need to implement, based on their differentiated skill levels. In September 2015, the first teacher trainings were held to show the functionalities of Pocket Code. In the two hour courses, teachers were given a short introduction to Pocket Code, after which they had a hands-on session in form of a step-by-step approach with the workshop facilitator. The sessions

[8] https://pocketcode.org/

were followed by short discussion rounds where teachers considered how Pocket Code could be used best to support their lessons. Afterwards the teachers created some excellent ideas together with their students. In the following weeks they sent these ideas either in form of storyboards or description to the NOLB team and got feedback from us. After both the teacher trainings and the student workshops, the following materials were prepared to kick off the feasibility study:

- Online course[9] including explanation of the bricks, tutorial cards for students, and helpful game metrics (shape of a game, game design process)
- Short video tutorials with gaming concepts[10]
- Pocket Code frameworks for Physics, Arts, English, Computer Science and Music[11]
- Media assets like graphics and sounds online through the Pocket Code Media Library

The effectiveness and usefulness of this material will be tested during this feasibility study.

Kick-off of the feasibility study

With the help of the NOLB team the teachers will use different approaches depending on the age of their students, their previous programming experience and their subject field. Table 1 points out the various approaches which all have a constructionist base in common e.g., students will be actively involved in the process of constructing a game, they will take charge of their own learning and find solutions by using the tutorial cards, asking their peers or just trying out alternatives and learn from their mistakes. Therefore, students will learn to be patience, feel ownership of their achievements and will truly understand many of the basic programming steps. To archive this, students will work in at least three units; in the Austrian pilots, girls will be grouped together. For the purpose of designing and creating artworks for backgrounds and objects we will provide students enough time during additional Arts lessons (important for girls see previous section).

Pilot school 1: "Akademisches Gymnasium"	
Arts (2nd grade)	Idea: 17 different languages are spoken in this class, thus the idea is to create a vocabulary game. Game play: A list with different vocabulary words is shown on the screen. In the game, balloons with pictures glide. The player should try to catch the balloons that show vocabulary words from the list using the inclination sensors of the phone. A score displays changes with right/wrong catches, and when all right balloons are caught the game is finished. The students should add functionalities, own graphics, and other languages/vocabulary to their games.
Computer Science (5th grade)	Idea: In these units the students will not receive a given project theme. The aim is that they will be able to work creatively and to explore the basics of a programming language on their own. The sequence of this lesson will be similar to the workshop: They will create a storyboard, present their ideas, and start programming.
Pilot school 2: "GIBS"	
Physics (3rd grade)	Idea: This game should deal with the density of different objects and liquids. The lessons should follow up with performing these experiments in real life. Game play: The game shows the behaviour of different objects (e.g., a bolt, a cork) falling in different liquids (e.g., water, oil). The students should add further objects (e.g., paper, coins) and liquids (e.g., honey).
English & Computer Science (3 classes 5th grade)	Idea: The idea is to program a quiz game with the students. In the English class the students had to read five different books and can choose which one of them they will use for the quiz. Game play: In the Computer Science class students will program these quizzes. At five different stations questions will be asked concerning the book, followed by a small mini game. If all questions are answered correctly, the game is finished. The students should create the stations with questions and mini games.
Arts (5th and 6th grade)	Idea: This teacher will use a different approach. He will use the brick cards the NOLB team created, print them out, and use them as a physical flashcard system to put together scripts. Thus students draw first the projected outcome on a picture. When they start programming their games in Pocket Code, they can see if it follows the same configuration that they predicted. They may choose the topic of their games freely.
Pilot school 3: "Borg Birkfeld"	

[9] https://edu.catrob.at/

[10] https://share.catrob.at/pocketcode/gaming-tutorials

[11] https://share.catrob.at/pocketcode/profile/3611

Computer Science (5th grade)	Idea: This class will do also a quiz game in Pocket Code and answer computer related questions. Game Play: The player listens to a question, e.g., "Catch all object-oriented programming languages" and then catch the correct image from several that are falling. A score display changes with right/wrong catches. When all questions are answered correctly the game is finished. The students should add additional questions to the game.
Music (5th grade)	Idea: Same concept as in Computer Science. Game Play: The player listen to music, e.g., "The Magic Flute" and then the player should catch the instruments that were part of the music. The students should add additional music and instruments.

Table 1. Ideas for the students' lessons in the different pilot schools.

Students of the 2nd grade will get a framework in which specific parts in the code are missing (indicated by a note brick): e.g., collision detection or using inclination sensors. Within small groups the team develops this missing functionality together with them to guarantee student-centered classroom, where students learning and discovery is in their own hands. Students up to 3rd grade will get a short introduction unit including a hands-on sessions guided by the team and tutorial cards. In these sessions one student at a time comes to the front of the class and tries to add one small but meaningful new feature to the game being developed by the class.

Pre and post questionnaires, observations, documentations, video, and picture material will be collected during all units. The outcomes of the feasibility study will be incorporated in the next planned cycles. These findings will be integrated in a new generation of Pocket Code, which will be designed and implemented especially for school contexts. For the first cycle in summer semester 2016 we will continue with the same classes that are already experienced in Pocket Code. The new version will be tested and evaluated during the second cycle, which will start in September 2016. In order to monitor these cycles, NOLB will select an experimental design (experimental vs. control group).

Future work

Future work will include the testing of more playable approaches to aid girls in certain topics. One way will be through Game Jams events. Goddard *et al.* (2014) see Game Jams as a way to generate and inspire game ideas and finding new ways of thinking. Thus NOLB will examine the opportunity of utilizing Game Jams as a design research tool in school classes. One event (at the time of this writing still upcoming) which will be also interesting for schools is the Alice Game Jam[12] in December 2015.

A series of data collection tools (teachers' questionnaire, student gaming questionnaire, and baseline collection tool) have been designed for understanding students' academic, social and inclusion needs, requirements and capabilities. They will help to identify students' and teachers' needs to guide the development of an inclusive generation of Pocket Code, influence the design of the full experimental pilots and provide benchmarks against which we can evaluate the system's potential in generating inclusive game creation experiences in formal learning situations.

Discussion and conclusion

Many of the changes in teaching and learning that resulted from the study of empowerment of girls improved the situation for all students, not just for girls (Kafai, 2008). Within NOLB it is assumed that learners who become game designers and creators will significantly contribute to closing the divide and participation gap in digital culture. Preliminary findings show that students were able to successfully design functional games in a very short time frame using Pocket Code. A more detailed analysis will be available after the feasibility study, the findings of which will help to understand the game design behaviour of girls and to identify obstacles in usage of Pocket Code.

[12] www.alicegamejam.com

Acknowledgements

This work has been partially funded by the EC H2020 Innovation Action No One Left Behind, Grant Agreement No. 645215.

References

Beltrán, M. *et al.* (2015) *Inclusive gaming creation by design in formal learning environments: "girly-girls" user group in No One Left Behind*, In Design, User Experience, and Usability: Users and Interactions. Los Angeles. July, pp. 153 – 161.

Blikstein, P. and Krannich, D. (2013) *The Makers' Movement and FabLabs in Education: Experiences, Technologies, and Research*. Proceedings of the 12th International Conference on Interaction Design and Children. New York. June, pp. 613 – 616.

Craig, A.; Coldwell-Neilson, J. and Beekhuyzen, J. (2013) *Are IT interventions for Girls a Special Case?* In Proceeding of the 44th ACM technical symposium on Computer science education. Denver, March. pp. 451 – 456.

Chandrasekaran, S. (2012) *Learning through Projects in Engineering Education*. SEFI 40th annual conference. Greece. 2012

Goddard, W.; Byrne, R. and Mueller, F. (2014) Playful Game Jams: Guidelines for Designed Outcomes. Proceedings of the 2014 Conference on Interactive Entertainment. Newcastle. December, pp. 1 – 10.

Greenfield, P. M. and Cocking, R. R (1996). *Interacting with video*. Norwood, NJ: Ablex (expanded version of Greenfield & Cocking, 1994).

Kafai, J. (2008) *Considering Gender in Digital Games: Implications for Serious Game Designs in the Learning Sciences*. In Proceedings of the 8th international conference on International conference for the learning sciences. Los Angeles, pp. 422 – 429.

Kafai, Y. and Resnick, M. (1996): *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Mahwah, New Jersey, Lawrence Erlbaum.

Papert, S. (1980) *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc.

Parmaxi, A. and Zaphiris, P. (2014) *Affordances of Social Technologies as Social Microworlds*. Extended Abstracts on Human Factors in Computing Systems. Toronto, April. pp. 2113 – 2118.

Ramnarine-Rieks, A. (2012) *Learning through Game Design: An Investigation on the Effects in Library Instruction Sessions*. Proceedings of the 2012 iConference. Toronto. Feb., pp. 606 – 607.

Schippers,S. and Mak. M *(*2015*) Creating Outstanding Experiences for Digital Natives. A survey of Digital Natives reveals a group of impatient users with fragmented attention spans who demand fast and intuitive products and services.* Article No*:* 1278*.* Retrieved from*:* http*://*uxmag.com*/*articles*/*creating-outstanding-experiences-for-digital-natives*.*

Sanchez, J. and Olivares, R. (2011) *Problem solving and collaboration using mobile serious games*. In Computers & Education 57(3). Santiago. November, pp. 1943 – 1952.

Catrobat (2015) *Catrobat*. Last visited: October, 6th 2015. Retrieved from: http://www.catrobat.org.

Weibert, A., Rekowski, T. and Festl, L. (2012) *Accessing IT: a curricular approach for girls.* In Proceedings of the 7th Nordic Conference on Human-Computer Interaction*:* Making Sense Through Design*.* Copenhagen, October*.* pp*.* 785 – 786.

Generalization processes: an experience using eXpresser with primary-school children

Cristianne Butto Zarzar *cristianne_butto @hotmail.com*
Universidad Pedagógica Nacional, Mexico

Joaquín Delgado *jdf@xanum.uam.mx*
Departamento de Matemáticas, UAM-Iztapalapa, Mexico

Ana Isabel Sacristán *asacrist@cinvestav.mx*
Departamento de Matemática Educativa, Cinvestav, Mexico

Abstract

In this paper, we report a research project in early algebraic thinking with students of primary school (10-12 year old) who have not yet been introduced to algebraic syntaxes. Using the software *eXpresser* (Figure 1), we introduced some algebraic ideas, along with the idea of generalization. This was carried out through problem-solving activities in a didactic sequence. Even though students in this age-group are in transition from additive to multiplicative thinking, our study revealed that students were capable of understanding the idea of sequence, and that they were able to identify the general rule and express it in pre-algebraic terms.

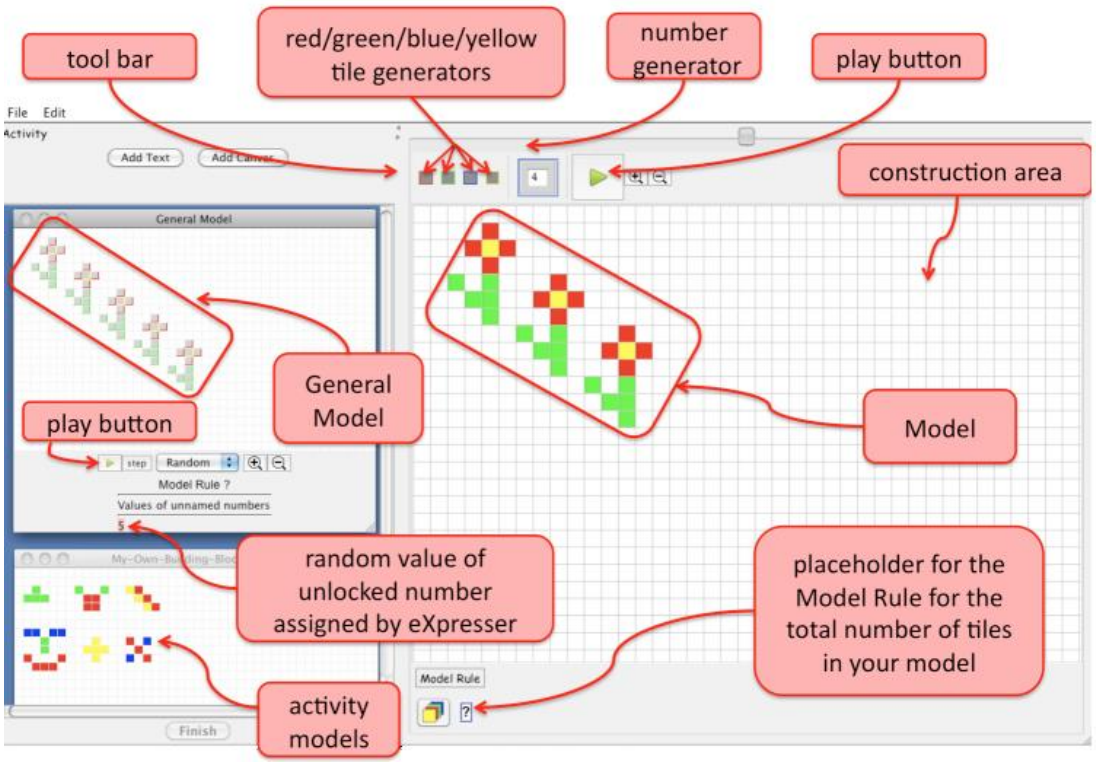

tool bar
red/green/blue/yellow tile generators
number generator
play button
construction area
General Model
play button
Model
random value of unlocked number assigned by eXpresser
placeholder for the Model Rule for the total number of tiles in your model
activity models
File Edit
Activity
Add Text
Add Canvas
General Model
step
Random
Model Rule ?
Values of unnamed numbers
Model Rule
Finish

Figure 1. Building patterns with eXpresser (https://migenproject.wordpress.com/)

Keywords

Generalization processes, early algebraic thinking, elementary school, eXpresser.

On early algebra

Many studies on early algebra have identified curricular issues in elementary school that can be explored in order to introduce some important algebraic ideas in students of this level (e.g. Da Rocha Falcão, 1993; Slavit, 1999, Carraher, Schliemann & Brizuela, 2000, 2001; Kaput & Blanton, 2000; Schliemann, Carraher, Brizuela & Earnest, 2003). These attempt to give plausible explanations on the difficulties that students face when they initiate the study of algebra in secondary school. One of these explanations is the lack of previous knowledge in students for dealing with numerical problems in such a way that it can lead to algebraic ideas, such as generalization and its expression.

According to Butto and Delgado (2012), it is important to distinguish *early algebra* from *pre-algebra*. The focus in early algebra are the thinking processes that lead to ideas in algebra, even when they are not fully developed, but that represent true routes of access to algebra. In pre-algebra, the focus is on the mathematical content previous to the formal introduction of equations and unknowns, such as integer and rational numbers, exponents, and surds. This is relevant for us, because in the Mexican curriculum, teaching and learning of algebra is postponed until the first and second years of secondary school (ages 12-14).

Teaching and learning algebra and generalization processes

In the literature there are at least four approaches to teaching and learning algebra (Bednarz, Kieran & Lee, 1996). (1) Generalization of numerical and geometrical patterns, and the laws governing the numerical relationships. (2) Modelling of mathematical situations and of concrete situations. (3) The study of functional relationships, and (4) the solution of problems and equations. Wheeler (cited in Mason, Graham, Pimm & Gower, 1985) points to the artificial character of such classification, since all four aspects are necessary in any elementary algebra program. Nevertheless this classification is useful in the didactics of mathematics, in order to know which of these four aspects is emphasised in a particular treatment in algebra teaching. Our interest is on the first: generalization.

Generalization processes consist in discovering a pattern or rule from a sequence of objects that can be numerical or geometrical. Pertinent investigations show that children can understand a rule, even though they may not be able to express it in what we call algebraic language. Nevertheless, they are able to construct a table and extrapolate or interpolate correspondences outside the data. According to Pegg (cited in Durán Ponce, 1999), discovering patterns requires three processes: (i) activities with numerical patterns; (ii) expressing the rules that characterize the particular numerical patterns by means of sentences (involving students in making clarifications and precisions); and (iii) promoting that students express such rules in short form. Radford (2006) makes the distinction of two types of generalization: one of arithmetic type, the other of algebraic type. The inductive procedures based on the formation of rules by trial and error in activities with patterns, do not lead to generalization.

eXpresser: a constructionist microworld

This path towards developing generalization processes in children can be supported by the use of technology. This is what the software eXpresser –part of the project “MiGen: intelligent support for mathematical generalization” (https://migenproject.wordpress.com/), led by R. Noss and collaborators– is designed for (Noss et al., 2011). This is a software for technology-enhanced learning, where children can construct patterns using tiles and colours, and encapsulate them in the form of blocks (shapes) that they can repeatedly reproduce by giving rules for shifting the block in the horizontal and vertical directions, as well as for how many times to repeat the block. This is achieved by means of placeholders that contain numbers and can be dragged. In this way children can repeat patterns and test their guesses relatively easy. As explained in the documentation on the eXpresser website (“Background | The MiGen Project,” n.d.), “this modelling approach is inspired by Papert’s notion of constructionism (e.g. Papert, 1990) which supports the idea that learning happens most effectively when learners are actively making things in the real world or, in other words, 'the idea that learners build knowledge structures particularly well in situations where they are engaged in constructing public entities'" Thus, eXpresser is considered a microworld, which Papert (1980,p. 204) defined as: " …a subset of reality or a constructed reality whose structure matches that of a given cognitive mechanism so as to provide an environment where the latter can operate effectively. […] microworlds so structured as to allow a human learner to exercise particular powerful ideas or intellectual skills."

Generalization with eXpresser

The eXpresser software allows students to explore, experiment and give meanings to algebraic symbols by means of figurative models and several levels of representation and generalization. Noss, Healy and Hoyles (1997) establish that it is important to introduce several approaches that allow students to build their own proper mathematical models. Figure 1 shows the general appearance of eXpresser (https://migenproject.wordpress.com/using-migen/expresser-microworld/). It contains two windows: "My world" on the right and "General world" on the left. In "My world", children drag tiles of different colours from the tile generator to the construction area and group them together into a building block. Then, they can repeat the building block to build a pattern, entering the number of horizontal and vertical shifts and the number of times the building block is repeated. These numbers can be entered in the number generator and then dragged into the cells of the building pattern window. When a pattern is built, the properties window appears, where children can drag numbers that give the right number of tiles of a particular colour. This is shown in Figure 1 (right), where the number of green tiles in the pattern are correctly answered and the corresponding tiles are highlighted. The number of tiles of other colours can be given similarly, by dragging numbers into the question marks icons. When the child is ready to guess a rule, (s)he unblocks, for instance, the number of blocks, in the properties window and drags it into the cells below the text "How many tiles?" (see Figure 1, left), to construct formulas involving the general number –the unblocked number thus acts as a general number or variable– and arithmetic operations: sum, difference, multiplication and substitution, which yield the number of tiles of a particular colour. When the number of tiles of a particular colour are computed correctly, then the tiles are highlighted in the pattern. But the pattern in the "General world" is not highlighted until the right formula is given in the placeholder of the Model rule below, for the total number of tiles. When this is done, then the pattern in the "General world" will be highlighted and animated, generating random values for the general number. Thus patterns in both, "My world" and in "General world" look similar, but when the model in "My world" is valid for any value of the variable, then in the "General world", random values are given to the variable and the pattern flashes, showing that generalization has been reached.

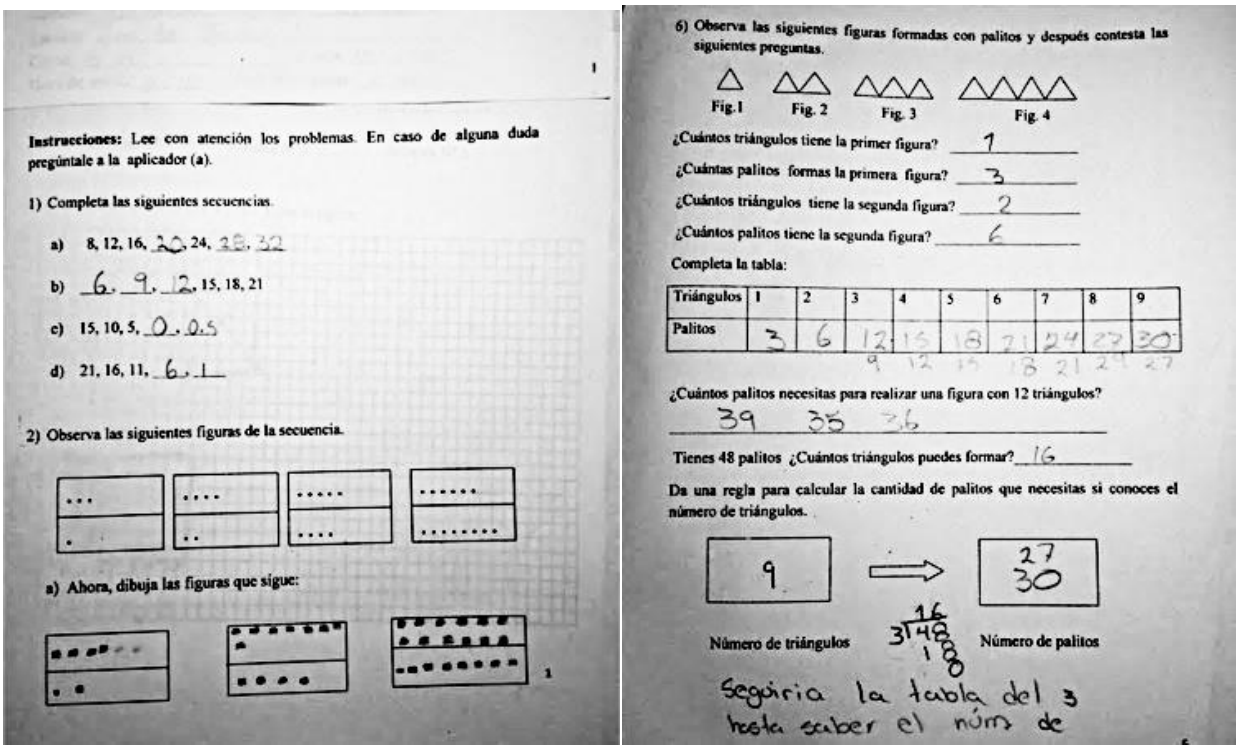
A study using eXpresser for early algebra in primary school children

In our study, we investigated a route towards early algebraic thinking in primary-school students in 5th-6th grades (10-12 year old), based on generalization processes, mediated by the use of eXpresser and paper and pencil,. Participants were 30 students from a public primary school in Mexico City.

We report here on two stages of the study**:** First, the application of an initial questionnaire followed by a clinical interview, which had as aim to explore the initial knowledge of arithmetical and geometrical sequences and, second, a didactical sequence using eXpresser.

First stage: the initial questionnaire

The initial questionnaire had the purpose of exploring children's abilities, strategies and difficulties in the identification of geometric and numerical sequences, as well as if they were able to understand the rule for constructing sequences in several ways. These issues were explored further by means of a clinical interview. The questionnaire explored: the recognition and construction of elementary sequences of numbers and figures; the comparison of an arithmetical sequence with a geometrical sequence of points; the construction of a rule in building a sequence of triangles with sticks; counting of tiles in the perimeter and inner area of a sequence of squares (swimming pool). For this, children were asked to complete simple numerical increasing and decreasing sequences, and explore if they could distinguish between an arithmetical and a geometrical sequence, completing a list of simple dots (Figure 2, left). Then, for a given sequence of triangles (Figure 2, right), students were asked to give the number of sticks needed for a particular number of triangles; to fill out a table of number of triangles vs. number of sticks; and give a formula (in this case, the multiplication of number of triangles times the number of stick needed). Then, for a particular value of number of sticks, they were asked to "invert" and give the corresponding number of triangles. The interviewer read the questions, asking the students to answer them. Figure 2 shows the answers given by a student in part of the initial questionnaire.


1

Instrucciones: Lee con atención los problemas. En caso de alguna duda pregúntale a la aplicador (a).

1) Completa las siguientes secuencias.

a) 8, 12, 16, 20, 24, 28, 32

b) 6, 9, 12, 15, 18, 21

c) 15, 10, 5, 0, 0.5

d) 21, 16, 11, 6, 1

2) Observa las siguientes figuras de la secuencia.

a) Ahora, dibuja las figuras que sigue:

1

6) Observa las siguientes figuras formadas con palitos y después contesta las siguientes preguntas.

¿Cuántos triángulos tiene la primer figura? 1

¿Cuántas palitos formas la primera figura? 3

¿Cuántos triángulos tiene la segunda figura? 2

¿Cuántos palitos tiene la segunda figura? 6

Completa la tabla:

Triángulos	1	2	3	4	5	6	7	8	9
Palitos	3	6	12	15	18	21	24	27	30
			9	12	15	18	21	24	27

¿Cuántos palitos necesitas para realizar una figura con 12 triángulos?
39 35 36

Tienes 48 palitos ¿Cuántos triángulos puedes formar? 16

Da una regla para calcular la cantidad de palitos que necesitas si conoces el número de triángulos.

$\boxed{9} \Longrightarrow \boxed{\begin{matrix}27\\30\end{matrix}}$

Número de triángulos Número de palitos

$3\overline{)48}$ 16 18 0

Seguiría la tabla del 3 hasta saber el núm de

Figure 2. Examples of questions from the initial questionnaire.

The initial questionnaire revealed that most of the students were acquainted with numerical increasing and decreasing sequences, but did not continue sequences through negative values. They were able to give a rule on the number of sticks needed for a given number of triangles at different degrees of generality: answers such as "count the number of sticks"; "they are multiples"; "use the three-times table", or "for 4, I need 12" showed that some students were in transit from additive to multiplicative thinking. The "inversion" of a particular item in the series presented some difficulties, even though it reduces to a division without remainder (48/3 = 16), as the following transcript from the clinical interview shows:

Interviewer (I) with a student (S)

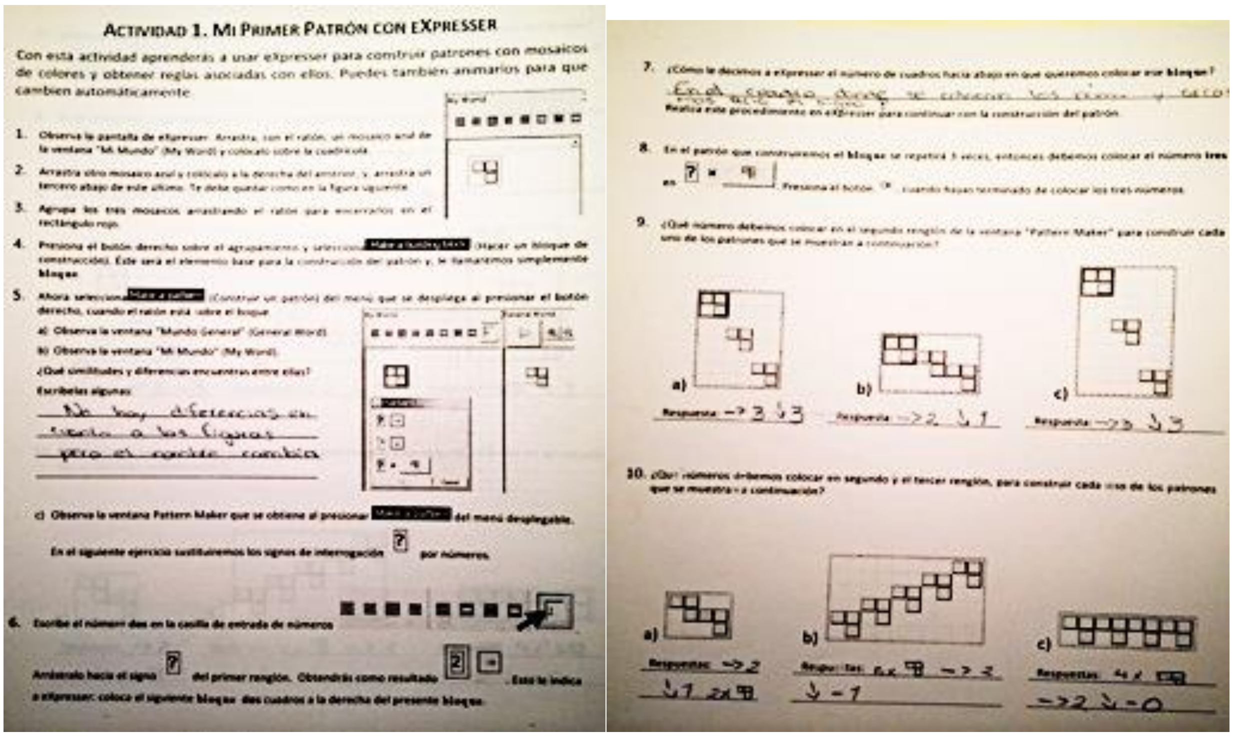
S: You have 48 sticks, how many triangles can you form? It would be 3 times 48.
I: Here is your sheet if you want (the interviewer brings the sheet close)
S: (she makes the math and answers) 144
I: Then you could form 144 triangles with 48 sticks, but where there is more? When we go doing this small table (the interviewer points out to the sheet) where there is more?, in the column of sticks or in the column of triangles?
S: in the sticks

I: Then with 48 sticks can I form more triangles?
S: I don't know.
I: How would you do? Here it says that forming a triangle need three sticks. It is just like this (the interviewer forms a triangle with pens), to form a triangle we need three stick, ok?
I: To form two, we need six. And how do you figure out with 48 sticks, if we need three to form each?.
S: I would continue with the table until reaching 48.
I: Do you want to do it?
S: It would take too long.
I: But you are now at 36, which is the next number?
S: 39
I: for 13 triangles, how many sticks do you need?
S: 39 for 14; 42 for 15; 45 for 16; 47 (erases an operation)
I: no, you don't erase it; let it that way, if you want we can continue here
S: I is not an 7, it is an 8… here it goes for17, 51.
I: (interrupts) and how many sticks did we have?
S: 48
I: Then, how many triangles can we form?
S: 16
I: Very well! Let us see the last one. Give me a rule to calculate the number of sticks you need, if we change the number of triangles.

Second stage: Didactic sequence

The didactic sequence on generalization processes was structured using worksheets. Children worked in pairs with eXpresser, and were asked to draw the resulting designs in the worksheets. There were two activities named "My first pattern with eXpresser" and "Working in the general world". In the first activity, children followed the instructions written in the worksheets and recorded their results from the "My world" window of eXpresser. In the first activity, they were asked to build a simple block to reproduce the patterns shown in the working sheet. Then they answered in their own writing and notation the steps they followed when shifting and repeating the building block to construct the pattern. In this way, they experienced their own ways to describe symbolically what they were doing. Then, for a particular number of times that the building block was repeated, children were asked to give the number of coloured tiles. At this stage, they explored with a single colour. They learned how to unblock a number and animate the pattern within eXpresser. In the activity named "Working in the general world", children constructed a pattern from a simple building block and verified that when the number of repetitions is changed, and the right number of tiles is the input in the properties window, then the pattern becomes highlighted. They were given a paired list of repetitions of building block and number of tiles in the pattern, some of them right, some of them not, and children verified (yes or no) in which cases the pattern became highlighted. Then they were asked to give a formula for the right (yes) cases. A second paired list of number of repetitions and highlighted patterns (yes) was given and children were asked to find the number of tiles, and try to give a formula. In a third single list, the number of repetitions was given and they had to give the number of tiles, and verify that the pattern became highlighted when the number of tiles was computed correctly (yes).

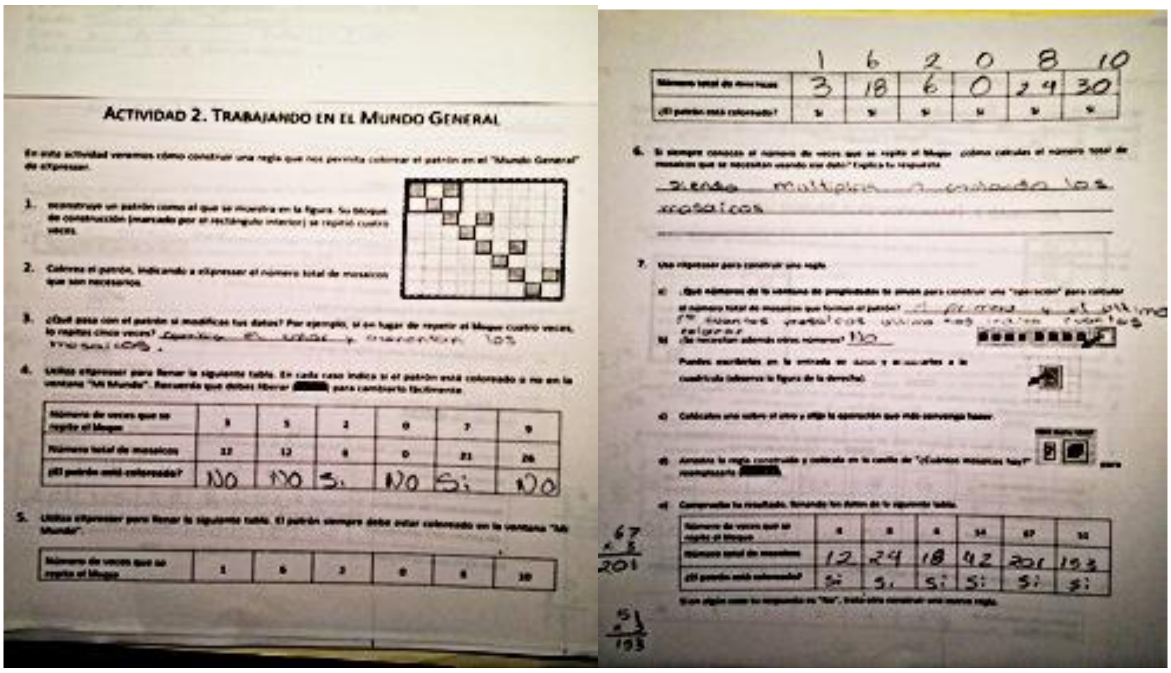
In the first activity, "My first pattern in eXpresser", students became familiar with building a pattern: When they were asked to write down the ways they construct several patterns from simple building blocks, they used expression like "#n right, #m down", or #n □, #m □ to say that moving the building block #n places right and #m places down yields the pattern. For left and upward shiftings, they even used negative numbers (which they hadn't used in the initial questionnaire). Figure 3 shows the worksheet with the first items of this activity, for the same student whose answers are shown in the initial questionnaire scans in Figure 1. We can see here, the use of a syncopated notation using arrow symbols.


ACTIVIDAD 1. MI PRIMER PATRÓN CON EXPRESSER

Con esta actividad aprenderás a usar eXpresser para construir patrones con mosaicos de colores y obtener reglas asociadas con ellos. Puedes también animarlos para que cambien automáticamente.

1. Observa la pantalla de eXpresser. Arrastra, con el ratón, un mosaico azul de la ventana "Mi Mundo" (My World) y colócalo sobre la cuadrícula.
2. Arrastra otro mosaico azul y colócalo a la derecha del anterior, y arrastra un tercero abajo de este último. Te debe quedar como en la figura siguiente.
3. Agrupa los tres mosaicos arrastrando el ratón para encerrarlos en el rectángulo rojo.
4. Presiona el botón derecho sobre el agrupamiento y selecciona Make a building block (Hacer un bloque de construcción). Este será el elemento base para la construcción del patrón y, le llamaremos simplemente bloque.
5. Ahora selecciona Make a pattern (Construir un patrón) del menú que se despliega al presionar el botón derecho, cuando el ratón está sobre el bloque.
   a) Observa la ventana "Mundo General" (General World).
   b) Observa la ventana "Mi Mundo" (My World).
   ¿Qué similitudes y diferencias encuentras entre ellas?
   Escribelas algunas:
   No hay diferencias en [illegible] a las figuras pero el [illegible] cambia
   c) Observa la ventana Pattern Maker que se obtiene al presionar Make a pattern del menú desplegable.
   En el siguiente ejercicio sustituiremos los signos de interrogación ? por números.
6. Escribe el número dos en la casilla de entrada de números.
   Arrástralo hacia el signo ? del primer renglón. Obtendrás como resultado 2. Esto le indica a eXpresser: coloca el siguiente bloque dos cuadros a la derecha del presente bloque.
7. ¿Cómo le decimos a eXpresser el número de cuadros hacia abajo en que queremos colocar ese bloque?
   En el espacio donde se colocan los [illegible] y [illegible]
   Realiza este procedimiento en eXpresser para continuar con la construcción del patrón.
8. En el patrón que construiremos el bloque se repetirá 3 veces, entonces debemos colocar el número tres en ? × [illegible]. Presiona el botón [illegible] cuando hayas terminado de colocar los tres números.
9. ¿Qué número debemos colocar en el segundo renglón de la ventana "Pattern Maker" para construir cada uno de los patrones que se muestran a continuación?
   a) Respuesta: ->3 ↓3
   b) Respuesta: ->2 ↓1
   c) Respuesta: ->3 ↓3
10. ¿Qué números debemos colocar en segundo y el tercer renglón, para construir cada uno de los patrones que se muestran a continuación?
   a) Respuestas: ->2 ↓1 2×[illegible]
   b) Respuestas: 6× [illegible] ->2 ↓-1
   c) Respuestas: 4× [illegible] ->2 ↓-0

Figure 3. An example of the worksheet for the activity "My first pattern in eXpresser", from the same student whose answers are shown in the initial questionnaire in Figure 1.


ACTIVIDAD 2. TRABAJANDO EN EL MUNDO GENERAL

En esta actividad veremos cómo construir una regla que nos permita colorear el patrón en el "Mundo General" de eXpresser.

1. Construye un patrón como el que se muestra en la figura. Su bloque de construcción (marcado por el rectángulo interior) se repitió cuatro veces.
2. Colorea el patrón, indicando a eXpresser el número total de mosaicos que son necesarios.
3. ¿Qué pasa con el patrón si modificas tus datos? Por ejemplo, si en lugar de repetir el bloque cuatro veces, lo repites cinco veces? Cambia el color y [illegible] los mosaicos.
4. Utiliza eXpresser para llenar la siguiente tabla. En cada caso indica si el patrón está coloreado o no en la ventana "Mi Mundo". Recuerda que debes liberar [illegible] para cambiarlo fácilmente.

Número de veces que se repite el bloque	3	5	2	0	7	9
Número total de mosaicos	12	12	6	0	21	26
¿El patrón está coloreado?	No	No	Si	No	Si	No

5. Utiliza eXpresser para llenar la siguiente tabla. El patrón siempre debe estar coloreado en la ventana "Mi Mundo".

Número de veces que se repite el bloque	1	6	2	0	8	10
Número total de mosaicos	3	18	6	0	24	30
¿El patrón está coloreado?	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

6. Si siempre conoces el número de veces que se repite el bloque ¿cómo calculas el número total de mosaicos que se necesitan usando ese dato? Explica tu respuesta.
   Se van multiplicando los mosaicos
7. Usa eXpresser para construir una regla.
   a) ¿Qué números de la ventana de propiedades te sirven para construir una "operación" para calcular el número total de mosaicos que forman el patrón? el primero y el último [illegible]
   b) ¿Se necesitan además otros números? No
   Puedes escribirlos en la entrada de datos y arrastrarlos a la cuadrícula (observa la figura de la derecha).
   c) Colócalos uno sobre el otro y elige la operación que más convenga hacer.
   d) Arrastra la regla construida y colócala en la casilla de "¿Cuántos mosaicos hay?" para [illegible]
   e) Comprueba tu resultado, llenando los datos de la siguiente tabla.

Número de veces que se repite el bloque	4	8	6	14	67	51
Número total de mosaicos	12	24	18	42	201	153
¿El patrón está coloreado?	Si	Si	Si	Si	Si	Si

Si en alguno caso tu respuesta es "No", trata otra construir una nueva regla.

$$\begin{array}{r} 67 \\ \times 3 \\ \hline 201 \end{array} \qquad \begin{array}{r} 51 \\ \times 3 \\ \hline 153 \end{array}$$

Figure 4. An example of the same student's worksheet for the activity "Working in the general world".

In the second activity, "Working in the general world" children built a pattern and learnt how to get the pattern highlighted when they tested their rule. Students were able to explain the rule. Figure 4 shows the worksheets from the same student whose answers were shown in previous figures.

Discussion and didactical considerations

Although initially (as indicated by the answers to the initial questionnaire and in the interviews) children exhibited some difficulties in recognising quantities which could act as variables, through the construction of patterns with tiles or coloured squares in the eXpresser microworld, expressing formulas for certain patterns, and reflecting on how to answer the questions on the worksheets (which included, for example, questions where they had to identify what numbers would form certain patterns), students seemed to overcome some of their initial difficulties. The fact that eXpresser provides a numerical and symbolic environment and syntax, gives students the possibility to express and communicate their ideas and experiment with different types of numerical and/or symbolic relationships.

Thus, students could come close to algebraic thinking by means of generalization of rules from simple patterns; the activities made them think about the relationships among variables, giving them the opportunity to operate mentally with these objects. They were thus able to symbolize some of the variables. And even though some of their abstractions were expressed, for instance, using syncopated notation (i.e. with arrow symbols), which shows the relationship of these with the construction process in the eXpresser microworld (thus indicating that these were *situated abstractions* –Noss & Hoyles, 1996), students were able to build some general arithmetic relationships that indicate some progress in the transit from symbolic to algebraic thinking, in a more structured way.

More specifically, children were able to understand the construction of sequences although they couldn't always express the rule in complete algebraic form. But they were able to recognize variable quantities, which is an important for developing the idea of an unknown in algebra.

Our study seems to indicate that early introduction to algebraic thinking by means of generalization processes is a feasible route at early ages (for 10-12 year old), jointly with the design of activities in two contexts: paper and pencil, and eXpresser, as has been found by other authors (e.g. Geraniou, Mavrikis, Noss & Hoyles, 2009). Thus, the eXpresser software is a powerful tool for initiation in algebraic symbolisation and for developing generalisation-related ideas.

However, the key point in using this software is to take advantage of its property of integrating the numerical and geometrical contexts, *within* a structured didactical sequence (guided by the teacher) that combines activities and materials in the two contexts (paper and pencil and eXpresser

Acknowledgment

This work was partially supported by the Conacyt Grant #145906: "Introducción temprana al pensamiento algebraico en entornos tecnológicos de aprendizaje: un estudio teórico-experimental en el nivel básico".

References

Background | The MiGen Project. (n.d.). Retrieved November 10, 2015, from https://migenproject.wordpress.com/theory/

Bednarz, N, C. Kieran, & L, Lee (1996). *Approaches to Algebra: Perspectives for Research and Teaching.* Netherlands: Kluwer Academic Publishers.

Butto, C & Delgado, J. (2012), Rutas hacia el álgebra; actividades en Excel and Logo, México: UPN, CONACYT.

Carraher, D., Schliemann, A. & Brizuela, B. (2000). Early Algebra, Early Arithmetic: Treating Operations as Functions. In: M.L Fernández (ed.), *Proceedings of PME- NA XXII*; Tucson, AZ.

Carraher, D., Schliemann, A., & Brizuela, B. (2001). Can Young Students Operate on Unknowns? In M. van den Heuvel-Panhuizen (Ed.) *Proceedings of PME 25*, Utrecht, The Netherlands Vol 1, pp. 130-140.

Da Rocha Falcao, J. (1993). A algebra como ferramenta de representaçao e resoluçao de problemas. In A. D. Schliemann, D.W. Carraher, A.C. Spinillo, L.L. Meira y J.T. da Rocha Falcao (1993), *Estudos em Psicología da Educaçao Matemática,* Recife, Editora Universitaria, UFPE.

Durán Ponce, R. (1999). Reconocimiento de patrones en secuencias numéricas y de figuras, por alumnos de sexto grado de primaria. Master's Thesis. México: Cinvestav

Geraniou, E., Mavrikis, M., Noss, R., Hoyles, C. (2009). A learning environment to support mathematical generalization in the classroom. In *Proceedings of CERME 6,* Lyon, France Available at: http://ife.ens-lyon.fr/publications/edition-electronique/cerme6/wg7-09-geranioumavrikis-hoylesnoss.pdf

Kaput, J. & M. Blanton, (2000). Generalization and Progressively Formalizing in a Third-Grade Mathematics Classroom: Conversations about even and Old Numbers. In: M.L Fernández (ed.), *Proceedings of PME- NA XXII*; Tucson, AZ

Mason, J. Graham, A.; Pimm, D, & Gower, N. (1985). *Routes of Roots of Algebra*. The Open University Press, UK.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: learning cultures and computers*. Dordrecht: Kluwer.

Noss, R., Poulovassilis, A., Geraniou, E., Gutierrez-Santos, S., Hoyles, C., Kahn, K., Magoulas, G. D., and Mavrikis, M. (2011). The design of a system to support exploratory learning of algebraic generalization. *Computers & Education* 59 (1): 63-81.

Papert, S. (1980). Computer-based microworlds as incubators for powerful ideas. In R. Taylor (Ed*.), The computer in the school: Tutor, tool, tutee* (pp. 203–210). New York: Teacher's College Press.

Papert, S. (1990): An Introduction to the 5th anniversary collection. In I. Harel (Ed.), *Constructionist Learning: A 5th anniversary collection of papers*. Cambridge, MA: MIT Media Laboratory.

Radford, L. (1996). The Role of Geometry and Arithmetic in the Development of Algebra: Historical Remarks from a Didactic Perspective. In Bernardz, N, Kieran, C and Lee, I (eds). *Approaches to Algebra. Perspectives for Research and Teaching*. Netherlands: Kluwer Academic Publishers.

Slavit, D.(1999) The Role of Operation Sense in Transitions from Arithmetic to Algebraic Thought. *Educational Studies in Mathematics, 37,* 251-274.

Grounding How We Teach Programming in Why We Teach Programming

Chris Proctor, *cproctor@stanford.edu*
Graduate School of Education, Stanford University

Paulo Blikstein, *paulob@stanford.edu*
Graduate School of Education, Stanford University

Abstract

This article extends a 2005 taxonomy of languages and environments for teaching computer programming to the current field of pedagogical programming languages and environments. The original taxonomy organized tools according to the barriers to learning programming they addressed, and the strategies used to surmount or avoid the barriers. Updating the taxonomy was not entirely successful; some strategies have emerged as widely-used best practices. As computers and programming have become more prevalent in everyday life, it has become harder to distinguish programming from non-programming, and harder to distinguish tools for learning from ordinary computational tools. Programming, formerly a specialized activity, has developed into a computational literacy comprising many different forms of cultural interaction. We propose a new taxonomy based on DiSessa's analysis of the structure of literacy. The new taxonomy allows learners, teachers, and designers to ground decisions about how to learn or teach programming in literacy aims expressing why they want to learn to program .

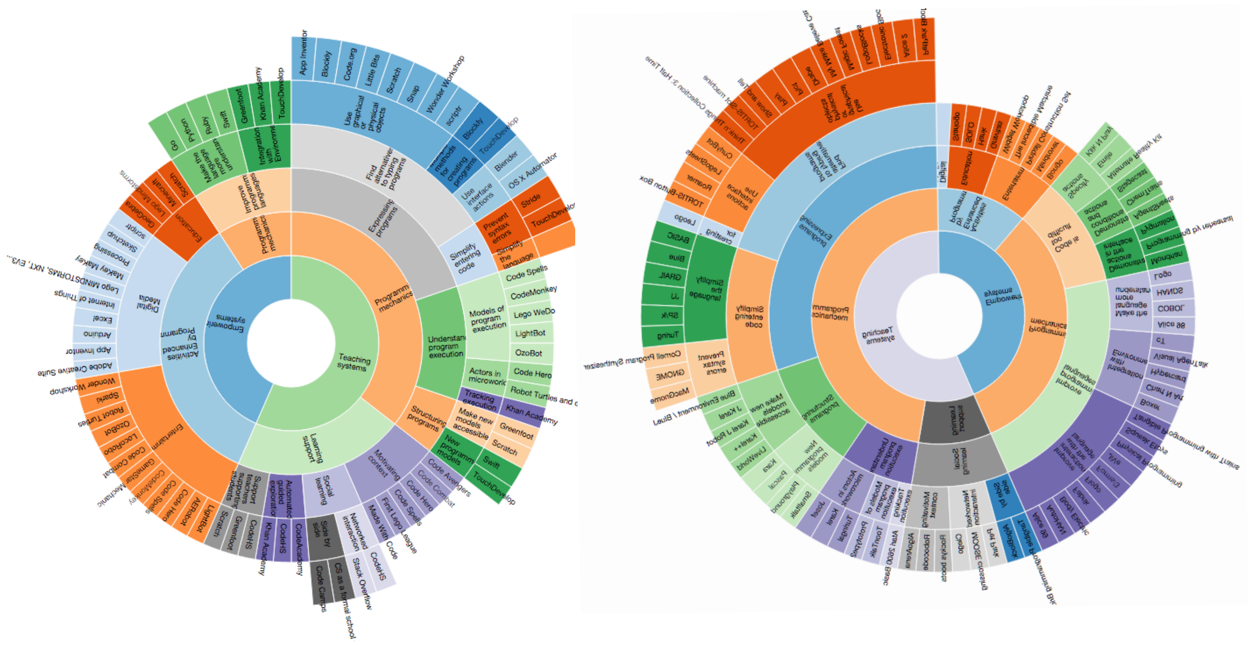


Figure 1. Changes in the landscape of tools for teaching programming, 2005-2016

Keywords

computational literacy, design, programming languages, computational thinking, educational technology

Introduction

The last decade has seen a surge of popular interest in learning to program, driven by the ever-deepening penetration of computers and mobile devices into our lives, an identification of programming as an important vocational skill, and celebrity endorsements going as far as the president of the United States. In response to this demand, there has been a proliferation of new programming languages and environments designed to teach programming to novice programmers. In 2005, Kelleher and Pausch published a taxonomy of languages and environments designed to teach programming, categorizing them in terms of how they sought to assist learners. This article extends Kelleher and Pausch's taxonomy to the current field of pedagogical programming languages and environments. We use our review as an indicator of which culture of programming is becoming more prevalent--shifts in the distribution of pedagogical approaches point to changes within the culture of computing and to the changing role of computing in the broader culture.

Despite these new tools and their many antecedents, it has always been hard to become a proficient programmer. Programming is a literacy demanding skills at many levels, spanning fluency with the symbols and syntax of programming languages, to practices such as structuring programs and debugging, to socialization into the community of programmers and development of a computational imagination. The earlier taxonomy defined programming narrowly, as "the act of assembling a set of symbols representing computational actions," allowing programmers to "express their intentions to the computer" and "predict the behavior of the computer." (p. 83) In the past decade, as computing has become a dominant medium in education, entertainment, journalism, and work, it has become more important to treat programming as a cultural practice, and to differentiate the levels of skills needed to be literate in the medium. The following three activities illustrate computational literacy at different levels:

- Writing a program to control your household thermostat.
- Hiring a programmer to change your newspaper's commenting policy.
- Discussing the tradeoffs between freedom and security in a cyber security law.

The pedagogical tools to facilitate mastery of these activities will differ not just in methods, but also in aims. A second goal of this article, therefore, is to taxonomize the components of computational literacy, and to sort the tools for learning to program according to the (often implicit) computational literacy activities they aim to teach their users. Framing teaching tools in terms of the forms of literacy for which they provide affordances will allow us to critique the shallow vocationalism underlying some efforts to "teach kids to code," to identify projects that have the potential to be socially transformative, and to suggest new directions for designers of educational programming languages and environments.

Comparing Pedagogical Approaches: 2005-2016

Kelleher and Pausch's taxonomy begins with a high-level division between "teaching systems," tools designed to serve as an intermediate step toward full-powered programming languages and "empowering systems," which aim to empower users directly, making full-powered languages unnecessary. The taxonomy then divides and sub-divides domains of barriers to programming. For example, most of the "teaching systems" address "mechanics of programming," which is divided into "expressing programs," "structuring programs," and "understanding program execution." The taxonomy then divides into groups of programs which take similar approaches to removing or bypassing the barrier. For example, in addressing the challenge of "expressing programs," some tools "simplify the language," while others "prevent syntax errors." After surveying the current field of languages and environments designed to teach programming, we organized them using Kelleher and Pausch's original taxonomy. The diagrams below show the 2005 taxonomy and its 2016 update. (See Appendix I for descriptions of the languages and environments taxonomized.)

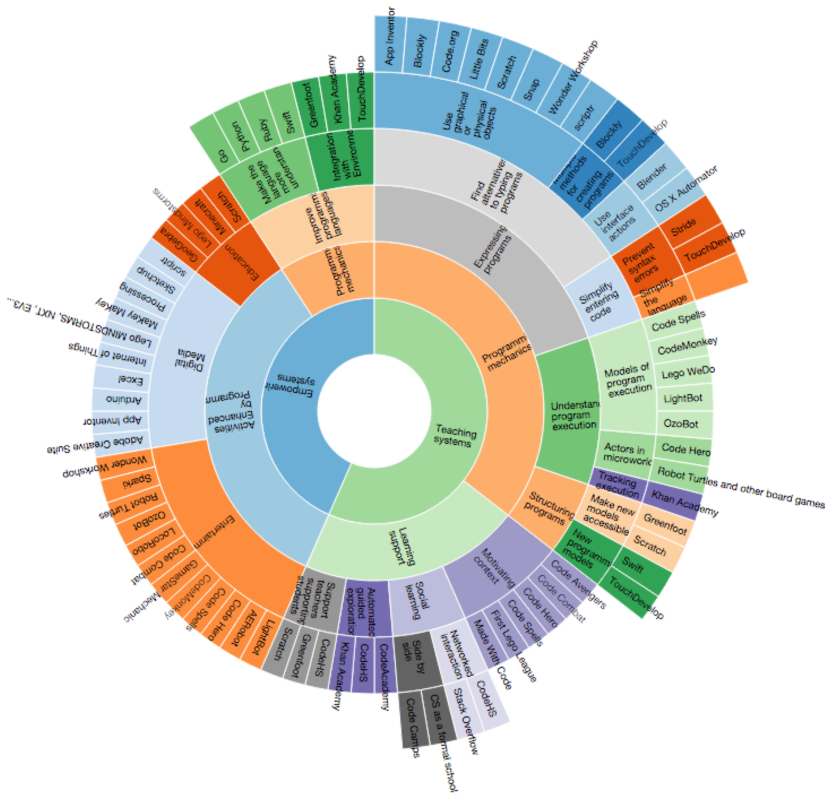


Figure 2. Kelleher and Pausch's taxonomy (2005) (chrisproctor.net/research/teachinglanguages/2005)

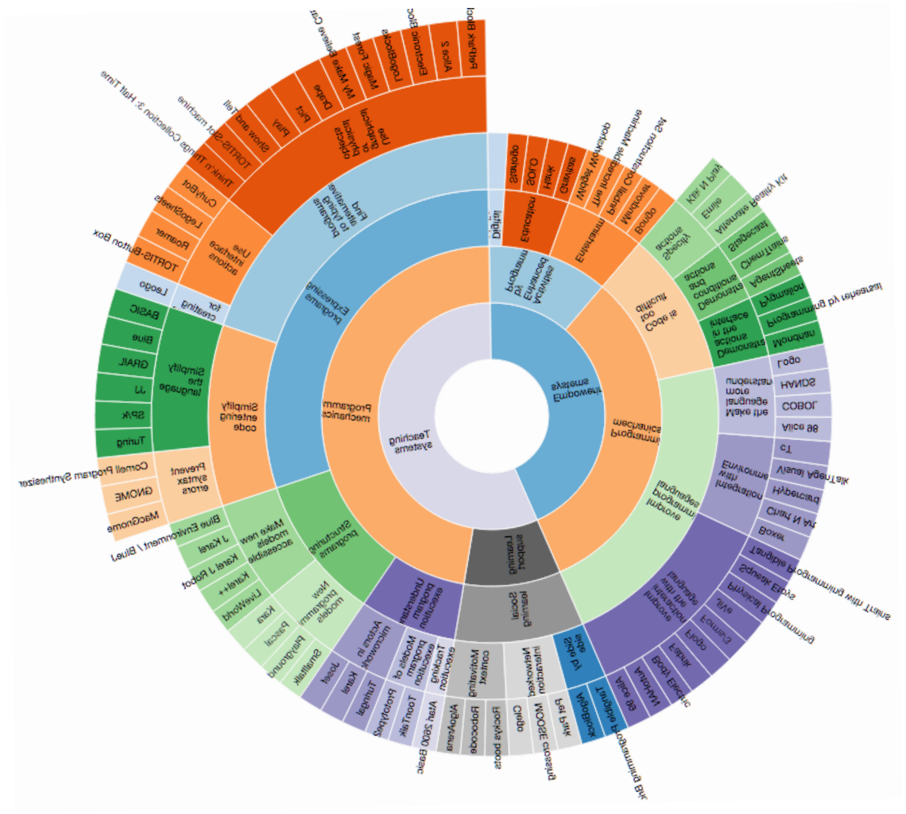


Figure 3. Updated taxonomy (2016) (chrisproctor.net/research/teachinglanguages/2016)

Comparing these two snapshots, with an interval of a decade, reveals similarities and some clear shifts. Emphasis continues to be put on helping learners overcome (or circumvent) the mechanics of programming; block-based programming remains a dominant paradigm for allowing users to express programs using affordances which prevent the possibility of some classes of syntax errors. In contrast, there are fewer efforts today to simplify the process of entering code by simplifying the language or preventing syntax errors. This suggests that two dominant strategies have emerged: teaching languages either avoid textual languages entirely, or engage with them fully. Two other significant changes within the category of teaching systems are the growth of tools that model program execution and the growth of social learning support. Over the last decade, rich computer graphics have become much more pervasive; it is now technically easier to model the execution of a program by animating blocks as they are executed, or by demonstrating the effect of a program via an actor in a microworld. Similarly, with the increasing universality of networked computers, it is much easier to incorporate social support for learning into a programming environment. Components of identity such as race, gender, and language have been identified as important factors in teaching programming, and sources of marginalization in the dominant culture of programming; some efforts at teaching programming are explicitly focused on supporting marginalized identities. One final trend to note in the category of teaching systems is the use of gamification to provide a motivating context. Several projects repurpose familiar game environments, such as the dungeon crawler role-playing game, so that the player scripts the avatar's actions rather than controlling them directly.

The other broad approach identified by Kelleher and Pausch, "empowering systems," has undergone even more significant shifts. General-purpose programming languages have become more expressive, user-friendly, and designed to be easily learned--an important language characteristic as the rate of technological change accelerates. One core task of any software developer is keeping up to date with new paradigms, patterns, languages, and frameworks. Consequently, there are now many more general-purpose programming languages included in the taxonomy. The category of "activities enhanced by programming" has also grown dramatically. The cultural activities of education, entertainment, and media production have all shifted toward the digital medium; participation in these activities increasingly demands the ability not just to write but to code. It is therefore not surprising that languages and environments designed to teach people how to program would do so in the context of empowering them to participate in cultural activities.

While Kelleher and Pausch's taxonomy serves as a useful lens for broad generalizations across time, this analysis would not be a strong basis for more specific comparisons. Partly this is due to a lack of methodological precision: we are not confident that either taxonomy was an exhaustive survey of the field. More fundamentally, we found that the taxonomy's categories, as well as the boundaries of its domain, are no longer stable. Three categories of ambiguity impede the task of taxonomizing languages and environments designed to teach programming: First, as the field has matured, new projects have multiple goals and draw on multiple strategies for making programming more accessible. For example, Microsoft's TouchDevelop provides a block-based interface, allows the user to switch to a text-based interface, allows the user to specify functionality with a graphical interface, dynamically points out syntactical errors, is focused on digital media production (especially games), and has social sharing built into the development environment. The decision of where to place TouchDevelop becomes quite arbitrary.

A second ambiguity lies in determining whether a programming language or environment is primarily designed to make programming more accessible--either by acting as a "teaching system" or as an "empowering system." User interface, user experience, and explicit teaching have become primary design consideration of many new programming languages. Yukihiro Matsumoto, the designer of Ruby, described his central goals for the language as to "help every programmer in the world to be productive, and to enjoy programming, and to be happy." (2008) Similarly, Guido van Rossum, the creator of Python, wrote, "Our aim is provide tools to support users when they are learning programming and when they are employing those skills in their homes and offices." (1999) As these and other new general-purpose languages have become popular languages for teaching, it has become difficult to identify a distinct subset of programming tools which are designed for teachability or accessibility.

Finally, as our dominant cultural medium has become computational and interactive, it has become difficult to distinguish programming from non-programming. For example, Adobe Photoshop can be viewed as a graphical user interface for composing matrix operations on a raster of pixels; users can apply computational concepts such as saving a sequence of commands for later use, defining new functions, and controlling the flow of data from one function to another. It is possible to explicitly script Photoshop using one of several mainstream programming languages. Should Photoshop be considered a programming environment? If so, similar cases can be made for Microsoft Excel, Facebook, Twitter, email, all our apps, and many of the new appliances we use at home and work ranging from photocopiers to Roombas to Internet-of-Things light bulbs. Computational culture has gone mainstream and code is becoming a literacy. While these shifts reduce the utility of Kelleher and Pausch's taxonomy, they underscore the importance of systems which make computation accessible, and therefore the need for making sense of how tools can help.

A Taxonomy Based on Literacy Aims

The original taxonomy, which organized tools according to how they support learning programming, depends on a definition of programming which is no longer adequate to describe the range of cultural practices that now comprise our interactions with computational media. We propose using DiSessa's analysis of the structure of literacy to recast programming as computational literacy, and therefore as a means of bringing more specificity to what we mean by teaching programming. DiSessa (2000) defines literacy as “a socially widespread deployment of skills and capabilities in a context of material support (that is, an exercise of material intelligence) to achieve valued intellectual ends.” (p. 19) DiSessa identifies three subsets of skills which comprise literacy: First, the material pillar involves “external, materially-based signs, symbols, depictions, or representations” which afford “particular modes of mediated thought.” (p. 6) When we read, the written word mediates our thought, permitting us to grapple with substantial ideas, provoking us to make undiscovered connections between ideas, and framing points of view, among other functions. This array of mental capacities forms DiSessa's second pillar, the cognitive pillar of literacy. Finally, the social pillar names the social niches and genres (socially-constructed patterns of meaning) which share a basis on a representational form. Socially-constructed text-based meaning implies shared expectations for readership, authorship, context, and form.

We can identify computational activities that correspond to these pillars as well. For example, writing a website or debugging a script primarily involve engagement with code, the basic material form of computation. The cognitive pillar is dominant when we use computational thinking to restructure a problem or develop a model of a situation. Similarly, collaborating on GitHub or playing multiplayer online role-playing games are examples of computational literacy which engage primarily with the social pillar. These three pillars enable a categorization of tools for enabling computational literacy according to their literacy aims, or what they aim to enable users to do. These aims may be explicitly stated or they may be implicitly present in the affordances provided to users.

We can further refine our analysis by distinguishing between literacy aims which assume a consumer role and those which assume a participant role. These roles correspond somewhat to the practices of reading but not writing (consumer) and both reading and writing (participant). However, critical reading goes beyond the more passive consumer role, while heavily-scripted participation, lacking a mechanism for creating new meanings, can be most appropriately identified as the consumer role. As with the three pillars of literacy, it will not be possible to draw a clean line between consumer and participant roles; for example, the practice of copying and pasting code written by others falls somewhere between consumer and participant. (Perkel, p.4) Nevertheless, it is important to distinguish roughly between literacy aims which assume the consumer role and the participant role. Promoting a more participatory "remix culture" or "read-write culture" has been a central commitment of media activists such as Lawrence Lessig (2008,

p. 28) and progressive educators whose goal is student empowerment. Drawing on the three pillars of literacy, and the two broad literacy roles of consumer and participant, we can pose criteria for a new taxonomy based on a tool's literacy aims:

- **Material (Consumer):** Does it aim to teach users how to interact with code? Does it provide affordances for learning to interact with code?
- **Material (Participant):** Does it provide affordances for or serve as a tool for using code to create new things?
- **Cognitive (Consumer):** Does it aim to develop recursive thinking, modularity and abstraction, heuristics, or other "mental tools" from the field of Computer Science? (Wing 2006) Does it provide affordances for learning to think in terms of computation?
- **Cognitive (Participant):** Does it provide affordances for or serve as a tool for applying computational thinking in original and personally-meaningful ways?
- **Social (Consumer):** Does it aim to teach users how to interact with computational artifacts created by others, or provide affordances for doing so?
- **Social (Participant):** Does it provide affordances for or serve as a tool for personally-meaningful computational expression? Does it support users in active, critical participation in computation-based communities?
-

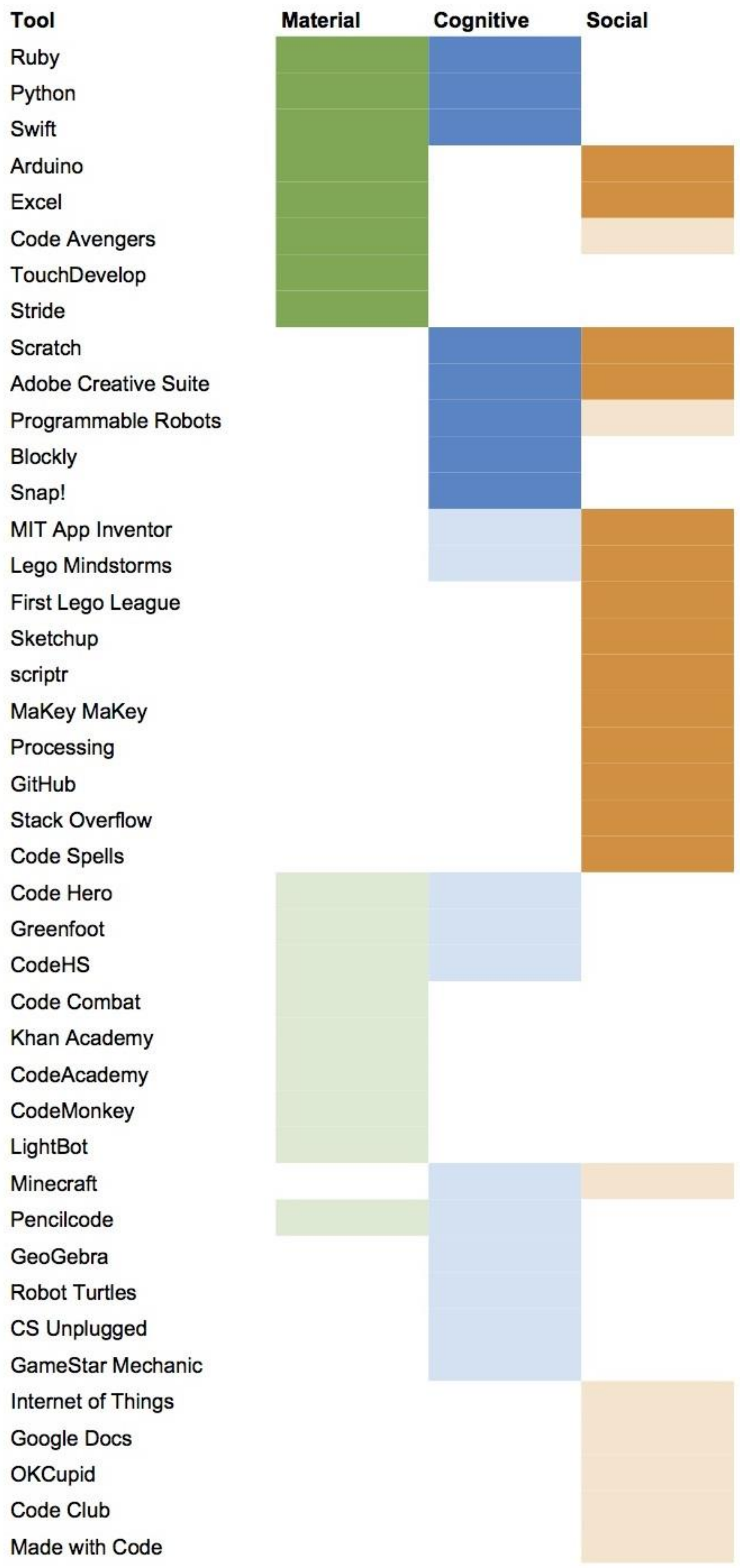
Using these questions, we re-taxonomized the tools for teaching programming (now broadened to include tools for enabling computational literacy); the results are presented in Figure 4. While most of the tools engage most strongly with one literacy, many also have literacy aims spanning more than one of the pillars.

The Future of Teaching Programming

This new taxonomy of the tools used for enabling computational literacy should be of use to both users and designers. The new taxonomy allows users (both learners and teachers) to select tools which are designed to support their own aims. The learning goals of a middle-school who wants to dabble with programming will be very different from those of a business owner seeking new ways to model the flow of her inventory, or a policy-maker who wants to consider how to apply ethical frameworks to the digital domain.

Teachers can now ask of these tools the questions that are important for other pedagogical curriculum and tools. Do they envision an empowered, participatory role for students? Is the social vision embodied by the tool's affordances congruent with the goals of the teacher or the school? For example, a programming curriculum which attends to nothing but the material pillar of literacy will be just as incomplete as an English curriculum which focuses exclusively on mechanics and which never offers students the opportunity to write anything they care about. In the field of secondary English education, teaching grammar in a contextualized, constructivist manner is a widely-accepted best practice. (Weaver, 1996, p.174-175) If this principle also hold when teaching programming, it suggests that tools with literacy aims spanning multiple pillars will be most effective.

One clear implication for designers of tools for enabling computational is the necessity of considering the literacy aims to be supported. Many of the tools we analyzed in this study had no clear explicit or implicit literacy aims; they do not ground user interaction in assumptions about why the user is learning to program. This trend corresponds with the popular demand that children should learn to program without a clear rationale for why children should learn to program, or with vague justifications based on future employability.



Tool	Material	Cognitive	Social
Ruby	🟩	🟦	
Python	🟩	🟦	
Swift	🟩	🟦	
Arduino	🟩		🟧
Excel	🟩		🟧
Code Avengers	🟩		□
TouchDevelop	🟩		
Stride	🟩		
Scratch		🟦	🟧
Adobe Creative Suite		🟦	🟧
Programmable Robots		🟦	□
Blockly		🟦	
Snap!		🟦	
MIT App Inventor		□	🟧
Lego Mindstorms		□	🟧
First Lego League			🟧
Sketchup			🟧
scriptr			🟧
MaKey MaKey			🟧
Processing			🟧
GitHub			🟧
Stack Overflow			🟧
Code Spells			🟧
Code Hero	□	□	
Greenfoot	□	□	
CodeHS	□	□	
Code Combat	□		
Khan Academy	□		
CodeAcademy	□		
CodeMonkey	□		
LightBot	□		
Minecraft		□	□
Pencilcode	□	□	
GeoGebra		□	
Robot Turtles		□	
CS Unplugged		□	
GameStar Mechanic		□	
Internet of Things			□
Google Docs			□
OKCupid			□
Code Club			□
Made with Code			□

Figure 4. Tools for teaching programming organized by literacy aims
(Light colors indicate a consumer role; Dark colors indicate a participant role)

Shifting the focus away from pedagogical tactics--strategies for supporting computational literacy--is not meant to suggest these are unimportant. Rather, this shift in focus suggests that this field is maturing. A decade ago, many tools were developed to help users overcome a particular barrier using a particular strategy. Today this remains an active field of research, but a set of dominant strategies or best practices seems to be emerging. Most importantly, considering pedagogical tactics in the context of a tool's aims allows designers to consider not just whether a pedagogical strategy will be helpful, but whether it will be helpful toward a particular end. As our society shifts toward a reliance on computation as a cultural medium, we will have to renegotiate what kind of society we want to have and the roles we want to have in it; this is an opportunity to address longstanding issues of marginalization and disempowerment. Designers of tools for interfacing with this new medium have both the opportunity and the responsibility to be clear about the roles for which they are preparing their users.

Appendix I: Notes on Tools for Teaching Programming

A table containing detailed notes on the programming languages and environments described in this paper are available at http://chrisproctor.net/research/teachinglanguages, where interactive versions of the visualizations included in this paper are also available.

References

DiSessa, A. (2000). Changing Minds: Computers, Learning, and Literacy. Cambridge, MA: MIT Press.

Kelleher, C. & Pausch, R. (2005). Lowering the Barriers to Programming: A Taxonomy of Programming Environments and Languages for Novice Programmers. ACM Computing Surveys, Vol. 37, No. 2, June 2005.

Lessig, L. (2008). Remix: Making Art and Commerce Thrive in the Digital Economy. New York: Penguin Press.

Perkel, D. (2006). Copy and Paste Literacy: Literacy Practices in the Production of a MySpace Profile. Informal Learning and Digital Media: Constructions, Contexts, Consequences.

Matsumoto, Yukihiro. (2008). Ruby 1.9. [Recorded Google Tech Talk]. Retrieved from https://www.youtube.com/watch?v=oEkJvvGEtB4

van Rossum, G. (1999). Computer Programming for Everybody: A Scouting Expedition for the Programmers of Tomorrow. [DARPA Funding proposal] Retrieved from https://www.python.org/doc/essays/cp4e/

Weaver, C. (1996). Teaching Grammar in Context. Portsmouth, NH: Heinemann.

Wing, J. (2006). Computational Thinking. Communications of the ACM, Vol. 49, No. 3, March 2006.

Inquiry Based Learning Project: a study of doing science with elementary public school students

José Armando Valente, *jvalente@unicamp.br*
Dept of Midia Studies, State University of Campinas (UNICAMP)

Abstract

The Inquiry Based Learning Project (Project ABInv) was developed with the objective to study the implementation of an inquiry-based learning approach, so that teachers and students could be engaged in "doing science", using features of the laptop in a 1-1 situation. The project was developed in three public schools and the results described in this article refer to the work developed in one of them. The project focuses on the professional development of teachers and administrators, to enable them to be able to integrate the laptops using inquiry based pedagogy across elementary school subjects.

The methodology used was action research. The university researchers worked with the teachers to encourage and assist them so they could appropriate the laptops according to this new inquiry based approach. It was an ongoing and in service training process which allowed the teachers to continue to work with their students, implementing investigations and using these experiences to debug theoretical and practical concepts about this pedagogic approach. The work developed by the teacher in the classroom followed her lesson plan, although the questions to be investigated were proposed and selected according to the interest of the students. To answer these questions the students had to do experiments, document the results and, based on the results, to come up with answers for the questions.

The results showed that students from first to fifth grade were able to perform investigations on various topics. The first grades were interested in how Brazilian native Indians produced dyes to paint their bodies, as shown in Figure 1a; second grades investigated the legs of the Tuiuiú bird, symbol of the Pantanal region – the fact that the legs bend backwards, different from humans, as shown in Figure 1b; students from the third grade studied the environmental conditions necessary to maintain an organism alive, such as the chrysanthemum plants (Figure 1c); fourth grades investigated organic and inorganic garbage decomposition over time and in different conditions, as Figure 1d; and students from the fifth grade were interested in the conditions for the growth of different plants, particularly bean, lettuce and onion seeds. They did a pilot experiment on the growth of beans (Figure 1e) to determine which type of soil was most adequate to use in their investigation.

Figure1a. Testing dye fixation on skin

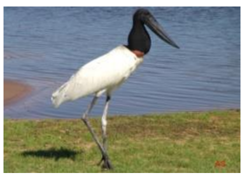


Figure 1b.The Tuiuiú bird

Figure 1c. Chrysanthemum s plant studied

Figure 1d. Garbage decomposition

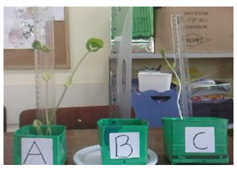


Figure 1e. Growth of beans

The Project promoted the development of the inquiry based learning approach, using the laptop's resources in different situations. It had a great impact on the teachers, students and the school community. Also it allowed the team of university researchers to work with schools and deeper their understanding about the use of technologies in a very innovate situation.

Keywords

Educational technologies; Inquiry-based education; ICT; Elementary school

Introduction

The idea behind the Inquiry Based Learning Project (Projeto Aprendizagem Baseada na Investigação – Projeto ABInv) came out of two situations that occurred almost simultaneously, although in distinct contexts. One was a literature review of the use of laptops in education when I found a document based on an interview with Alan Kay, the first researcher to propose, in 1968, that each child should have his/her own computer. Although Kay made this proposal before there even existed the idea of personal computers, when laptops finally began to be placed in the hands of students in the early 2000's, Kay pointed out that, different from what was anticipated, the resources available in these devices were not being used by the students to do science. Rather, they ware being utilized in a context in which the students continued to be taught about science. Students were not being given the opportunity to deal with uncertainties, with questioning, nor with incomplete or imprecise models, which could be debugged with the help of technologies, classmates, teachers or specialists. (The Book and the Computer, 2002). In general, the computers were being used to access already confirmed facts in order to reproduce, in large part, that which had already been accomplished with paper and pencil. This assertion is reaffirmed by various studies related to the implementation of laptops in several schools (Valente, 2011).

The other situation occurred when visiting schools that were incorporating laptops into their classrooms. When the teachers and students were asked about how they were using the technologies, in general, their response was that they were using the laptops to "do research". In fact, on observation of what was really going on, "doing research" meant using Google to search for information on the internet or in some data base. This use of computational resources is no different than that which was identified in other studies of 1-1 laptop implementation settings. For example, the evaluation of the "School, Teacher and Portable Computers Initiative" ("Iniciativa Escola, Professores e Computadores Portáteis") in Portugal, the most highly utilized resource in the portable devices were the search engines, 88.7% of use, more than any other software or resource in the laptop (Ramos, et al, 2009).

In this way, Project ABInv was proposed in the context of the schools participating in the One Computer for Student Project (Projeto Um Computador por Aluno - Projeto UCA) (Valente & Martins, 2012), through the Nucleus of Informatics Applied to Education (Núcleo de Informática Aplicada à Educação - NIED) of the State University of Campinas (Universidade Estadual de Campinas - UNICAMP), with the intention of creating the conditions for the students to utilize the laptops to develop activities devoted to "doing science", and, thereby, taking a step beyond the phase of simply studying scientific facts.

What is meant by inquiry based learning ?

The term, **"***inquiry based education***"** has primarily been used in the context of teaching science**.** Ramos, Giannella and Struchiner **(**2009**)** analyzed academic studies in the field of science education which adopt Design**-**based research **(**DBR**)** methodology for the development of educational environments mediated by information and communication technologies **(**ICT**).** The authors observed that **"**The theoretical construct most commonly adopted as a basis for these studies and to lead the development of the educational interventions was inquiry based education**".** Furthermore, according to the authors**:**

> "...the approach of inquiry based education is founded in the philosophy of John Dewey, which assumes that the child's curiosity is the starting point for the learning process. …and when put into practice, it includes a series of steps including 'formulating authentic and significant questions, planning tasks, collecting resources and information' (Krajcik, 2002, p. 411)". (Ramos, Giannella & Struchiner, 2009).

The results observed by Ramos, Giannella and Struchiner **(**2009**)** are in agreement with the work of Littleton, Scanlon and Sharples **(**2012**)**, editors of the book, **"**Orchestration Inquiry Learning", which is dedicated to the elaboration of theories and practical examples of inquiry learning. Among the various definitions of inquiry learning presented in this book, the one that is utilized by the National Science Foundation is most closely aligned with the work of Project ABInv:

> "Inquiry-based learning involves a process of exploring the natural and material world, and that leads to asking questions, making discoveries, gathering data to answer questions, and rigorously testing those discoveries in the search for new understanding."(Scanlon, Anastapoulou & Kerawalla, 2012, p. 8)

Another aspect that is noteworthy in the work of Ramos, Giannella and Struchiner (2009) is the fact that inquiry methodology is usually restricted to the area of science. In fact, as shown by Bagno (2010), investigation can be used in all fields, including language, art, history, and math. In addition, investigation does not have to be restricted to a particular class or to a specific grade, but can involve the entire school, a class, a group of students, or even an individual student.

Project methodology

The objective of Project ABInv was to help implement the inquiry based learning approach in several schools participating in the Projeto UCA, and to study the impact of this new approach on the use of the laptops, on the curricular activities, and on the students' education. The project focused specifically on the professional development of teachers and administrators, enabling them to integrate the laptops using inquiry based pedagogy in all the elementary school subjects.

The methodology used in this study was action research, a type of social research based on the observation of phenomena associated with actions and problem solving. It incorporates the researcher as an active participant, one who is involved with the subjects in a collaborative and participative fashion (Thiollent, 2004). The researchers worked with the teachers and students to define the thematic proposals for the development of the investigations based upon the school curriculum. The role of the researchers was, primarily, to encourage and assist the teachers' development so that they could adopt a creative and pedagogic view of the appropriation of the laptops according to this new inquiry based approach.

The Project ABInv was developed in three schools in the State of São Paulo and two schools in the State of Pará. The following schools participated from São Paulo State: EMEF Elza Maria Pellegrini de Aguiar, located in the Parque D. Pedro II neighborhood in Campinas (SP); EMEF Dr. Airton Policarpo, located in the municipality of Pedreira (SP); and EMEF José Benigo Gomes, located in the municipality of Sud Mennucci (SP) in the Bandeirantes D'Oeste district.

The Project began in September of 2011. The computational infrastructure related to the development of the Projeto UCA, including servers and wireless network, had already been implemented. The schools had already received the laptops dedicated for use by the students and teachers. The teachers had already been trained in how to use the laptops through a capacity building course that was given by the NIED/UNICAMP researchers, together with the professionals from the Nucleus of Educational Technology (Núcleo de Tecnologia Educacionais - NTE) of the city of Campinas (Valente & Martins, 2012).

The professional development of the administrators and teachers related to Project ABInv was ongoing and in service, thus allowing for the teachers to continue to work with their students, implementing investigations and using these experiences to debug theoretical and practical concepts about this pedagogic approach. The details of this process of professional development have been documented by Baranauskas e Martins (2014).

The first activity consisted of a seminar in which the Project ABInv was presented to all of the teachers and administrators from the three schools. Subsequently, the teachers and administrators were free to choose if they wanted to participate or not. Each school was assigned a researcher from the University to support the professional development and elaborate the activities in the school.

The professional development process began with an activity in the first seminar, in which the participants were asked to register their spontaneous conceptions related to three questions: "What is science?" "What is research?" and "What is scientific method?" They were also asked to do the same activity with their students. This small experiment initiated the entire process of professional development by which gradually there was a debugging of notions about how to work with students to raise questions related to the curriculum; how to decide together with the students what is a good question to be investigated; and how to create practical, implementable experiments; how to collect data to respond to the questions under investigation; and how to analyze and present the results from the experiments in order to respond to the questions they raised. The students explored the mobility offered by the laptops, working in alternative special configurations within the classroom, in other internal spaces within the school, as well as spaces outside the school building. They were encouraged to develop collaborative projects, dividing up the tasks so that the experiments could be executed and the data recorded and analyzed.

The results were registered using video, photographs, as well as activities in the students' notebooks. The material produced in the schools, along with the material elaborated for the seminar discussions, were stored in a repository that was created as the Portal ABInv (2015**).**

Results

The results from the Project ABInv were recorded in the book, *AbInv – Aprendizagem Baseada na Investigação* (Valente, Baranauskas & Martins, 2014). Part 2 of the book, entitled, "The Dynamic of ABInv in the School" contains chapters written by the teachers jointly with the researchers who supported the activities in the respective schools. In this section, as part of the Project results, brief examples will be presented of the activities at the EMEF Dr**.** Airton Policarpo school involving students from each of the elementary grades**.**

1st grade – Theme*:* Indians in Brazil

The students were studying Brazilian native Indians. Various questions were raised about the life and culture of the Indians. The question that most sparked the interest of the students was how the Indians produced dyes to paint their bodies.

The Problem:To discover how to make dyes using natural elements.

The Experiment: To make dyes using seeds and plants to discover which solvent will best fix the color. One group of students decided to produce red paint using urucum seeds, and another group to make green paint using plant leaves. Figure 2a shows the students crushing the urucum seeds; Figure 2b shows the use of various solvents such as water, alcohol, and kitchen oil; and Figure 2c shows a student testing which of the dyes best fixes to her skin.

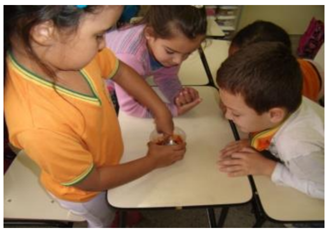


Figure 2a. Students crushing urucum seeds

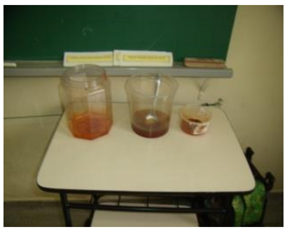


Figure 2b. Dyes in 3 types of solvents

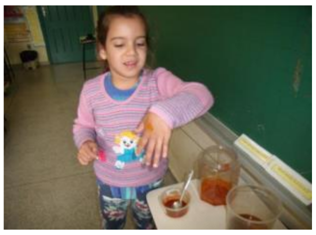


Figure 2c. Student testing dye fixation on skin

Results: The students concluded that the best dye was the one that used kitchen oil as solvent for the urucum seed, as shown in Figure 2c.

2nd grade – Theme*:* Animals from the Pantanal

The students were studying the Pantanal ecosystem and the topic that was of greatest interest was about the animals, especially the birds. The question to be investigated was about the legs of the Tuiuiú bird, symbol of the Pantanal region – the fact that the leg is so skinny and be able to support such a big bird, and the fact that it bends backwards, different from humans, as shown in Figure 3a.

The Problem: Why does the Tuiuiú's knee bend backwards when he walks?

The Experiment: Find answers to the question through a webquest. The students used their laptops to do their search. Figure 3b shows a student trying to understand how a bird's leg bends from a document found on bird morphology. Figure 3c shows the students building a prototype of the skeleton of the legs of the Tuiuiú.

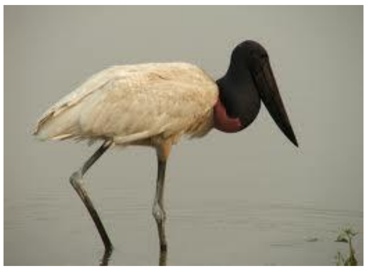


Figure 3a. The stance of the Tuiuiú bird

Figure 3b. Student trying to understand how the leg bends

Figure 3c. Students building the Tuiuiú prototype

Results: Despite various attempts such as observation of the skeleton, building the bird's legs in a wooden prototype, and observation of a mechanism, a portion of the students did not change their conviction that the Tuiuiú bends his leg backwards.

3rd grade – Theme*:* Astronomy

The students were studying the theme of astronomy, primarily the planets. Questions arose about the planets, in particular, if there is the possibility of life on them. In this way, the question to be investigated revolved around the environmental conditions necessary to maintain life.

The Problem: What are the environmental conditions necessary to maintain an organism alive?

The Experiment: To create three different environmental conditions – natural; a cold environment (refrigerator); a dark environment (the classroom closet) – and to observe what happens with the chrysanthemum plants (Figure 4a). The students were to evaluate the state of the plant in each of these three conditions (Figure 4b), register the physical state of the plants (Figure 4c) and conclude which condition was most favorable for the survival of the three plants (Figure 4d).

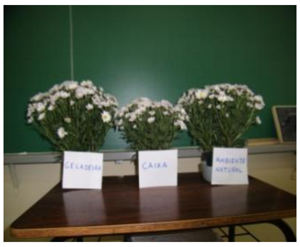


Figure 4a. Initial state of the Chrysanthemums

Figure 4b. Students evaluating a plant

Figure 4c. Register of the plants' state

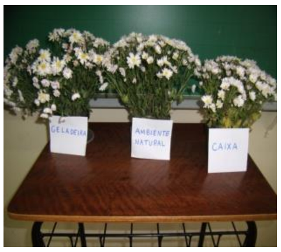


Figure 4d. The plants after one week

Results: The students' hypotheses in relation to the plant in the closet and the plant in the natural environment were confirmed. However, the plant that stayed in the refrigerator was a source of surprise for not having suffered under the low temperature. They later discovered that chrysanthemums can adapt perfectly to the cold.

4th grade – Theme: Garbage

The students were studying garbage. The questions that arose were in relation to organic and inorganic garbage; how garbage decomposes over time and if the conditions in which it is kept influences its decomposition.

The Problem: To discover the time it takes for organic and inorganic garbage to decompose in two environments: the soil around the school and a flowerpot in the classroom (Figure 5a).

The Experiment: To bury the following solid waste – a plastic bag from the supermarket, a page from a magazine, a banana, an orange, and an apple – in the ground and in the flowerpot, as shown in Figure 5b. After 30 days the students should observe the state of decomposition of the buried waste (Figure 5c) and record the data using the laptops (Figure 5d).

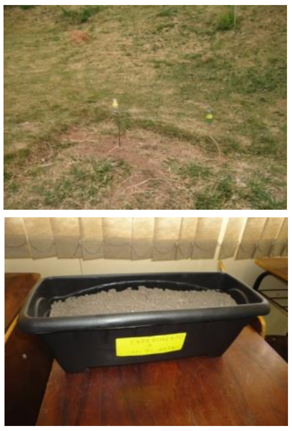


Figure 5a. Soil in the ground and the flowerpot in the classroom

Figure 5b. Students burying the waste in the ground and the flowerpot

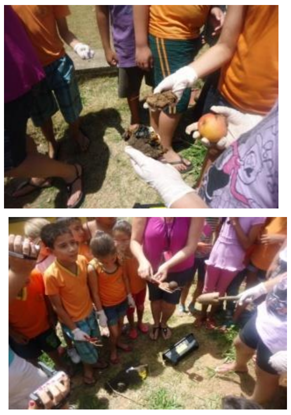


Figure 5c. Comparing the apple from the ground with that of the flowerpot

Figure 5d. Recording the results on paper and on the laptop

Results: The students concluded that the waste that was buried in the ground decomposed faster than the waste in the flowerpot in the classroom. They were surprised by the apple, which when unburied, remained practically intact, although dehydrated.

5th grade – Theme: Plant cultivation

The theme being studied was the growth of plants. The students were interested in understanding the conditions for the growth of different plants, particularly bean, lettuce and onion seeds, in different soil conditions. Before verifying the ideal conditions for plant cultivation and the growth of lettuce and onions, they decided to do a pilot experiment on the growth of beans to determine which type of soil to use.

Problem: which bean will grow first – in fertilized earth, normal earth, or in cotton (because it is softer and cozy)?

Experiment: To compare the growth of bean seeds planted in three different recipients: fertilized earth, normal earth, and cotton. The hypothesis was that the beans in fertilized earth would grow better because the earth is treated (Figure 6a). The students should follow the growth, measuring the plant's development (Figure 6b), recording in spreadsheets and graphs (Figure 6c), and verifying which condition is the most adequate for the growth of beans.

Figure 6a. Students planting the beans

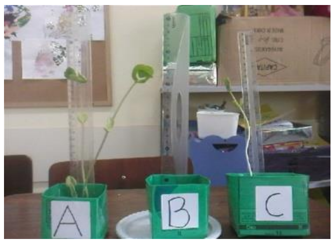


Figure 6b. Measuring the height of plants

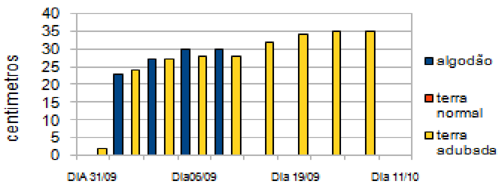


Figure 6c. Graph accompanying the growth of the beans

Results: They concluded that plant "A", with good earth (fertilized), promoted the best growth and, therefore, it would be used for planting the vegetable garden. The vegetable garden experiment consisted of observing the growth of the lettuce and the onion, comparing both processes: planting from seed and from seedlings in vases and in the ground.

These examples show that students of all grades at the EMEF Dr. Airton Policarpo elementary school were able to perform activities related to the inquiry based learning approach. This occurred according to the lesson plan proposed by each teacher, however, the questions that were investigated were proposed and selected according to the interest of the students. In this way, it was possible to prove some of the hypotheses, although many surprises emerged during the investigation process. This made for more realistic and challenging investigations, both for the students and for the teachers and researchers involved in the project.

Conclusions

The Project ABInv was designed to respond to the question of how to promote the process of change within classroom activities by way of implementing a pedagogic approach based on inquiry. The results indicate that this question was successfully answered.

First, the pedagogic approach of "inquiry based learning" was appropriated by various teachers in the three schools that participated in the Project. These teachers and administrators were able to work with a diversity of curricular subjects using this new approach. Second, a methodology was developed for how to establish the inquiry based education approach in the school, primarily in relation to providing in service training to teachers in a way that they can gradually change their theoretical and practical conceptions about what it means to do research. Third, a repository of investigative activities was created, containing examples of curricular activities that can be developed through inquiry based pedagogy and the utilization of laptops. The register in the Portal ABInv of the material elaborated during the activities contains examples that can be applied in classrooms. Fourth, the knowledge constructed over the course of the development of the Project ABInv enabled all of the participants to elaborate their own notions of the meaning of inquiry based education, reflecting, analyzing and incrementing the theories already existing and identified in the literature. For sure, the team of researchers has a far deeper and significant understanding of the educational approach based on inquiry and the role of the laptops in the development of these activities.

Finally, the Project ABInv promoted the development of a pedagogic approach that utilizes the laptop's resources in situations of inquiry based education. These situations were very stimulating and contagious, involving the students, teachers, administrators and members of the school community. It was truly pleasurable to be able to observe the involvement of the students who were not only studying scientific facts, but were doing research, using methods and techniques for generating scientific knowledge. This instigated the involvement of colleagues, teachers and parents who were interested in knowing how the experiments were progressing and who wanted to find out about the results obtained by the students.

Acknowledgements

This work was support by grants 550382/2011-9 and 306416/2007-7 from Conselho Nacional de Desenvolvimento Científico e Tecnológico,(National Council for Scientific and Technological Development - CNPq), Brazil. I thank the teachers and students from the schools who participated in the Project and the researchers Maria Cecília C. Baranauskas, Maria Cecília Martins, Maria Helena N. de Sá and Ann B. Valente for the collaboration during the development of the project.

References

Bagno, M. (2010). ***Pesquisa na Escola – o que é, como se faz***. São Paulo: Ed. Loyola, 24ª edição.

Baranauskas, M.C.C. & Martins, M.C. (2014).*ABInv Aprendizagem Baseada na Investigação – A Metodologia*. In ABInv – Aprendizagem baseada na investigação. Edited by J. A. Valente, M. C. C. Baranauskas and M. C. Martins. Campinas, SP: Unicamp/NIED, pp. 42 – 64.Available at: www.nied.unicamp.br/?q=livros.

Littleton, K., Scanlon, E. & Sharples, M. (2012).*Orchestration Inquiry Learning*. Londres: Routledge.

Portal ABInv (2015). *ABInv – Aprendizagem Baseada na Investigação*. Available at: www.nied.unicamp.br/abinv.

Ramos, J. L. et al (2009). *Iniciativa Escola, Professores e Computadores Portáteis*: *Estudos de Avaliação*. Lisboa: DGIDC- Direcção Geral de Inovação e de Desenvolvimento Curricular. Available at: http://erte.dge.mec.pt/files/@crie/1277481626_Estudo_Portateis_Junho2010.pdf.

Ramos, P.; Giannella, T. R.; Struchiner, M. A. (2009). *Pesquisa Baseada em Design em Artigos Científicos Sobre o Uso de Ambientes de Aprendizagem Mediados pelas Tecnologias da Informação e da Comunicação no Ensino de Ciências: uma análise preliminar*. Anais VII Enpec – Encontro Nacional de Pesquisaem Educação em Ciências. Florianópolis, 2009. Available at: http://posgrad.fae.ufmg.br/posgrad/viienpec/pdfs/1707.pdf.

Scanlon, E., Anastapoulou, S. & Kerawalla, L. (2012). *Inquiry Learning reconsidered: contexts, representations and challenges*. In: Orchestration Inquiry Learning, K. Littleton, E. Scanlon and M. Sharples. Londres: Routledge.

The Book and the Computer(2002). *The Dynabook Revisited* - a conversation with Alan Kay. Available at: www.squeakland.org/content/articles/attach/dynabook_revisited.pdf.

Thiollent, M. (2004). *Metodologia da pesquisa-ação*. São Paulo: Cortez, 13ª edição.

Valente, J.A. (2011). *Um laptop para cada aluno: promessas e resultados educacionais efetivos*. In O Computador Portátil na Escola, Edited by : M. E. B. Almeida and M. E. B. B. Prado, São Paulo: Avercamp, pp. 20-33.

Valente, J. A., Baranauskas, M. C. C. & Martins, M. C. (2014). *ABInv – Aprendizagem baseada na investigação*. Campinas, SP: Unicamp/NIED. Available at: www.nied.unicamp.br/?q=livros.

Valente, J. A. & Martins, M. C. (2012). *Preparing Teachers to Use Laptops Integrated to Curriculum Activities: the experience of One Laptop per Student project at Unicamp*. In Conference Proceedings of Constructionism 2012. Edited by C. Kynigos, J. E. Clayson and N. Yiannoutsou, August. pp. 250 − 259.

Integrating programming languages with web browsers

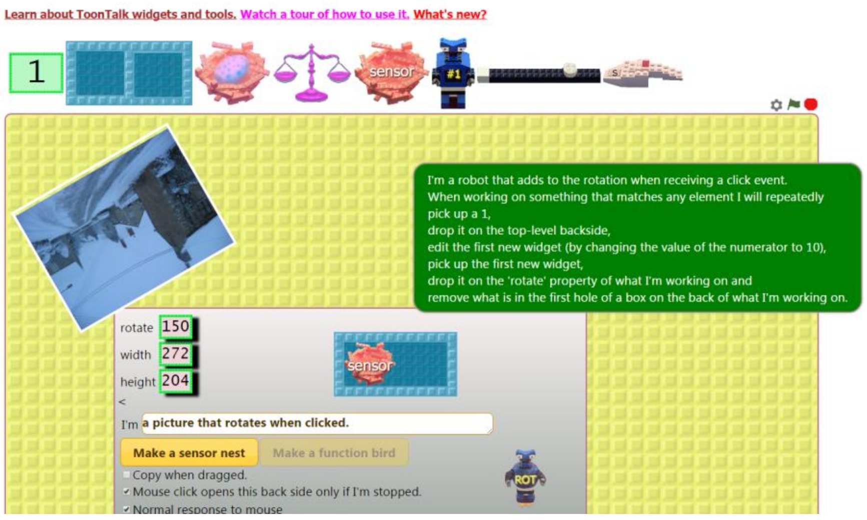
Ken Kahn, *toontalk@gmail.com*
Academic IT Services, University of Oxford

Abstract

Logo, Snap!, NetLogo, ToonTalk, and many other programming languages can run in modern web browsers without the need for plugins or installation. This paper is about the new possibilities this opens up: the integration with the wide range of functionality that browsers afford. Browsers support geometric transformations, animation, web storage, events, flexible styling, 2D and 3D graphics, drag-and-drop, multi-media elements, peer-to-peer communication, disability modes, cameras, microphones, and other sensors. Third-party libraries offer cloud storage, cloud publication, translation to over one hundred languages, and interfaces to social media sites.

A web interface to a programming language and environment can be implemented as web pages whose layout and styling can be customised for different users and contexts. Very different "looks and feels" can be created by users relying only upon HTML and CSS. Customised versions can be embedded into pages to produce interactive tutorials and documentation.

This paper describes ways that a programming language can exploit the web ecosystem. This is illustrated using ToonTalk Reborn, the web version of ToonTalk, as an example. ToonTalk Reborn is tightly integrated with browser elements, events, and styling. Any HTML element can be added to a ToonTalk web page and controlled by ToonTalk programs. Google Drive has been integrated to support sharing and publication of ToonTalk programs. Over one hundred language versions are available due to integration with Google Translate. Drag and drop to and from web pages and other applications supports sharing and saving programs, as well as importing media.


Learn about ToonTalk widgets and tools. Watch a tour of how to use it. What's new?
1
sensor
#1
S
sensor
Make a sensor nest
Make a function bird
ROT

I'm a robot that adds to the rotation when receiving a click event.
When working on something that matches any element I will repeatedly
pick up a 1,
drop it on the top-level backside,
edit the first new widget (by changing the value of the numerator to 10),
pick up the first new widget,
drop it on the 'rotate' property of what I'm working on and
remove what is in the first hole of a box on the back of what I'm working on.

rotate 150
width 272
height 204
<
I'm a picture that rotates when clicked.

- ☐ Copy when dragged.
- ✓ Mouse click opens this back side only if I'm stopped.
- ✓ Normal response to mouse

Figure 2. Making an image interactive in ToonTalk

Keywords

Web, ToonTalk, programming languages, HTML, DOM, constructionist environments, web programming

Opportunities in the Web Ecosystem

This paper takes as a starting point that programming languages and their programming environments can be implemented to run plugin-free in modern web browsers. Logo (github.com/inexorabletash/jslogo), Snap! (snap.berkeley.edu), NetLogo (netlogoweb.org), and ToonTalk (github.com/ToonTalk/ToonTalk/wiki) are examples[13] from the constructionist community. Nothing need be installed to use them. This is increasingly important as students and teachers lack administrator privileges. System administrators have many fewer security and maintenance concerns with browser-based programs than with executable programs. When a browser-based programming language is combined with cloud storage, users can easily move between computers and operating systems. This is all very important but this paper is about how there is much more that web browsers offer that could be exploited to make programming environments more powerful, more flexible, and more configurable. And yet this is rarely done.

Web Pages

Programming environments of web-based programming languages live on web pages. In most cases the entire page is devoted to the programming environment. Embedding the programming environment in only a part of a page can be very useful. The page can be a tutorial or part of a manual, and the reader can interactively explore the aspect of the language being discussed *in situ*. For example, the Joy and Beauty of Computing MOOC (www.edx.org/course/beauty-joy-computing-cs-principles-part-uc-berkeleyx-bjc-1x) experimented with embedding instances of Snap! in lesson pages. Embeddability enables the creation of more effective interactive learning resources as well as enabling learners to create active essays which blend writing and interactive programs.

Ideally the programming environment can be customised for different embedded uses. Subsets of the full functionality of the language could be provided when appropriate. Half-baked programs (Kynigos, 2007) might be embedded in lessons. The programming interface may be altered to use less screen real estate when sharing the screen with text or images.

If the language is implemented well, customisations should be performable by anyone with sufficient HTML and CSS skills. Different configurations of the programming environment could be designed for users of different ages or abilities (including those with disabilities). Prior to the emergence of web-based implementations of programming environments, such customisations would require knowledge of the language implementation's architecture and programming language. A well-designed web-based implementation with a clear separation between structure (HTML), style (CSS), and interactivity (JavaScript) makes radical customisation possible by many. This could be used to implement version tuned to learners with disabilities or different skill or age levels. In some cases this requires only changes to the CSS styling.

HTML Elements

Web pages consist of HTML elements that include rich text, images, video, audio, links, and interface widgets such as buttons, input areas, and menus. Web-based implementations of programming languages *use* these elements but don't provide interfaces for users to program these elements directly. Typically HTML elements can only be given behaviours using JavaScript. In principle browser-based languages could be implemented to give behaviours to elements. Logo, Snap!, and NetLogo have not been implemented to do so and cannot be used by students to create general interactive web pages.

[13] Scratch runs in browsers that support Flash and shares some of the web features discussed here but Snap! being a pure web-based implementation of a superset of Scratch is a better example for the purposes of this paper.

CSS Styling

If user programs incorporate HTML elements, then the CSS styling of those elements should be controllable by user programs. If, for example, the CSS properties *left*, *top*, *width*, and *height* are available to user programs, then those programs can alter those properties to move or scale an element. Hundreds of element properties could be made available (www.w3.org/TR/DOM-Level-2-Style/css.html#CSS-ElementCSSInlineStyle). Some properties describe transformations such as rotation or skew. Others define animated transitions. A programming language that provides an interface to element style properties gives the programmer a powerful tool for changing an element's size, colour, font, and much more. The implementers of a constructionist language can ride the wave of well-implemented new browser features with little effort.

DOM Events

A wide range of browser events are available for JavaScript programs to respond to. A web-based implementation of a programming language that provides an interface to these events enables its users to construct programs that respond to the keyboard, mouse, touch devices, media streaming, sensors and changes to interface widgets.

Browsers also support *custom events*. A programming language implementation can define new events that are appropriate for the language's computation model. For example, turtles could signal events such as reaching the edge of the screen.

Drag and Drop

Browsers can respond to drag and drop events. A web application can be programmed to respond to drops of files, URLs, rich text, and custom items. A dropped file can be an image, audio, video, or rich text. This can provide a very powerful easy-to-use means for users to import resources they find on the web into their programs. If a web-based programming language implements drag and drop of its primitive widgets then they can be dragged between browser tabs or windows. Many applications (e.g. Microsoft's WordPad) also support this so widgets can be dragged to a text file, saved or emailed, and later dragged into another instance of the programming environment. A language's support for drag and drop between applications can of course also be used within the language's programming environment and made available to programs constructed in the language.

Canvas and WebGL Graphics

Modern browsers support the construction of *canvases* where a full range of 2D and 3D graphics primitives are available. Browsers typically implement these primitives using the computer's graphics hardware and hence are usually very high performance. While very powerful, many find these APIs difficult to master. A programming language might still benefit beginning students by integrating this functionality since beginners could use higher-level, conceptually simpler primitives implemented by user code that interfaces to canvases directly.

Devices and Sensors

Web browsers provide access to devices and sensors. For example, they can access the camera to integrate photos and videos in a user's program. Web pages that attempt to use devices such as the camera or microphone ask permission before proceeding. While this can be awkward, it provides the user of the program assurance that their privacy will be respected.

Security Issues

Administrators, parents, and even students are rightly concerned about the possibility that they will run some malware that might cause damage, steal secrets, or violate someone's privacy. While not perfect, browsers are probably more secure than most other programs. Much of today's economy relies upon people trusting their browsers. Programming languages implemented in a browser are just as secure as the browser itself. A consequence is that web-based programming languages are severely limited in what operations on the file system are permitted.

There are efforts to define a secure subset of JavaScript (Miller et al, 2013). If a programming language is implemented in this subset then it opens up the possibility that programs written in that language can securely cooperate.

Storage, Sharing, and Publishing

The consequences of the limitations on access to the file system is somewhat remedied by the ubiquity of cloud storage. Instead of relying upon the local machine's file system, users can rely upon their Google Drive, Dropbox, or OneDrive cloud storage. These services provide APIs that language implementers can use so that users need not concern themselves with the technical details of the system. Alternatively, the organisation behind the web-based programming language could provide their own server storage as, for example, the University of California does for Snap! users. When cloud storage is inappropriate, programs can be saved in the browser's *web storage*. It typically provides 50MB of storage that is private to each web application.

Cloud storage greatly simplifies sharing and publishing programs. If the user's own cloud storage is used they can keep it private, share it by URL access or user identities, or make it public to the world. And in some cases (e.g. Dropbox) a URL can be obtained of a public file that the browser will render as a web page instead of downloading it.

Web Workers

Web workers is a browser standard that enables programs to run concurrently on multi-core computers. This ability could be used by browser-based languages to enable users' programs to run significantly faster.

WebRTC

WebRTC (developer.mozilla.org/en-US/docs/Web/API/WebRTC_API) supports peer-to-peer communication between browsers. In addition to data, it supports streaming of audio and video. Currently it is supported by Chrome, Firefox, the Android browser, and Opera. Microsoft is working on a compatible variant but it is unclear if and when Apple's Safari will support it. It could be used to support collaborative programming and the construction of multi-player programs.

Browser Features

The interface of web browsers is familiar to billions of people. The difficulties of learning and using a web-based programming language can be reduced by relying upon the browser's interface. People already know how to navigate within and between web pages, how to zoom in and out of pages, how to bookmark pages, how to use browsers' spell checkers, etc. There is a price to pay, however: for example, right mouse button clicks bring up a browser-specific menu. This can be overridden but not without interfering with the normal operation of the browser.

Browsers maintain a history that users can navigate using back and forward buttons. Initially this history was limited to the history of web pages visited but there is now an API for programs to extend this. One could use it as an interface to an undo/redo functionality.

Third Party APIs

In addition to the functionality that browsers provide, there is a vast number of web services that can be accessed from browsers. Google, for example, provides cloud storage, translations, maps, data visualisations, real-time collaboration, communication with household devices and toys, and much more. A web-based implementation of a programming language can integrate any of these services.

There are many open data sources from government agencies and research labs (e.g. environment, astronomy, or climate). A programming language can provide abstractions that hide many of the technical details involved in accessing these data sources.

ToonTalk Reborn

ToonTalk (Kahn, 1996) was conceived in 1992 with the goal of using video game technology to bring the most advanced computer science research in programming languages to a wide audience. The core idea is that each abstraction in the computation model has a corresponding *concretization* that captures the fundamental properties of the computational abstractions. These concretizations can be playful and understood in their own terms, even by very young children. The computation model chosen was *concurrent constraint programming* (Saraswat, 1993) which provides a safe well-grounded way of introducing concurrency, communication, and synchronisation to programs. The other core idea underlying ToonTalk is the construction of programs by demonstration followed by the removal of details for abstraction (Kahn, 2001).

From Desktop to Web

As described in (Kahn, 2014) ToonTalk Reborn is a re-implementation and re-conceptualisation of desktop ToonTalk for the web. The desktop ToonTalk relies upon many of Microsoft Windows APIs. It uses DirectDraw for 2D graphics, DirectInput for input events, DirectSound for audio, and DirectPlay for peer-to-peer communication. Consequently desktop ToonTalk only runs on Microsoft Windows and exists only as a large executable that must be installed and trusted.

Desktop ToonTalk is implemented in C++ and performance was a critical requirement due to its animated game-like interface. A web-based implementation was inconceivable until the relatively recent adaption of HTML5 and high-performance implementation of JavaScript by modern browsers that makes rich and powerful browser-based programming systems feasible.

Just a Web Page

Any web page can be turned into a ToonTalk Reborn page by including a few lines of HTML that specifies the necessary JavaScript and CSS. ToonTalk widgets (numbers, boxes, birds, nests, scales, robots, and elements) and work areas can be anywhere on the page. The page that implements the default programming environment includes a single work area and an array of widgets that acts like a tool palette (see Figure 2). Straight-forward edits to the HTML of this page can produce very different environments with different tool palettes and layouts.

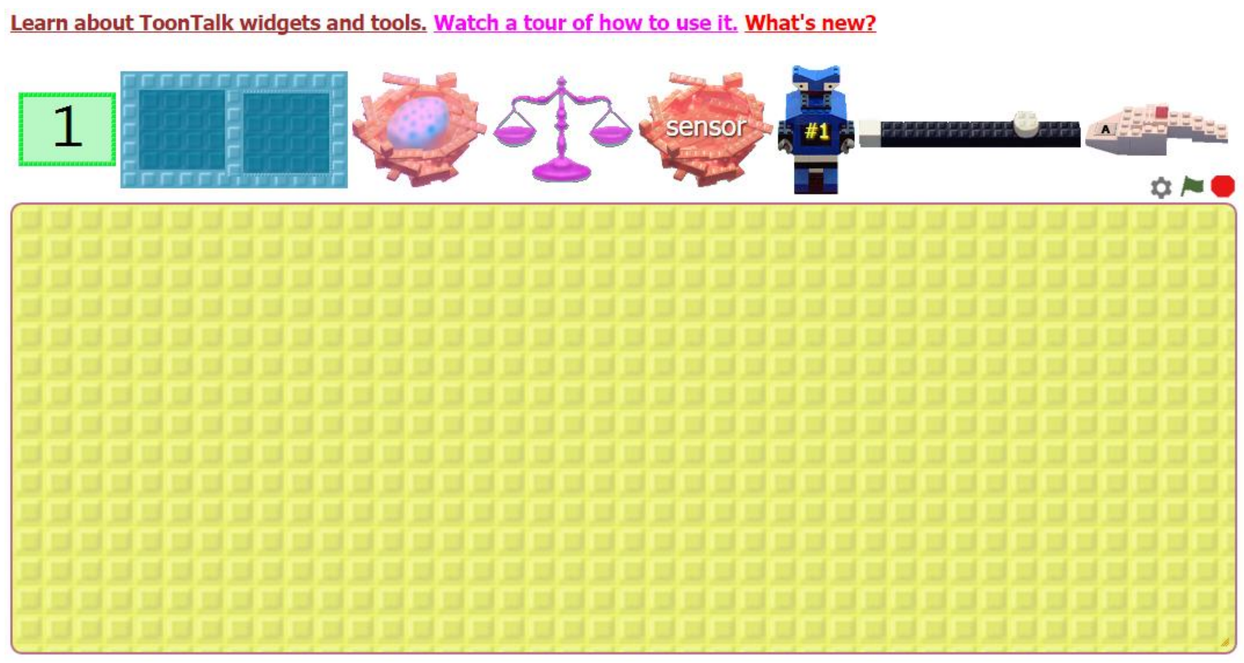

Learn about ToonTalk widgets and tools. Watch a tour of how to use it. What's new?
1
sensor
#1
A

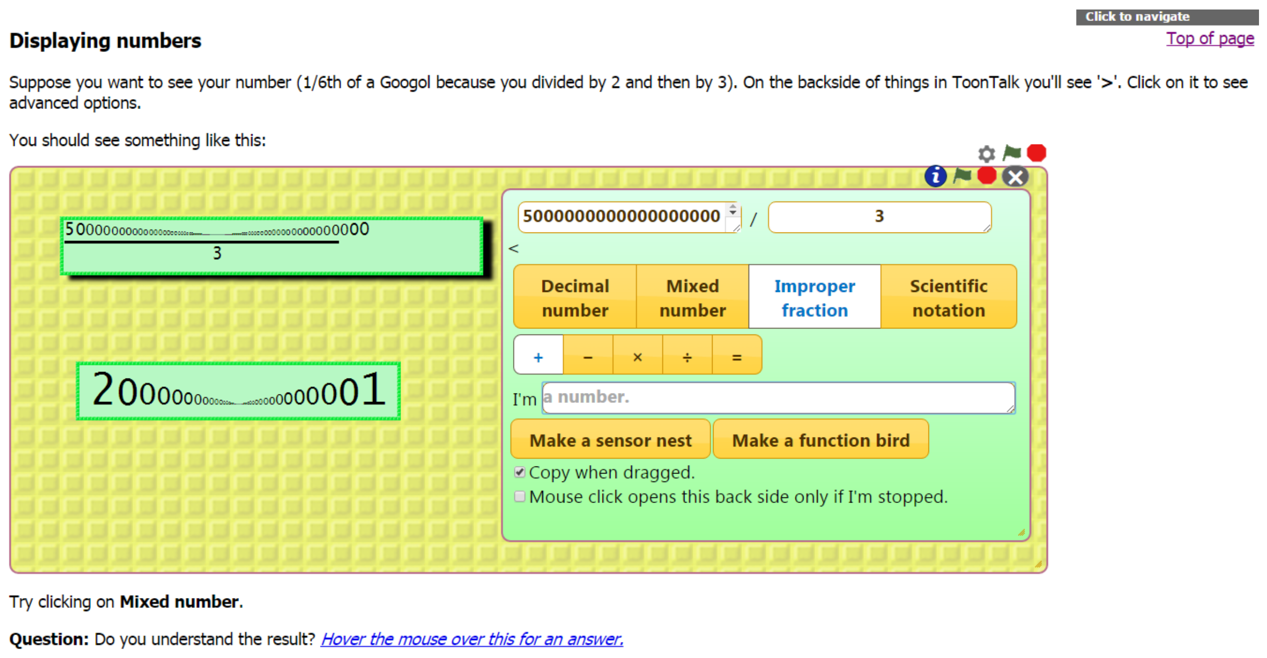
Figure 3. The default ToonTalk Reborn programming environment

A typical ToonTalk Reborn manual page contains many live instances of the widget being documented. Each instance is customised as needed. Small specialised workspaces are available on manual pages to facilitate the exploration of the widget being documented (see Figure 3).

The ToonTalk Reborn CSS style sheet defines more than the colours and fonts of the interface. Most of the widgets can be animated in multiple ways (e.g. a bird hatching, a bird flying in any of eight directions, a bird waiting to be given something). The CSS defines the images and how they are animated. A graphic designer could leave all the JavaScript implementing ToonTalk untouched and create a different CSS with different imagery and animation transitions to give the system a very different 'skin'.

HTML Elements in ToonTalk

Any HTML element can be added to a ToonTalk program. One can drag images, sounds, videos, plain text, rich text, links, and any other HTML element from any web page to a ToonTalk work area. Element widgets, like all other widgets, can be placed in boxes, given to birds, copied, etc. Element widgets can be composed hierarchically (e.g. ball and paddle images can be added to a Pong background image).



Click to navigate

Top of page

Displaying numbers

Suppose you want to see your number (1/6th of a Googol because you divided by 2 and then by 3). On the backside of things in ToonTalk you'll see '>'. Click on it to see advanced options.

You should see something like this:

Try clicking on **Mixed number**.

Question: Do you understand the result? *Hover the mouse over this for an answer.*

Figure 4. A portion of the manual page describing numbers

The backside of a widget contains a control panel for that widget. An element widget control panel includes access to the CSS style of that element. One can select style properties including those that control the location, dimensions, colour, font, and transformations (rotation, skew, etc.). An property widget of an element widget is a number widget or plain text widget with two additional aspects: (1) it displays the up-to-date value of that property and (2) changes to its value become changes to its style (see Figure 4).

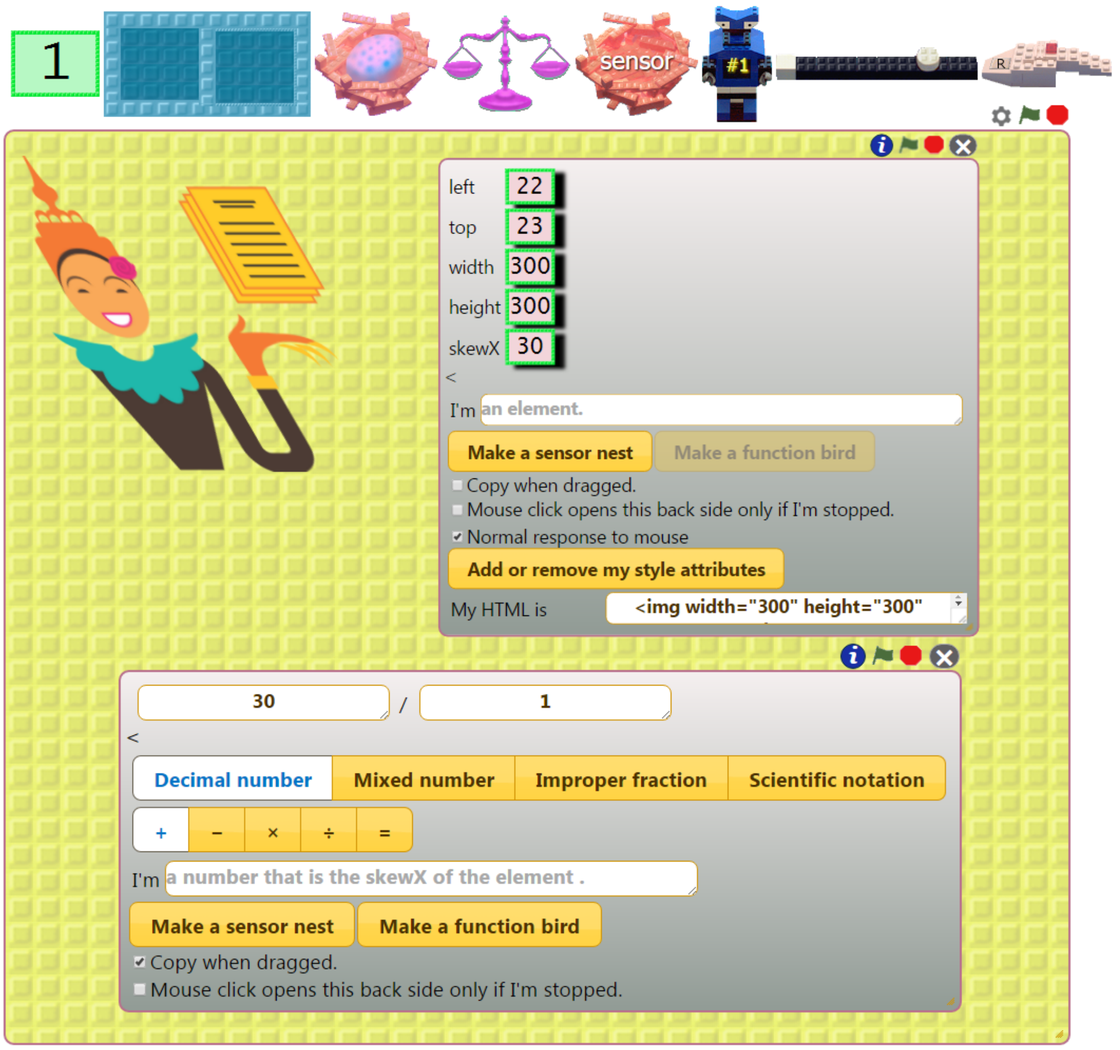
DOM Events in ToonTalk

Communication and coordination in ToonTalk is provided via the metaphor of birds who take messages (widgets) to their nest. Robots who encounter such nests wait for something to arrive before proceeding. ToonTalk sensors are represented by nests whose bird is given an event message "off screen" by the browser. For example, a sensor could be told to listen to 'keydown' events; when they occur, the bird will take a plain text element widget with the description of which key was pressed to the sensor nest. Sensors can also be associated with a widget to, for example, respond to 'click' events on the widget. Custom events are also dispatched in addition to the DOM events that are standard to all browsers. For example, an event is triggered when a ToonTalk widget is added to another widget.

The Many Uses of Drag and Drop in ToonTalk

HTML5 supports the transfer of data when a user drags and drops items. ToonTalk Reborn uses this to transfer the JSON of a widget so the widget can be reconstructed in another browser running ToonTalk. Some applications support the drop of a ToonTalk widget by inserting the text of the JSON. For example, Microsoft WordPad can be used to both receive and send ToonTalk widgets. Selecting the JSON string and dragging it to ToonTalk causes the widget to be reconstructed. This can be used as a way to "manually" save and share ToonTalk programs.

ToonTalk is also able to receive drops of plain text, files (images, audio, video, and text), elements, and URLs. For example, the image in Figure 4 was creating by dragging an image from the Constructionism 2016 website to ToonTalk.



left 22
top 23
width 300
height 300
skewX 30
<
I'm an element.
☐ Copy when dragged.
☐ Mouse click opens this back side only if I'm stopped.
☑ Normal response to mouse
My HTML is <img width="300" height="300"

30 / 1
<
I'm a number that is the skewX of the element .
☑ Copy when dragged.
☐ Mouse click opens this back side only if I'm stopped.

Figure 5. An image element widget, its backside, and the skewX property widget backside

Saving and Publishing

ToonTalk Reborn, if granted permission, will use the user's Google Drive to save programs. Optionally, they are updated automatically as the user changes his or her program. The files are initially private but the user can share or make them public using the Google Drive web interface. Support for other cloud storage providers (Dropbox and OneDrive) is planned.

Programs can also be saved automatically to the web storage associated with the browser. This is particularly important when network connections are not available (ToonTalk Reborn runs fine if its files have been copied to the local disk).

ToonTalk also supports publishing programs to the web. A ToonTalk web page is created containing the widgets in the current workspace. The user can add and edit rich text around those widgets. New widgets can be dropped on these pages. This functionality relies upon Google Drive's support for serving saved documents as web pages which, unfortunately, Google announced it is stopping in 2016. Dropbox continues to support this functionality and will replace Google Drive for publishing web pages with embedded ToonTalk elements. Alternatively a proxy server could fetch pages from Google Drive and serve them as web pages.

Translation

Google Translate can be associated with a dynamic web page. As the page changes under program control text can be translated to the desired language. It is possible to customise the translations of some phrases and to crowdsource corrections.

Future Developments

ToonTalk Reborn has yet to be integrated with the 2D and 3D graphics of canvases and WebGL. Logo turtles have been implemented as a user program in ToonTalk Reborn but there is no fully general way to implement a trail left behind when the turtle's pen is down. After canvases are integrated with ToonTalk this should be relatively easy.

Desktop ToonTalk supports *long-distance birds* that can deliver widgets to nests on other computers. This is used for collaboration and implementing multi-player games. Long-distance birds could be implemented using WebRTC.

Google's real-time API for Google Drive (developers.google.com/google-apps/realtime/overview) could be used to enable simultaneous collaboration when programming ToonTalk. It could enable simultaneous editing of shared programs in a manner similar to collaborative editing of Google Docs.

ToonTalk has yet to be integrated with devices and sensors. Such integration would enable ToonTalk programs to access cameras, microphones, game pads, and smart phone sensors.

Lessons for other Languages

This paper has argued that web browsers offer programming language implementers a wide range of functionality that has mostly been ignored. ToonTalk Reborn was presented as a case study of how some of this functionality could be exploited. Many of the lessons here could be applied to make richer and more powerful versions of Snap!, Logo, NetLogo, and other constructionist programming languages.

References

Kahn, K. (1996) *ToonTalk - An Animated Programming Environment for Children*. Journal of Visual Languages and Computing, June. pp 197–217.

Kahn. K. *(*2001*) Generalizing by Removing Detail: How Any Program Can Be Created by Working with Examples*, Communications of the ACM, March 2000, pp 21-43*.* Long version in *Your Wish Is My Command: Programming by Example*, edited by H*.* Lieberman, Morgan Kaufmann, February*.*

Kahn, K*. (*2014*) TOONTALK REBORN - Re-implementing and re-conceptualising ToonTalk for the Web.* Constructionism 2014*.* Vienna*.* August*.*

Kynigos, C. (2007) *Half-Baked Logo Microworlds as Boundary Objects in Integrated Design*. Informatics in education. Volume 6 Issue 2, January. pp 335-358.

Miller, M., Van Cutsem, T., and Tulloh, B. (2013) *Distributed Electronic Rights in JavaScript*. 22nd European Symposium on Programming. Rome. Springer. March. pp 1-20

Saraswat, V. (1993) *Concurrent Constraint Programming*, MIT Press, March.

The source code and the web application are available at github.com/ToonTalk/ToonTalk/wiki.

Learning Emotional Aspects of Digital Competence By Creating Artefacts

Michael Weigend, *mw@creative-informatics.de*
Arbeitsgruppe Didaktik der Informatik, University of Münster, Fliednerstr. 21, Münster, Germany

Lisa-Marie Jung, *Lisa-Marie.Jung@rub.de*
Ruhr-Universität Bochum, Germany

Sarah Lenzen, Maike Gebhardt, Caroline Flaßhoff, Alisha-Sophie Unger, Julian Wagner, Weronika Jarosz, Stella Krämer, Stefanie Schweitzer
Holzkamp Gesamtschule, Willy-Brandt-Str. 2, 58453 Witten, Germany

Abstract

This contribution presents a 45-min workshop on media education designed for 11-years old children (grade 6). After a brief introduction offering some background information, the children are challenged to be creative. They work in groups of three or four plus a student from a pedagogy class (grade 12) as "gamemaster". Using "task cards", the children create stories or images individually and then present and discuss them within the group. After that, each group creates a drawing or Lego sculpture collaboratively (fig. 1). Workshops of this kind covering "cyber-bullying" have been conducted with 120 pupils.

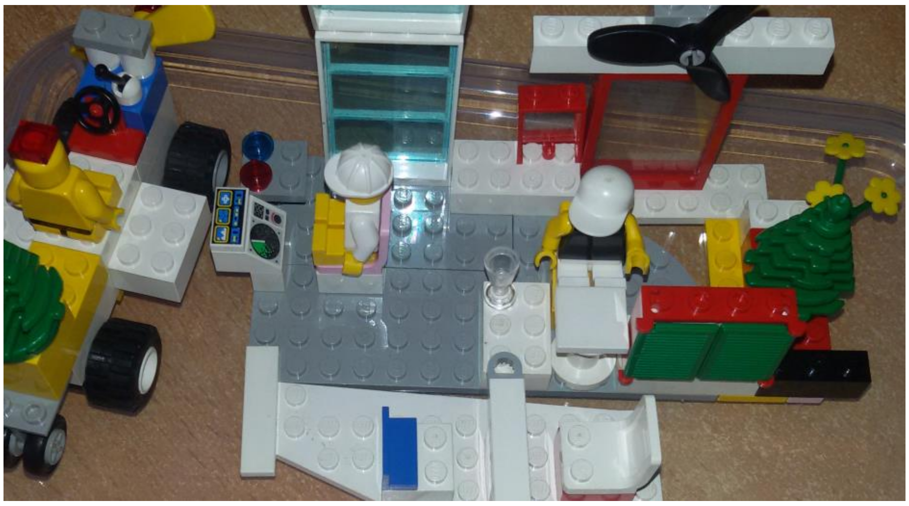


Figure 1: A Lego sculpture representing a "media consultant office"

Keywords

Cyber-bullying, digital competence, Lego, collaborative learning, media education

Introduction

The “Media Passport” (in German: "Medienpass NRW") was initiated in 2010 by the government of the German federal state North Rhine Westphalia (https://www.medienpass.nrw.de/de). The intention is to increase children’s media literacy. It defines a framework of 20 competences, clustered in five groups:

1) Operating and applying. This includes technical background knowledge about the world wide web, usage of basic functions of an operating system and digital media editors (text, image, presentation etc.)

2) Information retrieval and research. This includes finding and evaluating information from the WWW.

3) Communicating and collaborating. Use digital media for communication and collaboration in a responsible way.

4) Producing and presenting. Create text documents and presentations using digital technology.

5) Analyzing and reflecting. Understand functions, effects and meaning of media products.

The framework is specified in more detail for four age groups: Kindergarten, elementary schools grade 5 and 6 and grade 7 to 10. This contribution focuses on grade 5-6. The “Medienpass” is not compulsory. Schools may take part or may not. Meanwhile two thousand schools have registered. Most of them are elementary schools, but more and more secondary schools want to use this opportunity as well. A school taking part must integrate lessons on media education covering the competences mentioned in the “Medienpass” into its curriculum. Students in grade 5 and 6 get a booklet, in which achieved competences are documented. For grades 7-10 there exists a web-based competence tracking system run by the department of education. For example designing presentations and image processing (covering competencies from groups 1, 2 and 4) might be integrated in grade 6 biology classes.

Some competences listed in the Medienpass framework have emotional implications, for example:

- Students are careful and consent, when expressing their personal opinion and revealing personal data in the web (3.2).
- They can describe patterns of cyber-bullying and possible consequences for the victims. They know which persons they can ask for advice and how to react when they have been bullied (3.3).
- Students know age classifications of movies and video games and discuss effects of excessive media consumption as well as approaches to cope with it (5.2).

The workshop discussed in this paper focuses on the second competence description (3.3). In the examples given in the bullet list, the Medienpass framework explicates competences using terms like “they can describe” or “they know”. However, in the underlying scenarios the cognitive aspect is almost irrelevant compared to the emotional implications. It is trivial just to *know* that when you get bullied by classmates, you could talk to the teacher or to your parents. (In fact only a small minority of bullied kids in Germany do (see below). Obviously, the cause for avoiding talking to educators is not a lack of knowledge. The reason could be missing trust or the fear that humiliations and offenses get public. Often, a bullied person’s behaviour is determined more by emotions than by cognitions.

The term “emotional intelligence” was coined by John D. Mayer and Peter Salovey in 1990. It describes the ability to perceive (correctly) one’s own and others’ feelings, to understand and to control them. Daniel Goleman has popularised this concept with his book EQ (1995). He points out that that not only cognitions, but also emotions determine experience, behaviour and activity. Goleman distinguishes four competence areas of emotional intelligence:

- Self-perception (self-awareness): Accurate perception and understanding of one’s own feelings, confidence.
- Self-management: Control of one’s own feelings and impulses.

- Empathy (social awareness): Perceiving and understanding other people's feelings and relations.
- Relationship management: Understanding and influencing interpersonal relations.

There are several inventories to assess emotional intelligence. Popular tests are the MSCEIT inventory by Mayer and Salovey and the ECI (Emotionally Competence Inventory) by Goleman.

In the context of cyber bullying high school students need different emotional competences depending on the role they play in the specific case.

- A person who gets bullied (victim) needs self-management competences to control his or her own negative feelings and for example to not start a vendetta.
- A victim's friend needs empathy to realize that his or her friend is in trouble.
- A member of a fairness team or media support team which is involved in a bullying case needs relationship competences.

83% of German children in the age of 12 to 13 own a smartphone with internet connectivity (KIM 2014, p. 11). On their way to school and during school breaks smartphone and internet are the most important media for German children in the age from 6 to 13 (KIM 2014, p. 18). The average age for registering in a social network (mostly Facebook) is 10.4 years. 70% of German internet users in the age of 12 to 13 are members of social networks (KIM 2014, p. 37). These numbers suggest that communication in online communities is new and relevant competence for sixth-graders in Germany. In a survey of 6.993 students from secondary schools in winter 2012/2013, 17% reported they had been victims of cyber bullying (mostly verbal insults) and 19% said that they had bullied someone (Schneider et al. 2013, p. 94, 98). The motives seem to be rather trivial. 53% of the offenders said they did this “out of boredom”. Other motives were “just for fun” (51%) or “because other persons do the same” (36%). The victims' emotional reactions are mostly anger (44%) and fear (36%). Peers seem to be the most important source for emotional support for bullied children. 33% of the boys and 51% of the girls ask friends for advice and try to find a solution together with them. These numbers underline the importance of emotional competences when it comes to preventing and coping with cyber mobbing.

In our workshop we try to use collaborative Lego constructing and collaborative drawing for developing media competences. Lego plays a considerable part in technology, science, math and language education. When assembling machines, children understand mechanical parts like cogs or levers better because they can touch them with their own hands and watch the machinery working. Lego constructions are incentives to count and calculate lengths. Lego is also a means to foster creativity and language development. Children build sceneries, take photos with a tablet and use digital tools like Lego StoryVisualizer (https://education.lego.com/en/lesi/elementary/storystarter/ ss-software) to create a comic-like story with speech bubbles. There are lesson plans with Lego that require several hours or even days. In contrast to this, in our workshops the students have only 10 min for the creation process. There is no planning before building. The students are not supposed to think of a story first and then build a 3D-illustration with Lego. Instead, they start building immediately and create the story – or many “micro stories” – collaboratively while moving Lego blocks (“think with your hands”). Since they are creating in a team they should communicate all the time and explicate their ideas. In the workshop we focus on this process, the artefact at the end is more or less irrelevant. This approach is not necessarily a contradiction to Constructionism, since – as Papert (2000) points out - “big ideas” are found *en route* while constructing. The artefact may be valuable and precious, but its quality is not essential for the intellectual learning during the creation process. Piaget's statement “to understand is to invent” (1973) can be interpreted in a constructivist way (to understand is to construct subjective knowledge) and in a constructionist way (to understand is to create a relevant artefact).

Collaborative drawing is used in (art) education to foster social learning and creativity. It is supposed to strengthen the children's tolerance towards other children. They learn to understand and to accept other persons' actions and get inspired by the others to new ideas at the same time. Briony Barr, an Australian conceptual artist who conducts art workshops for children and families, points out that collaborative creative activities require rules that both limit how to start the process and what to do within. “Limitations promote creative problem solving.” (Barr 2015, p. 5). A simple set of rules is this: The coach (teacher) draws an arbitrary line on the blackboard or paper sheet. One by one, each child draws another line so that in the end there is a complete collaborative drawing. In our workshops we did not specify strict rules. We preferred a rather open situation hoping that students would talk about cyber-bullying while making design decisions.

Design of the Workshop

A group of eight students from a pedagogy class in grade 12 were responsible for a 45-min-workshop in a grade 6 class with approx. 30 pupils. The workshops were designed according these principles:

- No special pre-knowledge is assumed.
- Students activities do not require digital technology (laptops, tablets etc.).
- “High speed” individual and collaborative activities (max. 10 minutes each)
- The activities should challenge imagination and should be fun, even if the matter is serious.

Each workshop starts with a presentation (approx. 5 min) prepared and performed by the pedagogy-students. They could choose any form (theatre play, presentation with slides etc.) but were encouraged to make their presentations interesting and entertaining. They could also involve their audience for example by asking questions. Each presentation is different, but it has to cover these points:

- What is cyber-bullying? What different patterns exist?
- What are consequences for bullied persons?
- What can children do, when they are bullied?

In the second part the children are challenged to be creative. They worked in groups of four children sitting at a table plus a pedagogy-student from grade 12 as “gamemaster”. His or her first duty was to explain two rules or mottos, which are printed on cards placed on the table:

- Everything is correct.
- If you do not know what to do, just start doing anything.

This is to encourage the children to start drawing or writing immediately. There are two rounds of group work. In the first round each child gets a card with a task, which must not be shown to the others. The children have to create stories or images individually and then present and discuss them with the group. The role of the game-master is to chair this discussion. In the second round each group creates an image or Lego artefact collaboratively. The gamemasters do not draw or build anything. They just observe and participate at the communication process in a careful and natural way, for example by asking about the meaning of certain details. During the last five minutes the gamemasters evaluate the session together with the children using a questionnaire.

We have designed two different versions of the second part (working in groups), which are described in the following sections.

Version 1: Individual Drawing and Collaborative Lego Building

In the first round each player gets a card with a drawing task and a question to the audience, for example: “Tom sends frightening messages to a classmate like ‘This night something terrible will happen’. Draw smileys that show Tom’s feelings, when he is writing a message like this. Question to the audience: The smileys express Tom’s feelings. What is going on with Tom?”

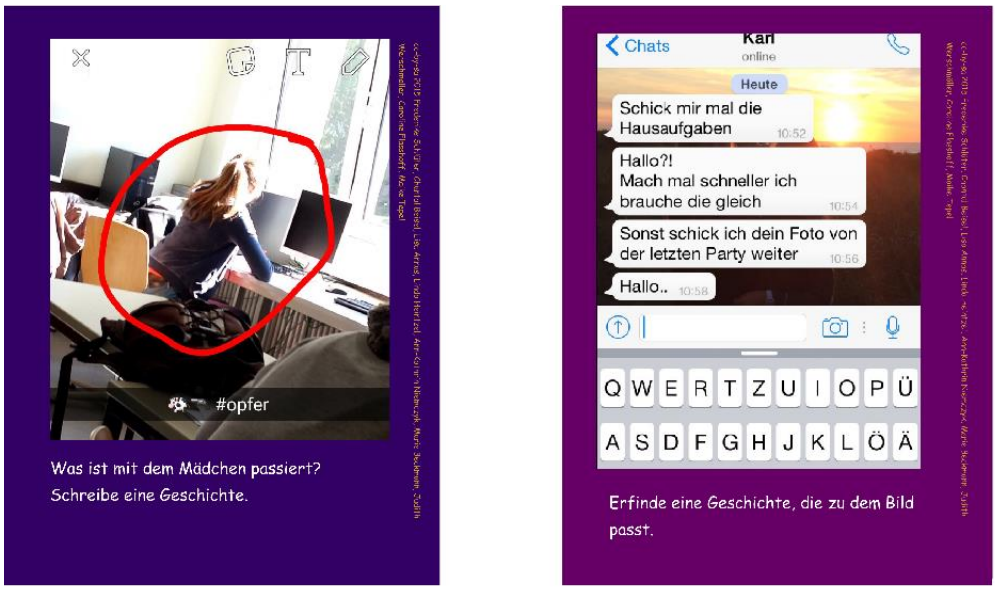
After five minutes the players present their drawings one after the other and ask the given question to the audience. In the second round the gamemaster pours put the Lego box and tells the task: “At many schools there are ‘media consultants’. These are students, who can give you advice using digital technology and help you when somebody tries to bully you on a social media. Build a media consultant’s office with Lego.” At the end of this activity the gamemaster takes photos of the artefact and might ask questions about meaning of certain parts.

Version 2: Individual Storytelling and Collaborative Drawing

In the first round each player gets a card with a screenshot from a mobile phone. There are different cards of this type related to different patterns of cyber-bullying.

Figure 2 shows two examples. On the left hand card you see a girl which is tagged with the German word for “victim”. The text below the photo tells: “What happened to the girl? Write a story!” The right hand card depicts a series of black-mailing messages: “Send me your home work ... otherwise I will publish your photo from our recent party...”

After approximately five minutes the gamemaster collects the cards, shuffles them and lays them face up in the middle of the table. Then everybody reads his or her story aloud and the others guess which card belongs to the story. The idea is to create a situation that provokes reflection and spontaneous discussion about cyber-bulling, verbalizing emotions (anger, fear etc.).



Was ist mit dem Mädchen passiert? Schreibe eine Geschichte.

Schick mir mal die Hausaufgaben 10:52

Hallo?! Mach mal schneller ich brauche die gleich 10:54

Sonst schick ich dein Foto von der letzten Party weiter 10:56

Hallo.. 10:58

Erfinde eine Geschichte, die zu dem Bild passt.

Figure 2. Two “task cards“ initiating story telling.

In the second round the gamemaster puts a big sheet of paper in the middle and passes a card to one of the players, who reads the story about a girl who gets frightening messages all the time. She is scared and her friends want to help her. The task for the team is to draw a picture illustrating what the friends can do to improve the girl’s situation.

Evaluation

The game masters observed the younger students while doing the activity. They filled a questionnaire during the workshop and immediately after the workshop. This way we got data about 120 pupils.

Question	Average score	
	Version 1 (N=53)	Version 2 (N=67)
1) How interesting was the presentation?	1.79	1.54
2) How interesting was the first exercise (individual design plus presentation)?	Drawing: 1.77	Story telling: 1.66

3) How interesting was the second exercise (collaborative design)?	Lego: 1.0	Drawing:1.58
4) Which exercise was more fun?	Drawing more fun: 3 Lego more fun: 45 Both the same: 5	Story more fun: 30 Drawing more fun: 29 Both the same: 8

Table 1: Rating the parts of the workshop (1=very interesting, 6 = completely uninteresting)

In the workshops version 1 (drawing and collaborative Lego) we got data about 53 participants (28 girls, 25 boys, average age 11.1 years). In workshop version 2 (story telling and collaborative drawing) we have data about 67 participants (34 girls, 33 boys, average age 11.3 years). Table 1 displays the results from questions the young students were asked during the evaluation phase. The scores suggest that the pupils liked the workshops. One may not compare the answers to question 4, since there were different types of individual work in the two workshops The answers to question 3 do not differ significantly (T-test, alpha = 0.05). However, the Lego activity got the highest score. The gamemasters observed the pupils' behaviour and communication. Table 2 displays the results.

Item	Version 1 (N = 53) Drawing and Lego	Version 2 (N = 67) Story and drawing
1) Average time for individual work	Drawing: 6.6 minutes	Story: 6.5 minutes
2) Average Level of focussing. 3 = students work quietly and are very focused, 0 = students mostly do not work voluntarily, gamemaster (GM) must intervene.	Drawing: 2.2	Story: 2.2
3) Average time for presenting and discussing results from individual work	Drawing: 5.2 minutes	Story: 5.5 minutes
4) Average amount of communication in the context of presentation. 3=many spontaneous comments, GM needs to interrupt discussions 0= practically no comments.	Drawing: 2.2	Story: 1.7
5) Average amount of communication during collaborative work. 3=constant spontaneous communication 0=everybody is quiet.	Lego: 2.3	Drawing: 2.4
6) Average estimated percentage of communication related to "cyber-bullying"	Lego: 50%	Drawing: 71%

Table 2: Observing individual and collaborative work.

Both types of collaborative tasks inspired the pupils to talk about cyber-bullying. The gamemasters estimated that 50% of the conversation during the Lego construction and 71% of the conversations during collaborative drawing were domain-related. In the Lego version, the children used the Lego blocks they found (seemingly as many as possible) and built fantasy rooms with unusual features. Fig. 1 shows an example. They built offices but they did not build stories. However, unusual elements of a scenery provoke questions, for example (fig. 2): Who are the two guys? Why are they wearing helmets? Why is there a car in the office? And they initiate discussions.

We made observations illustrating some advantages of Lego building against drawing:

- While building with given Lego blocks, (accidently) meaningful structures emerge that cause relevant conversations. For instance, a pupil put a human head on a fir. This strangely looking character inspired the children to discuss the idea that “this guy is bullied because he looks like a Christmas tree”.
- Lego sculptures can be developed, refined and changed easily and quickly. Each (intermediate) state looks nice. In contrast, a drawing can only be *extended* easily. But it is difficult to remove or exchange parts without destroying the beauty. A drawing always displays its history of creation. Even when you erase a part instead of crossing it out (which would be quicker) you still can see traces of it.
- Pupils sometimes apologize for not being able to draw persons properly. Additionally – due to of lack of time – they cannot edit an image until they are satisfied with its look. Most students sketch persons quickly as (black and white) stick figures. In contrast, Lego offers 3D and colour.
- Every Lego sculpture – even, when it took just minutes to create it – can be documented in an attractive way. A gamemaster can take photos from interesting perspectives, that could be put at the wall in the classroom and serve as decoration and souvenir.

Let us finally mention some qualitative observations on the collaborative drawing in the workshops of version 2 (“Draw a picture that shows how to help a girl that has constantly been annoyed by frightening messages”):

- Most drawings included speech bubbles.
- The collaborative drawing sometimes turned to some kind of role play. The pupils pretended to be the friends of the fictive victim and discussed how to help her.
- The most frequently mentioned support strategies were talking and making the victim to tell her feelings.
- Most drawings depicted a “happy end”.

Conclusions, Open Questions and Further Work

We did not try to find evidence that the 45-min workshop lead to increased media competence. According to Gijlers et al. (2013) domain-related communication during collaborative drawing in elementary science classes (using a digital canvas) leads to significant knowledge gain. Since our young participants were so active, focused and engaged one can assume that this intervention was not without effect. Our observations suggest that collaborative drawing under time pressure requires special tasks and incentives. One approach could be to reduce the complexity by avoiding naturalistic images and suggesting to use symbols or easy-to-draw fantasy figures representing humans. Digital drawing tools on tablets can offer predesigned elements and make it easier to create aesthetically acceptable pictures.

The main challenge for the design of Lego activities is to find plausible building tasks that can be done in 10 minutes and inspire conversation. The artefact is not supposed to illustrate a consistent story. But it can be related to many micro stories. For example, the artefact might be a special location, where certain stories can take place or some kind of machine, or vehicle that can be associated with domain-specific stories. We have planned further workshops covering topics from the “Medienpass”, including “age classification of movies and video games” and “effects of excessive media consumption”. Lego tasks related to these domains could look like these:

- Tom is addicted to his smartphone. Whenever it is possible, he checks Facebook and WhatsApp. Create a school yard that lets Tom forget his mobile phone and makes him play with his class mates instead.
- Tina (11 years old) had watched a horror movie together with her elder sister. Since then she has nightmares, wakes up in the night often and is afraid of dark rooms. Create a bedroom for Tina that makes her sleep well.

We will try to use simple image editing apps on smartphones (e.g. Phototastic Collage, PicSay, Photo Talks, Lego StoryVisualizer), to document the Lego artefacts and associated fantasy conversations. The gamemasters or participants take pictures of the built scenery and add speech bubbles together with the sixthgraders. But this requires some experience and must be organized in a way that it does not take too much time.

All group activities discussed in this paper are chaired by "gamemasters" who know the rules and the material and keep the activity running smoothly. This kind of support seems to be very efficient and is common in many areas including educational games for adult professionals like the XP-Game, a simulation of Extreme Programming (van Cauwenberghe & Peeters 2010).
In a secondary school the coaching can be done by students from higher grades. Working with younger pupils can be interesting and rewarding for students attending pedagogy classes, if the practical experience corresponds to theoretical content required in the curriculum. In this case this would include topics like cognitive development (Piaget), moral development (Kohlberg), emotional competence (Goleman) and symbolic interactionism (Mead).

References

Barr, B. (2015) .Creative. Artplay - Artists essays No. 1, URL: https://www.melbourne.vic.gov.au/artplay/research/Documents/Essay1_Briony_Barr_creative.pdf.

Gijlers, H., Weinberger, A., van Dijk, A.M. Bollen, L., & van Joolingen, W.R. (2013). *Collaborative drawing on a shared digital canvas in elementary science education: The effects of script and task awareness support.* International Journal of Computer-Supported Collaborative Learning, 8, 427-453.

Goleman, D. (1995): *Emotional Intelligence: Why it Can Matter More Than IQ.* Bantam Books.

Hay Group (2005) *Emotional Competence Inventory (ECI)Technical Manual,* McClelland Center for Research and Innovation, updated by Steven B. Wolff, DBA, URL: http://www.eiconsortium.org/pdf/ECI_2_0_Technical_Manual_v2.pdf

Medienpädagogischer Forschungsverbund Südwest (2015): KIM-Studie 2014. Stuttgart. URL:

Papert, S. (2000). *What's the big idea? Toward a pedagogy of idea power.* IBM Systems Journal, 39(3.4), 720-729.

Piaget, J. (1973): *To Understand is to Invent: The Future of Education*; New York: Grossman Publishers.

Schneider, Ch., Katzer, C. and Leest, U. (2013): Cyberlife – zwischen Faszination und Gefahr [Cyberlife – Between Fascination and Threat], Karlsruhe, URL: http://www.buendnis-gegen-cybermobbing.de/Studie/cybermobbingstudie.pdf

Van Cauwenberghe, P. & Peeters, V. (2010): The XP Game. http://www.xp.be/xpgame/ *http://www.e-russell.com/images/ESCI_Article.pdf*

Learning Intentions and Educational Robots

Dave Catlin, dave@valiant-technology.com
Valiant Technology Ltd.

Abstract

Some teachers run excellent lessons with educational robots. Others fail. Good teaching practise, is the key to success and prevails despite diverse and difficult challenges. What is good practice? How can we make sure teachers apply it to educational robots? Constructivism underpins the use of robots, but putting theory in to practise has met with difficulties. The increased focus on curriculum and high-stakes testing makes matters worse. Most teachers I meet feel bullied into "teaching to test" and feel forced into abandoning constructivism for more direct teaching methods. Can teachers deliver lessons that meet their curriculum duties and keep the constructivism spirit alive? These practical questions concern the educational robotic community[14]. This paper is one of a series that looks at these issues.

In previous work, I proposed Assessment for Learning (AfL) answered these questions. AfL summarises good practice and provides a way to improve the success of educational robots. In later papers I looked in more detail how AfL (Success Criteria and Peer Assessment) might work with robots. In this paper, I continue this effort by exploring issues to do with educational robots and another AfL strand - Learning Intentions. I review teacher and expert opinion on this topic and develop a definition that works with robots. Finally, I use these ideas with selected Roamer® activities[15] to highlight some of the application issues.

Keywords

Assessment for Learning, AfL, Learning Intentions, Learning Challenges, Learning Objectives Educational Robots, Valiant Technology, Roamer, Turtles, Constructionism, Constructivism.

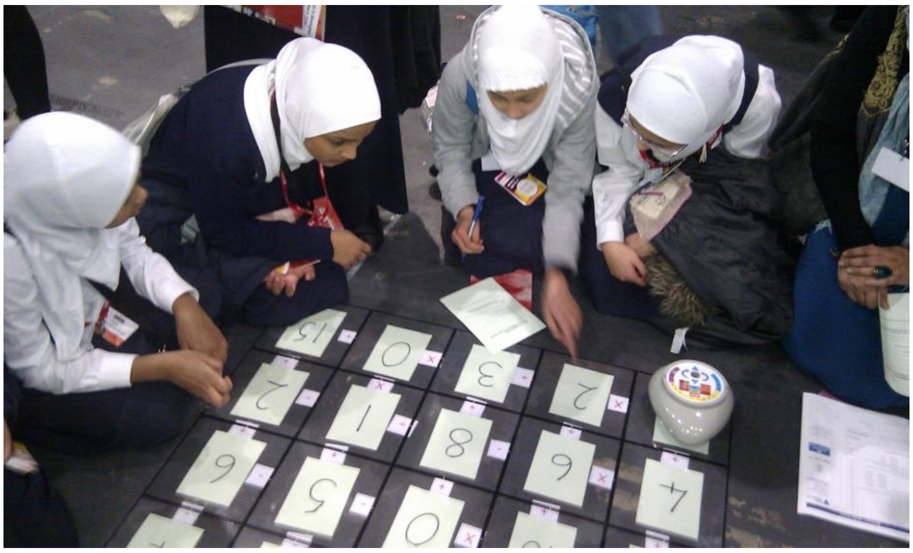


Figure 1 Muslim girls solving a problem with the Roamer robot. The Learning Objective is, "Students fortify their understanding of arithmetic. The students program the robot to follow a route doing calculations at each step. The Learning Intention is, "Finding out how to get the biggest score". Activity by Stephen Wooley.

[14] I refer to the agenda of RiE (Robots in Education) and TRTWR (Teaching Robotics and Teaching with Robotics). These are annual and bi-annual educational robotic conferences based in Europe.

[15] The activity are published BY-NC-ND Creative Commons Licence. They are available at www.activities.roamer-educational-robot.com. Attribution is Valiant Technology Ltd and the activity authors.

Introduction

The Problem

I started designing Turtle type educational robots in 1983. Over the years, I have noticed some teachers successfully use the technology and others struggle. The problem is not the teacher's technical skills. I have seen teachers who know how to use the robot but manage to deliver sterile lessons. I have also seen the opposite: wonderfully imaginative lessons with children full of enthusiasm, eagerly exploring ideas. The teacher had limited technical knowledge, but excelled in exploiting what they knew. What can we do to make this the norm?

The ERA Principles

As a robot manufacturer, I face a problem: not everyone using the robot is a trained teacher. How can I help all users get success? Mike Blamires and I wrote the ten Educational Robot Application (ERA) Principles (Catlin, D. and Blamires, M., 2010). These explain the value of educational robots. I use them to collate the benefits of robots used in diverse educational scenarios. It helps me compare how the robot aids a 4-year old understand number to how it helps an 18-year-old grasp vector analysis. ERA has another purpose. It provides a "design specification" that guides our creation of educational robots and the way we use them. In this paper I focus on the Curriculum and Assessment Principle which states: Educational Robots can facilitate teaching, learning and assessment in traditional curriculum areas by supporting good teaching practice. A key phrase here is "good teaching practice". What is it? If we find that out, how can we apply it to the way we use educational robots? AfL offers a solution.

I also refer to the Engagement Principle: *Through engagement Educational Robots can foster affirmative emotional states and social relationships that promote the creation of positive learning attitudes and environments, which improves the quality and depth of a student's learning experience.* This is a broader idea than "makes learning fun", I will propose that it is an integral part of setting up Learning Intentions.

Introducing AfL – Assessment for Learning

In 1998 Black and Wiliam first published a seminal paper "Inside the Black Box – Raising Standards through Classroom Assessment" (Black & Wiliam, 2006). In this analogy, the classroom is the black box. Conventional testing evaluates what comes out of the classroom. This is Assessment **of** Learning (AoL). AoL benefits others. For example, it helps politicians review whether the school is value for money. It allows parents to compare schools. Assessment **for** Learning (AfL) happens in the lesson. Teachers use AfL methods (Table 1) to check student responses. Do they need extra help with something? Have they discovered something interesting they should explore? AfL helps the teacher manage and improve the learning taking place during the lesson. I believe that AfL captures and codifies good teaching practice. Some of its techniques may appear novel, but, in essence, it summarises what expert teachers do intuitively. Indeed, what they have done for decades.

Table 1 Elements of AfL adapted from Smith (Smith, 2007)

Element	Explanation
Learning Intentions	The student's view on what they are learning.
Success Criteria	How will the student recognise successful learning?
Quality interactions and feedback	Positive, enriching ways for teachers and students to interact during a lesson. Methods a teacher can use instead of grading a pupils work. The aim is not to tell them how well they did, but how they can improve what they did.
Peer Assessment and Self-Assessment	Using what students think of their work and the work of their classmates to improve learning.

Applying AfL to Educational Robotics

The US Department of education classifies teachers according to their experience (US Dept of Education). A Master Teacher is a good educator who models effective teaching practises and shares their skills to other teachers. My studies of master teachers found:

- They used constructivism teaching methods.
- They worked successfully with educational robots.
- They work at the unconscious competence level (Burch, 1970)[16].
- They unconsciously use variations on AfL methods.

I proposed that AfL offered a way of organising activities with educational robots that would help address the basic issues (Catlin D., 2012). I developed the peer and self-assessment ideas (Catlin D., 2014) and proposed using Success Criteria for evaluation (Catlin, Csizmadia, OMeara, & Younie, 2015). In this paper, I look at the details of Learning Intentions and educational robots. I start by analysing diverse teacher opinions. I then look at the views of an AfL expert. I make a distinction between Learning Objectives and Learning Intentions. Finally, I look at applying these ideas to robot activities and some of the issues it raises.

Analysis of Learning Intentions and Objectives

General Views

In an interesting blog, Canadian teacher Joe Bower (Bower, 2011) states: "Stop writing Objectives on the Board". The blog has attracted over 70 comments from various countries. The debate was still active in September 2015 when over 33,000 people read the blog that month. It is a popular topic. The following summarises the debate:

- An analysis of commentators showed 31% favoured making the students aware of the objectives, 58% thought it bad practice and 11% preferred a mix of the two approaches.
- The blog did not present a clear definition of terms. People used various terms interchangeably when clearly they were talking about different things. Few people distinguished between Learning Objectives and Intentions. This included people familiar with AfL ideas.
- Several teachers normally display objectives to satisfy administrators:
    - They do not believe it helped students learn.
    - Lesson evaluators mark them down if they do not display the objectives.
    - Sometimes they must display objectives matching curriculum statements.
- Some comments discussed the students perspective:
    - Teachers recalled knowing the objectives helped them when they were students.
    - Some constructionist teachers proposed students should set the objectives.
    - Some places now require students to copy objectives into their books.

[16] This model states 4 levels of competency. At the bottom, it starts with unconscious incompetence. The next level is conscious incompetence. This gives way to conscious competence and then the final stage of unconscious competence. In this stage, people demonstrate their skills spontaneously.

- People suggested:
  - Objectives should focus on process, not outcomes.
  - Establishing objectives using key questions and Bloom's taxonomy.
- Reasons for disliking objectives were:
  - It implies learning is driven by the teacher and not the student.
  - It "gives away the ending" and stops student discovery.
  - It deters pursuit of student inspired lines of inquiry.
  - It is extra work for the teacher without a clear advantage.
- Reasons for liking objectives included:
  - It is usually a good idea for people to know where they're going.
  - It helps the students monitor their progress.
  - It helps students to become responsible for their learning.

Some constructivism practitioners strongly disagree to setting objectives. "How dare you tell me what I am going to learn"; "How can you call 'learning' something you predetermined?"

Dylan Wiliam on Learning Intentions

AfL expert Professor Wiliam (Wiliam, 2011; 2012) made several observations:

- Assessment becomes 'formative' when teachers use the evidence to adapt their teaching to meet student learning needs.
- Teachers lead formative assessment, but it is something students do.
- A narrow Learning Intention is not always enough to direct student learning.
- Creating a good Learning Intention is more craft than science.
- You need to make sure writing Learning Intentions does not become a dull routine.
- Sometimes
  - Giving a Learning Intention takes out the mystery of a lesson.
  - The Learning Intention leads to one answer (maths).
  - The Learning Intention leads to various answers (Literary Criticism).
  - You can have a whole horizon of goals and it's ok if kids go in different directions.
  - Setting up a good Learning Intention leads to inspiring lessons.
  - You want to focus on process.
  - You cannot always write a Learning Intention for a lesson.
  - A better way of starting a lesson is with a question that grabs their attention.

The Mayflower Lesson

Bower's blog starts with a clip of educator Alfie Kohn (Kohn, 2011) discussing a lesson where a teacher introduced the idea of non-standard measurements to students. The lesson epitomised constructivism in practice. When pupils arrived they found the outline of the Mayflower on the floor. A student, Zeb, appeared dressed as a royal herald and read from a scroll. The decree told the class they could board the Mayflower if they could tell the King how long the ship was. What followed was a lesson which showed the children's creativity and engagement as they explored the ideas of measurement.

Misunderstanding Constructivism

The Mayflower lesson challenges those blog commentators who think students must set the objectives, or that you could not set an objective because you did not know what was going to happen. Misconceptions like these bolster opponents of constructivism (Cox & Dyson, 1971; Price, 2009). The 'Mayflower' teacher had a **clear plan and objective that met curriculum needs**. Few plans ever work out without management. Student inspirations may take them away from the objective. Teachers need to use their professional judgement to manage the lesson: does the detour benefit the student learning or should they bring them back on task?

AfL helps organise activities in a way that lets the teacher connect to the curriculum, but still allow the freedom associated with constructivism. The role of Learning Intentions is to start the lesson in a way that helps achieve these aims. Often schools use educational robots in special events or after-school clubs. The ERA Curriculum and Assessment Principle says their proper place is supporting normal schoolwork. Learning Intentions help set up lessons using the robot.

Learning Intentions versus Lesson Objectives

I find it useful to distinguish between Learning Intentions and Lesson Objectives. A Learning Intention focuses on what the student thinks they are learning. A Lesson Objective connects the work to the curriculum. The Lesson Objectives of the Mayflower example might say, "Students will learn about standard and non-standard units of measurement". To most students this is gobbledegook. Compare it with what children might say, "We are finding out how to measure the ship". Educators claim you should negotiate Learning Intentions with students, but it seems to me this rarely happens. Often, the teacher creates the Learning Intention and the "negotiation" turns into an explanation. Nothing the students say contributes to what the teacher writes on the blackboard. Setting up Learning Intentions should aim to engage students and help them to take ownership of their learning. This has nothing to do with "telling students what to do". The Mayflower teacher had no idea what the students would do. She was ready to help and guide, but not direct. Although she did not make it explicit, she did set up a Learning Intention by engaging the students with their learning.

A lesson rarely achieves an objective like, "Students will learn about standard and non-standard units of measurement". As Vygotsky makes clear, knowledge gradually emerges from many different experiences. The Learning Intention "We are **finding out** how to measure the ship", contributes to the Learning Objective in an achievable way which you can measure. Note "finding out" highlights learning.

Engagement

Engagement is a characteristic of the Mayflower example. Getting students to buy-in to a task is a key part of setting up Learning Intentions. Trevor Thomson is a teacher who helped me understand how to put constructivism theory into practice (Catlin, Thompson, & Year 6 Students, Fleet School, Class of 1998, 2014). The Design Technology (D&T) Exhibition invited his school to show their work in their 1998 exhibit. At the time making fairground rides was a favourite D&T project. The pupils chose to make various automata for a circus. When I said it was lucky they had chosen this richer theme, Trevor replied, "I knew they would". He used a strategy straight out of Dale Carnegie's How to Win Friends and Influence People (Carnegie, 2006): If you want people to do something, get them to think it is their idea. Student buy-in is critical.

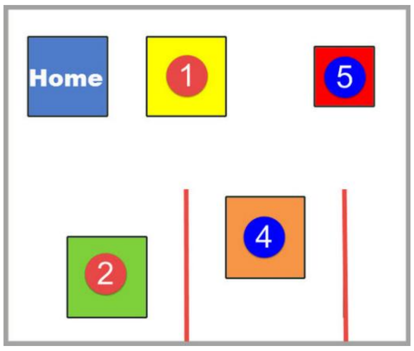
Examples

These examples do not explain anything about robots or how we use them in the classroom. That is beyond the scope and editorial limits of this paper. You can find this information elsewhere[17]. The examples look at setting up Learning Intentions for robot activities.

Number Grab

The playing area of this game (Fig 1) has several targets labelled with a score. The students write a program that will move the robot from target to target. They score points for each target the robot lands on within in a 2-minute session. Only points the robot gets back home count. You vary the activity by changing numbers, for example, to mixed fractions. You can add more targets and change numbers for operators. Students work in pairs. The pairs come together to test their programs. While one team is running a test, the others act as referees (peer

Figure 2 Number Grab Target Area. Authors Dave Catlin and Alan



[17] Check out the Roamer website www.roamer-educational-robot.com.

review). They check whether the robot landed on the target and the score is correct.

Table 2 A Possible Way of Finding the Best Answer

▲ 1 ▼ 1	Roamer moves forward 1 and back 1.	Score 1	Distance moved 2	Score Distance 1:2
▲ 1 ▲ 3 ▼ 3 ▼ 1	Roamer moves forward 1 then forward 3. Then it moves back 3 and back 1.	Score 7	Distance moved 8	Score Distance 7:8

The task offers students the chance to improve their computational fluency in a problem-solving context. They get to practise estimation and coding. The activity has hidden depths (table 2). Students might also think about trying to work out the speed. So if the robot travels 1 unit in 1-second, the students can repeat the program 120/9 - fifteen times.

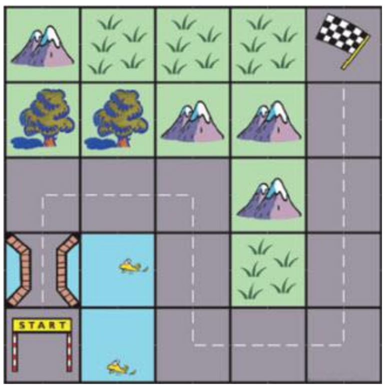
In tasks like this, you start by explaining the game followed by a question: "How are you going to do this?" Students will make statements like, "I'll go lots of times to the nearest number?" The aim is to get the students discussing the problem and potential solutions. This brings out an important adjunct to AfL – effective questioning. Students can surprise you with their thoughts, but in the words of Oscar Wilde – spontaneity is a meticulously prepared art. You can prepare questions (Wragg & Brown, 2001; Pope, 2013). "Do you think you can get a bigger score?" "Will you run out of time?" "What would happen if you went for the big numbers?" You should not force the students to discover the hidden mathematics mentioned above. However, asking questions in a discussion will prompt them to think more deeply about the problem.

At the end of a discussion, you can talk to the students about what they think they're learning. Instead, of them regurgitating something you wrote on the blackboard, they will express their thoughts. You can guide them, but recording their words helps them own their learning. A Learning Objective "developing computational fluency and mathematical thinking" becomes a Learning Intention, "Finding out how to get the biggest score" or "Thinking about how to win".

Robot Rally Race

The robot runs at different speeds over the various topographies. The task is to program the robot to get from the start to the finish in the fastest time. The first job is to do speed trials over the terrains. Pupils use that data in their programs. The teacher gathers the data from each team, which the class reviews using statistics.

Figure 3 Robot Rally Race by Dave Catlin and Alan Coode.



You will normally find it easy to engage students in games like this. The activity is split into different parts. Each part needs a different Learning Intention. The first phase of this work is gathering and using data. The students get to reinforce some calculation skills, run experiments and use mathematics to solve problems. In the second phase students start to analyse the collected data.

Sindbad's Treasure Hunt

Sinbad and his crew have to make a long journey in a hot desert[18]. Before they set off they need to make a tent. Students split into 3 teams of 2. One team use their Roamer and measure some "tent poles". Another team uses Roamer to measure "the tent covering" (paper). The final team use their Roamer and measure the "guy ropes". When they bring the materials together they find a problem – each Roamer has a different unit of movement so the parts will not come together to make a tent. This is an example where starting with a Learning Intention (Learning Objective:

[18] Authors: Dave Catlin, Kate Hudson and Alan Coode.

exploring the need for standard units of measure) will not work. You need to set up the Learning Intention when the students realise the problem. A set of effective questions aimed at guiding the student discussion on how to resolve the problem is valuable. The aim of the discussion is to help the students discover why we have standard units and that the choice of units is arbitrary.

Spacecraft Rescue

You can see details of this activity[19] in (Catlin D. 2012) and in the doctoral thesis (Holmquist, 2014). A spacecraft has crash-landed in a deep ravine. Your team needs to bid to recover it. You need to build a crane to lift the spacecraft out of the ravine, use the robot to move the crane to the crash site and bring the craft back to base. All the materials and manufacturing processes have a cost. The robot movement has a cost. Your challenge is to complete the task for a minimal cost. Holmquist did this activity in an elementary school. I did the project with a Grade 12 Gifted group. With the younger students, the project broke down into several subtasks. Each of these had a different Learning Intention, mostly targeting learning of subject matter. The older students also dealt with the content, but the Learning Intention-focused on higher levels of Bloom's taxonomy and the development of life skills. This suggests the same activity can require different Learning Intentions for different students.

Delivering Letters

A preschool activity[20] asks students to program Roamer to deliver letters to some house in a street. What is the quickest way of delivering the mail? The Learning Objective aims to "Develop an understanding of ordinal numbers". One teacher told me that when she asked a pupil, "How did you know that", the child replied, "Magic". Early Years teachers spend a lot of time helping students understand what and how they are learning. You may need to remind students of the Learning Intention several times.

Conclusions

This paper develops further the notion that AfL methods can support the effective use educational robots. The following summarises key points about Learning Intentions:

- I define Learning Intention as what the child thinks they are learning.
    - It should reflect their understanding and language.
- In contrast, a Learning Objective links student work to curriculum objectives.
- Learning Intentions set up constructivist learning environments.
- Setting up a Learning Intention involves engaging students in the tasks and helping them to take responsibility for their learning.
- Learning Intentions focus on student learning, not what-to-do instructions.
- Involving students in discussions and using effective questioning is a good way of setting up Learning Intentions.
- As a rule, teachers should set up Learning Intentions at the start of a lesson, but this is not always desirable.
- It is not always possible to set up a Learning Intention.
- Some activities require multiple Learning Intentions.
- You might use different Learning Intentions if you use an activity with different age groups or different cultural settings.
- You might need to remind students of the Learning Intention during in the lesson.
- You can change the Learning Intention in a lesson if it improves a student learning.

[19] Author: Dave Catlin

[20] Author: Chrissie Dale

Gathering data on the value of this approach is ongoing under the e-Robot Project (Catlin & Blamires, 2010a). This is an online project where teachers provide data on their use of the Roamer activities. I will publish these results in the future.

References

Black, P., & Wiliam, D. (2006). *Inside the Box: v. 1: Raising Standards Through Classroom.* NFER Nelson.

Bower, J. (2011, October 27th). *Stop writing the objectives on the board*. Retrieved December 1st, 2015, from For the Love of Learning: http://goo.gl/865VHm

Burch, N. (1970). *The conscious competence ladder*. Retrieved December 1st, 2015, from Mind Tools: https://www.mindtools.com/pages/article/newISS_96.htm

Carnegie, D. (2006). *How to Win Friends and Influence People.* London: Vermilion.

Catlin, D., Thompson, T., & Year 6 Students, Fleet School, Class of 1998. (2014). *Fleet School Circus Project*. (Valiant Technology Ltd) Retrieved April 11th, 2014, from Roamer Podcast Site: http://podcast.roamer-educational-robot.com/other-projects

Catlin, D. and Blamires, M. (2010). The Principles of Educational Robotics Applications (ERA): A framework for understanding and developing educational robots and their activities. *Constructionism 2010.* Paris: Proceedings of Constructionism 2010.

Catlin, D., & Blamires, M. (2010a). The e-Robot Project: A Longitudinal On-line Research Collaboration to Investigate the ERA Principles. *TRTWR 2010 Conference, part of SIMPAR 2010.* Darmstadt, Germany.

Catlin, D. (2012). Maximising the Effectiveness of Educational Robotics through the Use of Assessment for Learning Methodologies. *3rd International Conference on Teaching Robotics, Teaching with Robotics.* Riva La Garda, Italy.

Catlin, D. (2012). *Spacecraft Rescue*. Retrieved December 1st, 2015, from Roamer Tumblr: http://roamerrobot.tumblr.com/post/27181743875/spacecraft-rescue

Catlin, D. (2014). Using Peer Assessment with Educational Robots, Peer Review. *Peer Review, Peer Assessment and Self-Assessment in Education, ist International Workshop, collated with the 13th International Conference on Web-based Learning ICW.* Tallinn, Estonia: Springer Verlag.

Catlin, D., Csizmadia, A. P., OMeara, J. G., & Younie, S. (2015). Using Educational Robotics Research to Transform the Classroom. *RiE 2015: 6th International Conference on Robotics in Education.* Yverdon-les-Bains.

Cox, C. B., & Dyson, A. E. (1971). *The Black Papers.* (C. B. Cox, & A. E. Dyson, Eds.) London: Davis-Poynter.

Holmquist, S. (2014). *A multi-case study of student interactions with educational robots and impact on Science, Technology, Engineering, and Math (STEM) learning atitudes.* University of South Florida. Tampa Bay: University of South Florida.

Kohn, A. (2011, October 24th). *Alfie Kohn Math*. Retrieved December 1st, 2015, from YouTube: https://www.youtube.com/watch?v=9gkplk3uEW4&feature=youtu.be

Pope, G. (2013). *Questioning Technique Pocketbook.* Alresford, England: Teachers' Pocketbooks.

Price, B. D. (2009). *The Con in Constructivism*. Retrieved December 1st, 2015, from Improve Education: http://www.improve-education.org/id55.html

Smith, I. (2007). *Assessment and Learning: Pocketbook.* Alresford, Hampshire: Teachers' Pocketbooks.

US Dept of Education. (n.d.). *Section IX—Appendix: Sample Teacher Role Structure.* Retrieved January 26th, 2015, from The Respect Project Vision Statement: The RESPECT Project: Envisioning a Teaching Profession for the 21st Century: http://goo.gl/DIok2t

Wiliam, D. (2011). *Embedded Formative Assessment.* Bloomington: Solution Tree Press.

Wiliam, D. (2012, December 12th). *Unpacking Formative Assessment*. (N. W. (NWEA), Producer) Retrieved December 1st, 2015, from YouTube: https://goo.gl/btXsAz

Wragg, E. C., & Brown, G. A. (2001). *Questioning in the Primary School.* London and New York: Routledge.

Math-based Coding Education in Korean School

Han Hyuk Cho, *hancho@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Jin Hwan Jeong, *hyperdak@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Jong Jin Kim, *themavine@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Yong Hyun Seo, *dhsyh7@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Seung Joo Lee, *2ndclassroom@snu.ac.kr*
Dept of Mathematics Education, Seoul National University, Seoul 08826, Korea

Abstract

Recently, the Korean Ministry of Education announced three policies for education -software (SW) education, free semester system, and character education - to strengthen the future skills of learners. The implement into schools of the policies, however, has been hindered by the absence of curricula and the ambiguity of the assessment that meet the purpose of each policy. A way to resolve the problem is proposed in the study by reflecting Constructionism. An educational program, based on Constructionism, will foster the future skills of learners through 'learning by making'. In Fig. 1, where the program is summarized, exploratory activities through symbolic coding lead to meaningful experiences through physical coding, which in turn leads to another exploration on the next level. Within the process of coding, symbolic coding exploration is expected to promote learners' computational thinking (CT) competencies and physical coding experience is expected to enhance their career skills and communication competencies, respectively.

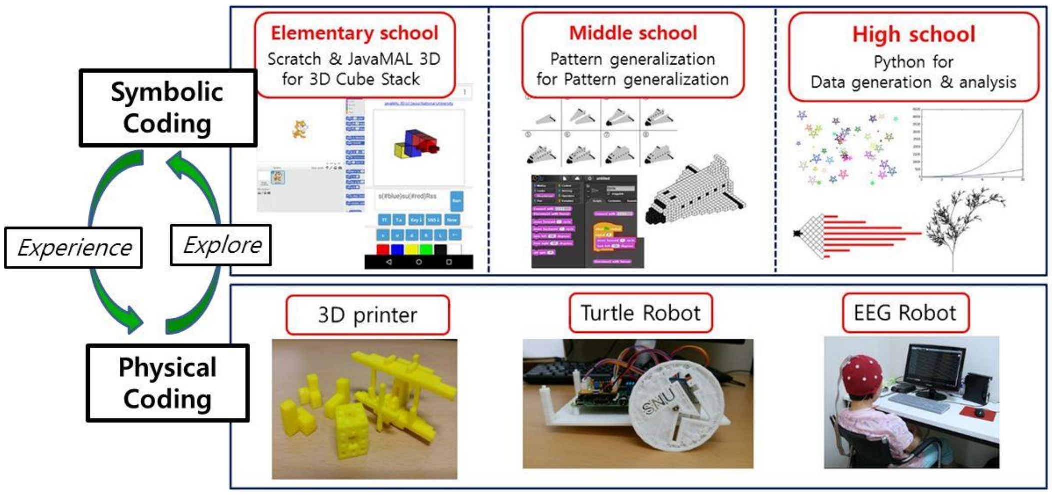

Symbolic Coding
Experience
Explore
Physical Coding
Elementary school
Scratch & JavaMAL 3D for 3D Cube Stack
Middle school
Pattern generalization for Pattern generalization
High school
Python for Data generation & analysis
3D printer
Turtle Robot
EEG Robot

Figure 1. Proposal for a coding curriculum

Keywords

Constructionism, Computational Thinking, Coding, Microworld, Mathematics education, Pattern Generalization, Mental tool

1. Coding Education in Korean School

1.1 Introduction

As three future skills of learners in the 21st century, the Partnership for 21st-century Learning (P21) stresses: learning and innovation skills; life and career skills; and information, media, and technology skills. The South Korean Ministry of Education is preparing for the 21st century with a focus on “SMART (Self-directed, Motivated, Adapted, Resource enriched, Technology skills) education”, and such a tendency has been reflected in the recently announced national curriculum revised in 2015. The curriculum presents the educational vision and the core competency as in Fig 2, and proclaims the three educational policies of SW education, free semester system[21], and character education:

Policy	Educational vision	Core competency
SW education	Fostering creative people equipped with the problem-solving skills to realize creative ideas as SW	Computational Thinking
Free semester system	Realizing a happy education that fosters students’ dreams and talents	Career
Character education	Making a happy society that fosters students’ dreams and talents through communication and respect	Communication

Figure 2. Educational vision and core competency

However, the application of these three educational policies has not been smooth. Originally scheduled for 2015, the introduction of SW education has been postponed to 2018. Although it is due also to external factors such as the insufficient number of SW-related teachers and equipment, the action of internal factors such as the insufficient establishment of specific directions and contents might be much critical. In addition, it has been pointed out that the selective programs in the free semester system and character education consist mainly of fragmentary education, thus failing to present a continuous process, and are insufficiently linked to the curriculum.

The study presents constructionism-based coding education[22] program in order to overcome the limitations of the three educational policies. The goal of the coding education, we present, is the improvement of mathematics-related CT competency, which has been evaluated as an essential competency in the 21st century, together with Reading, wRiting, and aRithmetic (3Rs). (e. g., Wing, 2006; Bundy, 2007). The goal must be linked to acquire not only simple programming skills but also problem-solving strategies. Consequently, the problem of insufficient contents will be resolved and the ambiguous direction will be concretized by curriculum-related career activities and group project-centered exploration in the context of career and character education, respectively.

[21] ‘Free semester system’ in Korea, where students take a course selectively, is the system similar to ‘Transition Year’ in Ireland and ‘Gap year’ in England.

[22] Because “SW education” can be mistaken for the training only of computer engineers, the present researchers use the expression “coding education” instead.

Therefore, the constituting principles of constructionism-based coding education, which emphasizes ‘learning by design’, were established as follow:

1. Learners’ CT competencies are to be strengthened through problem solving in the curriculum through coding.
2. Career education and character education are to be derived by assigning to learners the roles of individuals in society.

In the reflection of principles above, a coding education curriculum on each level of school was constructed and applied, as in Fig 3. Through the curriculum, learners can acquire mathematics-related CT competency and mental tools. In the ‘microworld’ presented as a tool, students begin exploratory activities through symbolic coding. The exploration is expanded to meaningful experiences through physical coding with 3D printers and turtle robots. This experience in turn leads to symbolic coding on the next level, thus resulting in learning on a higher level. For example, the exploration of activities with 3D cube stacks at primary schools are linked to career experience related to 3D printers, and these experience in turn serve as a mechanism for further enriching the exploration of pattern generalization at middle schools:

	SW	Curriculum	Career Exploration	Assessment
High school	Python	Data generation analysis	Programing Physical coding	Computational Thinking -Abstraction, Automation
Middle school	Snap!, JavaMAL(3D)	Pattern Generalization	EEG Robot	
Primary school	Scratch, JavaMAL(3D)[23]	3D CubeStack	3D Printer Turtle Robot	

Figure 3. Coding education

The next section will introduce a case of actually applying the education programs to a middle school and discuss the direction of coding education at primary and high schools.

2. Math-based coding education for Korean middle school

2.1 Pattern generalization with computational thinking

Pattern generalization is selected as the contents of coding education at middle schools. It stems from the exploration of activities with 3D cube stacks and experience related to 3D printers at primary schools. JavaMAL (3D), a microworld based on ‘learning by design’, is an educational tool for learners at primary and middle schools to enhance CT with simple action symbols, which are the conversion of action commands to overcome the difficulty of Logo, a programming language (Cho, et al., 2010). The students perform symbolic coding in JavaMAL (3D), where 3D structures are designed through the six basic commands of forward (s), backward (t), right (r), left (l), upward (u), and downward (d):

[23] JavaMAL microworld : http://www.javamath.com

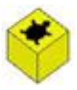
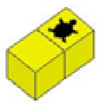
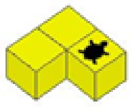
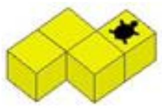
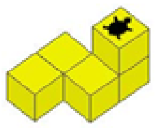
					
initial state	s	s	r	s	u

Figure 4. Basic commands in JavaMAL (3D)

Experience with 3D printers in relation to 3D cube stacks at primary schools is linked to making constructs richer and more elaborate at middle schools, thus leading to the exploration of pattern generalization including substitution process with variables. The exploration of pattern generalization naturally occurs during design activities, as in Fig. 5.

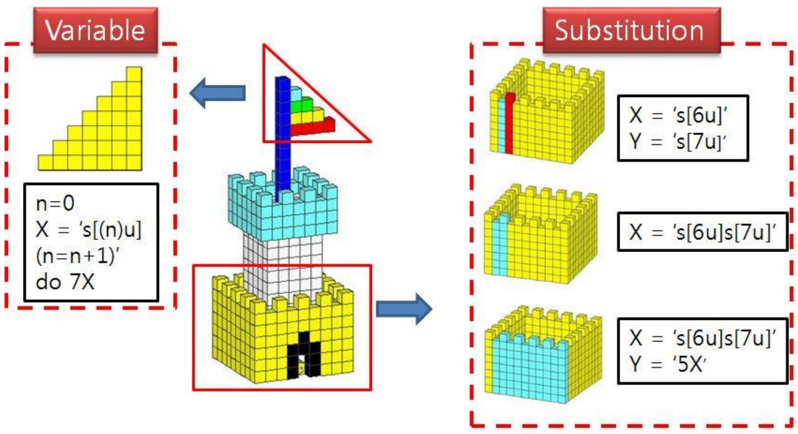


Figure 5. Learning pattern generalization by design

Fig. 5 shows a representative example for students to figure out the power of substitution with variables themselves, which is a process of the abstraction in CT. In the process of creating castle walls, students come to feel the need for substitution so as to represent repetitive objects.

Beyond the construction of simple objects by piling 3D cube stacks up one by one at primary schools, the process of establishing repetitive parts as something like the Lego bricks in Fig. 6, and expressing them automatically is one that well shows abstraction and automation, which are key to CT (Wing, 2006). Lego brick metaphor serves as a mental tool which is related to substitution, which perceives repetitive objects as a pattern and expresses it as a single symbol.

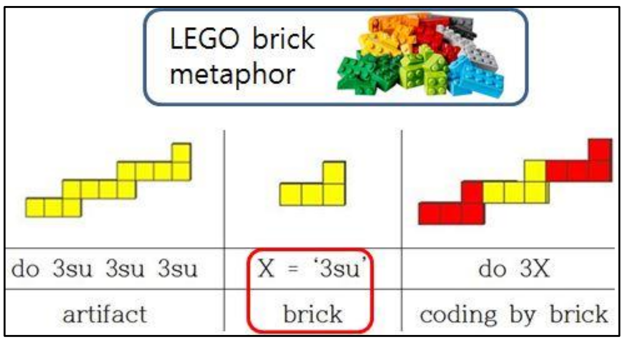


Figure 6. 'LEGO brick' metaphor(Jeong, 2014)

The flag-creating activity in Fig.5 enables students to recognize growing patterns and is related to the introduction of the variable n. Here, the expression "n = n + 1" can also serve as a mental tool related to sequences.

2.2 CT (abstraction-automation) based assessment

In addition, an assessment plan was constructed. In particular, CT competencies in the coding curriculum of middle schools were thus classified into the levels of usable skills and fluency in those skills. This can be revised and applied in the future according to diverse CT goals:

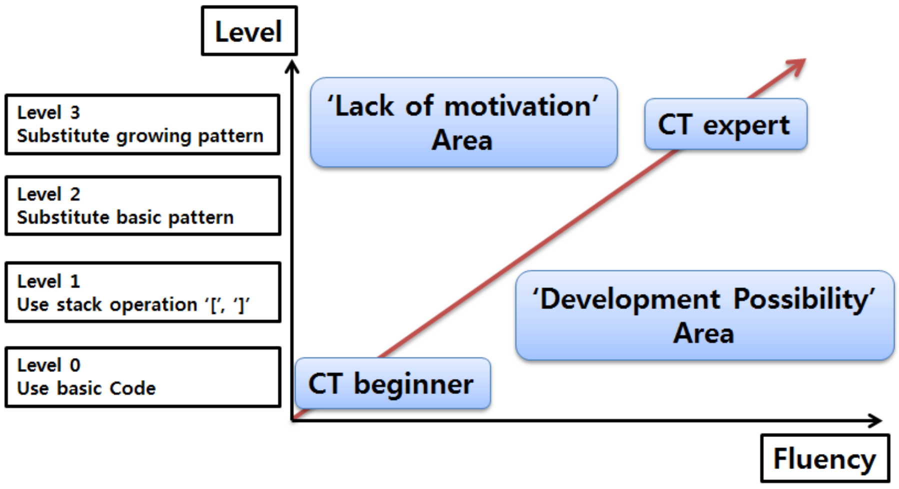
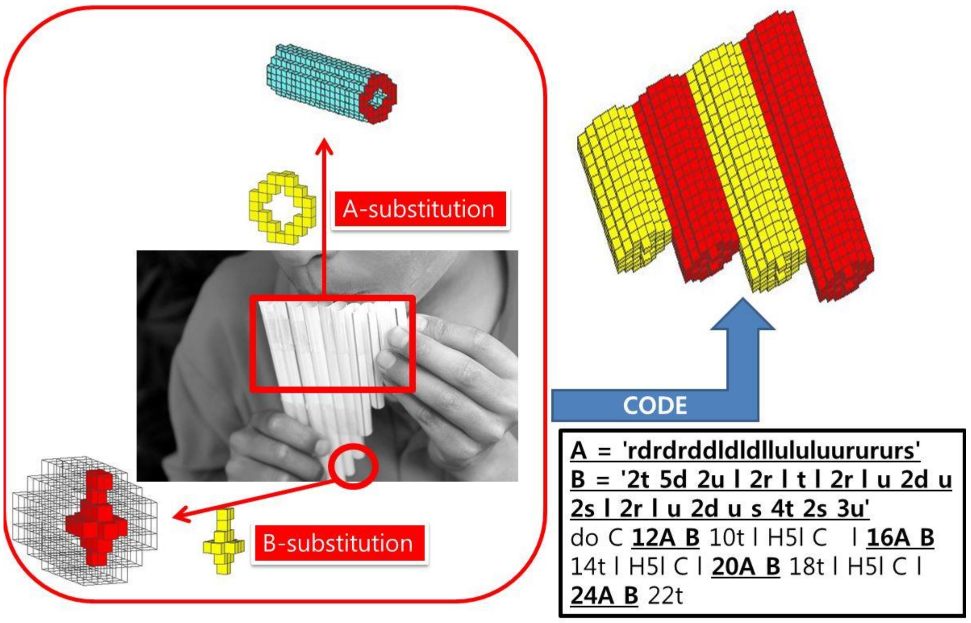

Level
Level 3
Substitute growing pattern
Level 2
Substitute basic pattern
Level 1
Use stack operation '[', ']'
Level 0
Use basic Code
'Lack of motivation' Area
CT expert
'Development Possibility' Area
CT beginner
Fluency

Figure 7. CT assessment tools.

Assessment is used to check and to provide feedback on learners' achievement of educational goals. The coding environment prompts students to be positioned in the Development Possibility Area in Fig. 7. In the area, students become an expert, if they experience difficulties in coding. Below is a work by a student positioned in the Development Possibility Area:



```
A = 'rdrdrddldldllululuururururs'
B = '2t 5d 2u l 2r l t l 2r l u 2d u
2s l 2r l u 2d u s 4t 2s 3u'
do C 12A B 10t l H5l C   l 16A B
14t l H5l C l 20A B 18t l H5l C l
24A B 22t
```

Figure 8. Straw pipe coding

The student in Fig. 8 can be seen as positioned in the level 2 Development Possibility Area. Below is an excerpt from a related interview:

'…Because I used many blocks, it was **impossible to draw without substitution**. So, I substituted two parts and used them a lot. I substituted A and B and shortened the commands considerably. Also, I would have been able to shorten them even more if I'd **used function expressions**.'(Jeong, 2014)

By mentioning the usefulness of substitution, the student above reflected that she had failed to use function expressions. In other words, the student can be seen as possessing the possibility of development because she feels the need for growing patterns demanded by level 3.

2.3 Coding education for character education and career

As shown by changes to the manufacturing industry stemming from 3D printers and the development of the Internet of Things (IoT), coding is closely related to the field of occupations. Pattern generalization education at middle schools goes beyond symbolic coding and includes a series of physical coding experience including robot design, activation command coding, and driving experience through brainwaves, as in Fig. 9:

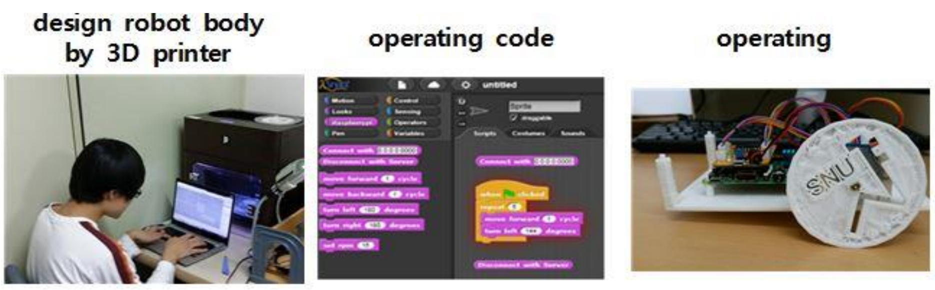
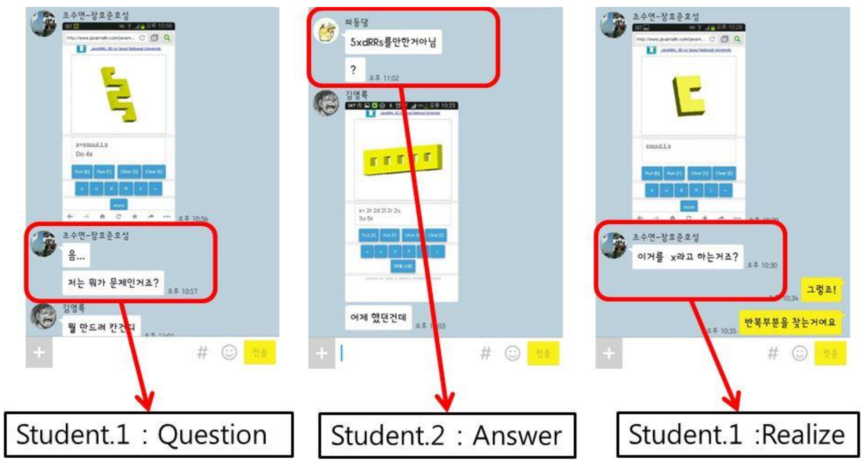


Figure 9. Robot experiences

In addition, with the possibility for individuals to communicate with the world thanks of the development of the Internet and social media, correct communication competencies have been mentioned as a key goal of character education. Since CT education enhances not only individual CT competencies but also social communication competencies, activities naturally performed in the course of implementing coding project — such as clicking on the "Like" button, posting remarks to others' works, and reconstructing others' works with citation of their sources — play an very important role in character education as well.


조수연-장호준호성
x=ssuuLLs
Do 4x
조수연-장호준호성
음...
저는 뭐가 문제인거죠?
김영록
뭘 만드려 칸건디

파동댕
5xdRRs를안한거아님
?
김영록
x= 2r 2d 2l 2r 2u
3u 5x
어제 했던건데

조수연-장호준호성
ssuuLLs
조수연-장호준호성
이거를 x라고 하는거죠?
그렇죠!
반복부분을 찾는거여요

Figure 10. Communication through social media

The present researchers provided students with a smart coding learning environment so that they could share and communicate the works that they had coded through social media, as in Fig. 10. In the process, students came to present opinions on and to critique freely one another's works, which could yield an effect as communication-centered character education where one helped, expressed appreciation for, and worked with others.

3. Connection from primary via middle to high school programs

Since coding education was designed in careful consideration of the connectivity of exploration and experience, the educational programs for middle schools, presented in section 2, are constructed in continuity with programs for primary and high schools. Just as experience with 3D printers at primary schools become a topic of exploration at middle schools, experience with EEG robots at middle schools in turn become a topic of exploration at high schools.

At high schools, students generate and analyze data necessary for activating EEG robots with brainwaves. Here, the present researchers present Python, a programming language, as a tool for dealing with data. Python that starts with turtle modules is linked to the exploration of microworlds at primary and middle schools, thus serving to lower the entry barrier for students in performing exploration at high schools:

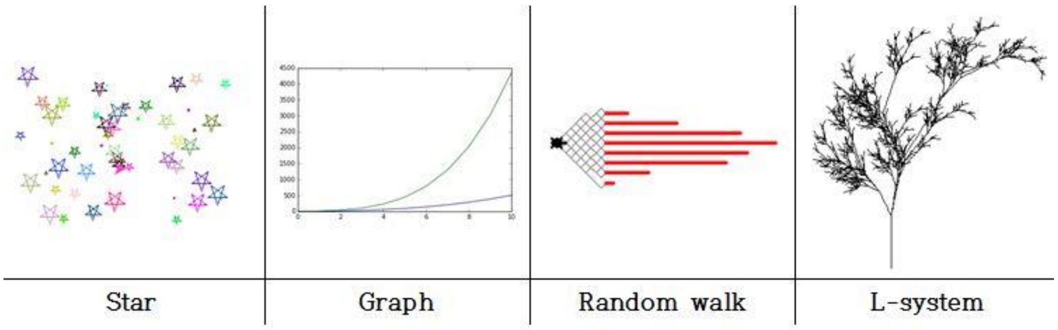
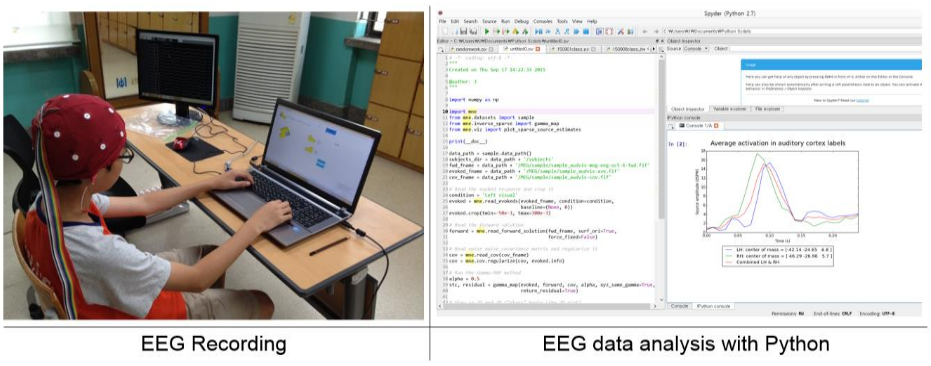


Figure 11. Turtle module in python

The researches using robots and 3D printers so far were attempts to bring Logo turtles out of the computer. The attempts are inevitable in today's education, which stresses the IoT and coding. Consequently, tool with regard to constructionism must become more diverse. Students can form mental tools not only on personal computers (PC) but also in all coding-related activities


EEG Recording
EEG data analysis with Python
Spyder (Python 2.7)
Average activation in auditory cortex labels

```
# -*- coding: utf-8 -*-
"""
Created on Thu Sep 17 14:22:33 2015

@author: J
"""

import numpy as np

import mne
from mne.datasets import sample
from mne.inverse_sparse import gamma_map
from mne.viz import plot_sparse_source_estimates

print(__doc__)

data_path = sample.data_path()
subjects_dir = data_path + '/subjects'
fwd_fname = data_path + '/MEG/sample/sample_audvis-meg-eeg-oct-6-fwd.fif'
evoked_fname = data_path + '/MEG/sample/sample_audvis-ave.fif'
cov_fname = data_path + '/MEG/sample/sample_audvis-cov.fif'

# Read the evoked response and crop it
condition = 'Left visual'
evoked = mne.read_evokeds(evoked_fname, condition=condition,
                          baseline=(None, 0))
evoked.crop(tmin=-50e-3, tmax=300e-3)

# Read the forward solution
forward = mne.read_forward_solution(fwd_fname, surf_ori=True,
                                    force_fixed=False)

# Read noise noise covariance matrix and regularize it
cov = mne.read_cov(cov_fname)
cov = mne.cov.regularize(cov, evoked.info)

# Run the Gamma-MAP method
alpha = 0.5
stc, residual = gamma_map(evoked, forward, cov, alpha, xyz_same_gamma=True,
                          return_residual=True)
```

Figure 12. EEG with Python

The present researchers' studies on the convergence of physical coding and constructionism will continue in the future as well. Recently, the present researchers have focused on driving EEG robots that are activated through brainwaves and, to accomplish this, analyzed brainwave patterns by using Python, as in Fig. 12. This undoubtedly will be a meaningful activity in constructionism, being not only a complex and creative exploratory process capable of encompassing all of the curricula presented in the present study but also an activity related to data mining skills, which are essential for the era of big data.

References

Bundy, A. (2007). *Computational thinking is pervasive.* Journal of Scientific and Practical Computing, 1(2), 67-69.

Cho, H. H., Kim, H. K., song, M. H. & Lee, J. Y. (2010). *Representation systems of building blocks in Logo-based microworld.* The Constructionism 2010 Conference. Paris: AUP

Jin Hwan Jeong (2014). *Mathematics Education of Pattern Generalization by Computational Thinking Game*. M.ed dissertation**, Seoul University, Seoul, Korea.**

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. Cambridge, Massachusetts: Perseus Publishing.

Papert, S. & Harel, I. (1991). *Situating Constructionism.* First chapter in Constructionism.

Papert, S. (1996). *An Exploration in the Space of Mathematics Educations*. International Journal of Computers for Mathematical Learning, 1(1). 95-123.

Papert, S. (1999) *The Eight Big Ideas of the Constructionist Learning Laboratory*. Unpublished internal document. South Portland, Maine.

Piaget, J (1965), *The Child's conception of Number*, W. W., Norton & company, Inc.

Wing, J. M. (2006). *Computational Thinking*. COMMUNICATIONS OF THE ACM 49(3), 33-35

This article is released under the CC-BY-ND license.

Music Blocks: A Musical Microworld

Walter Bender, walter@sugarlabs.org
Sugar Labs, a member project of the Software Freedom Conservancy

Devin Ulibarri, devin@devinulibarri.com
New England Conservatory Preparatory School and YMCA Malden

Yash Khandelwal, khandelwalyash51294@gmail.com
International Institute of Information Technology, Hyderabad

Abstract

Music Blocks software is designed for teachers and learners to explore the fundamental concepts of music in a visual-coding environment. Music Blocks is both innovative and beneficial to music education in a number of ways: On the one hand, it is a new method for understanding the fundamental concepts of music; on the other, it is a tool for learning coding and logic skills. It integrates both music and STEM fundamentals in a fun, scalable, and authentic way. Lastly—and very importantly—the tool itself is Free/Libre Software, which we argue is the best choice for an equitable and just education because it gives students the freedom to study, without restriction, both how to use the software and how the software itself works, i.e., how it transforms their programmed instructions into their musical inventions.

Keywords

Logo, Constructionism, Programming, Music

1. Introduction

1.1 Music Box: Voices from the past

In 1970 Edward Fredkin and Marvin Minsky designed and built the first digital synthesizer (Fredkin and Minsky 1970), which was commercialized under the name Triadex Muse. Triadex Muse was not intended to just be performed, but also programmed. It came with both the Triadex Muse Manual (http://trovar.com/muse/book.html) and the Triadex Muse Programming Guide (http://trovar.com/muse/book2.html), which includes a section on music theory. It is reasonable to infer that Fredkin and Minsky intended that the Triadex Muse present an opportunity for creating music while learning music theory and programming.

In a related effort, Jeane Bamberger and Terry Winograd built a Logo programming language interface to Minsky's Music Box. The software helped learners make connections between music ideas by actively making music (Bamberger 1991) A 1972 video of children using Music Box is available at https://www.youtube.com/watch?v=xMzojQFyMo0. Papert and Cynthia Solomon describe the interaction between learner and computer as a “conversation” in words or numbers [and music]. “Thing to do” #11 is “Make a Music Box and program a tune.” Rather than simply “printing” notes to the synthesizer, learners program songs using procedures attuned to musical structures (Papert and Solomon 1971).

In the example shown at the top of Figure 1, *Frère Jacques* (a nursery rhyme of French origin) is programmed by transcribing each note individually. Pitch is in semitones and represented by a letter of the alphabet (i.e., A = tonic, C = second scale degree, E = third scale degree, etc.). Rhythmic duration is achieved by adding more of the same pitch together in a string. One flaw in this design is that it is not obvious that there are two distinct parameters for each note. A second flaw is that it limits one’s ability to specify unique rhythmic values, such as triplets, etc. Papert and Solomon describe this as a “very bad” design for the program. Papert and Solomon assert that a “better way [is to use] descriptions of music in a good notation”. In the bottom example (which uses the “SING command”), the music is structured around the ideas behind pitch, rhythm, and phrasing, as well as higher-level concepts such as repeat and loop. The goal of Music Box was not just synthesis, but to engage the learner in constructing knowledge about music through programming.

```
TO FREREJACQUES
1 PRINT “AAA!CCC!EEE!AAA!AAA!CCC!EEE!AAA!EEE!FFF!HHHHHH!EEE!FFF!HHHHHH!...
END
```

```
TO FRERE1
1 SING MUSIC OF “1! 3! 5! 1!” “2 2 2 2”
END
TO FRERE2
1 SING MUSIC OF “5! 6! 8!” “2 2 4”
END
TO FRERE3
1 SING MUSIC OF “8! 10! 8! 6! 5! 1!!” “1 1 1 1 2 2”
END
TO FRERE4
1 SING MUSIC OF “1 -8! 1!” “2 2 4”
END
```

```
TO FREREJACQUES
1 FRERE1
2 FRERE1
3 FRERE2
4 FRERE2
5 FRERE3
6 FRERE3
7 FRERE4
8 FRERE4
9 FREREJACQUES
END
```

Figure 1 Two ways to play Frère Jacques *using Music Box (from Papert and Solomon 1971): (top) individual notes are sent to the synthesizer by concatenating the individual note values into a their total durations; (bottom) the SING command takes two familiar arguments—semi-tones and note durations.*

1.2 Forty years later: Voices from the present

The growth in sophistication of synthesizers has largely paralleled the growth of computing; digital music has become mainstream. Programming languages such as Barry Vercoe's Csound (Vercoe 1986) give unprecedented control over music. Sophisticated tools and apps have been developed on these platforms. Music professionals have access to tools almost unimagined in the early 1970s. But in regard to elementary education, those who were the target users for Logo and Music Box, the advances in music have been less pronounced.

There are sophisticated music applications targeting children that aim to help a novice compose music. Their user interfaces are intuitive and approachable, but they provide little access to tools essential to explore underlying musical structures beyond reusing precomposed phrases.

Csound has been used as the underlying engine for many music applications, but its design is intended neither for children nor beginning programmers. There are a plethora of programming environments for children, notably Scratch (http://scratch.mit.edu), which do provide some easily accessible mechanisms for exploring music. However, when it comes to exploring music, these programming environments have notable drawbacks. When programming music in Scratch software, for example, there is a block to generate a tone of a specific frequency and duration (in some extensions of Scratch these tones can be specified using MIDI notation), but any higher-level musical manipulation tools, even ones that are essential to music, must be programmed entirely from scratch. While it can be fun to program music with Scratch, the approach taken tends to resemble the “very bad” example described by Papert and Solomon as they lack the mechanisms to create larger, concrete musical structure.

There are a variety of tools used routinely in music education. Tools popular for teaching and learning musical dictation and ear training tend to be limited to music dictation and have no creative component. Furthermore, they expect that the end-user is proficient in reading Western music notation—there are no representations for learners who may be unfamiliar with notation. Tools popular for transcription and musical engraving (music notation) are used to create sheet music with affordances for composition. A variety of representations are available (such as guitar tablature and chord charts), which may be more useful to those more familiar with these systems. However, it would be daunting to use this software to explore music's rudiments in a creative and straightforward way. Tools popular for sampling and recording let you play, mix, and record songs/pieces. There is no requirement that the user be versed in language specific to music.

However, none are geared towards open-ended exploration of the fundamentals of music. Furthermore, many of these tools are proprietary—students are not able to view or study the internals of the tools and they may be "locked in" to using a proprietary tool in the future.

1.3 Microworlds

Papert used the term "Microworld" to describe the world of geometry explored when children used Logo. A microworld is a "subset of reality or a constructed reality whose structure matches that of a given cognitive mechanism so as to provide an environment where the latter can operate effectively. The concept leads to the project of inventing microworlds so structured as to allow a human learner to exercise particular powerful ideas or intellectual skills." (Papert 1980)

In a microworld, an individual is able to use a technological tool for thinking and cognitive exploration that would not be possible without the technology. But not just any technology. "The use of the microworld provides a model of a learning theory in which active learning consists of exploration by the learner of a microworld sufficiently bounded and transparent for constructive exploration and yet sufficiently rich for significant discovery." (Papert 1980)

There are some precedents for microworlds of music. Evgenia Sendova used a Logo microworld to enable learners to test hypotheses about musical phenomenon such as melody, rhythm, and harmony (Sendova 2001). Viera Proulx created a Java library that is used to design classes that represent musical phrases, melodies, chords, and effects (Proulx 2010); this addition of musical structure led students to add musical effects and to manage more complexity in their programs.

Music Blocks provides a collection of technological tools specific to a microworld of music, starting with pitch and rhythmic note values, but also providing affordances for repetition, transposition, etc. Music Blocks is more than an interface to a synthesizer and more than a transcription/engraving tool—it is a scalable and modular collection of essential building blocks which are at the crux of all powerful ideas in music.

Minsky remarked, "Until you understand something more than one way, you don't really understand it." Music Blocks encourages children to explore multiple ways of understanding concepts in music. (With programming, as with music, there is rarely one "right way" to do something.) The very recognition that there are multiple ways to solve problems gives learners a deeper understanding of the world around them. Music Blocks gives the learner the ability to seamlessly connect ideas across domains: music, graphics, mathematics, sensors, etc.

With Music Blocks, there are no black boxes; the learner is given full liberty to view and to study what something does and how it does it. The source code to every block is available and the license and means to modify or extend the language are provided. Taking something apart and reassembling it in different ways is a method to understanding it.

Music Blocks users have the freedom to run, copy, distribute, study, change, and improve the software. By using Music Blocks, they become members of a community that openly shares programs they have written. Any one who uses the software has the right to use the program as-is, or to adapt it, without needing permission. Free/Libre Software is a culture that encourages every child to be a creative force within their community, and to take ownership—and the responsibility that comes hand-in-hand with ownership—of the tools that they use for learning.

2. Power Piece: An approach to understanding music

Learning with Music Blocks is based on an approach known as "power peice". A power piece is a melody or a song that is taught because it is powerful and becomes more powerful as it is taught. A power piece is authentic, archetypal, and timeless, yet historically relevant, integrative, and infinitely malleable. *Twinkle-Twinkle Little Star*, a popular English lullaby sung to the tune of the French melody *Ah! vous dirai-je, maman,* published in 1761, is a classic power song. Mozart recast it as a highly embellished theme in his set of *Twelve Variations*. Since its inclusion as a Suzuki-method staple, it has become even more widely known and transformed, further expanding its potency. Its simple and elegant design allows for embellishment, refinement, and pedagogical repurposing (e.g., to teach new rhythms, as a vehicle for learning harmony, etc.).

The *power piece* approach differs from just learning more repertoire. Musical pieces are chosen, studied, and manipulated as a means to deepen one's understanding of musical concepts. For example, studying even the first section of the jazz standard *Autumn Leaves* (originally *Les feuilles mortes,* published in 1945 by Joseph Kosma), in all twelve keys in various ways (melody, harmony, guiding tone lines, with improvisation) would deepen a learner's understanding of the archetypal and important minor cadence in jazz.

Music Blocks brings the building blocks of music to the fore. It starts with music's most fundamental parameters: pitch and rhythmic note values. Combining the two parameters creates a note with a definite pitch and duration. Ordering notes in a sequence creates melody. Stacking different notes of the same rhythmic note value at the same point in time creates harmony. Once a melody has been created (with or without chords), "chunks" can be created and combined to create higher-level musical forms. Building in complexity, Music Blocks software provides blocks to manipulate pitch and rhythmic note values through repetition, transposition, note value augmentation and diminution—just to name a few. These tools, because of their inherently fundamental nature, become a powerful metaphor when scripting power pieces.

Take, for example, *Frère Jacques* which exemplifies the following musical concepts: perpetual canon/round (each sub-phrase can be performed with any other sub-phrase to create pleasant harmony in a major key); repetition (each sub-phrase repeats twice); and scale degrees 1–6 of a major scale. When tasked to dictate *Frère Jacques* with Music Blocks, a resourceful user will do the following: (1) Put repeat blocks around each chunk so that they repeat; (2) Create and save each of the four sub-phrases as "chunks" that can be ordered, reordered, and manipulated; (3) Find a way—through adding rests or by staggering the order of the chunks—to achieve the harmonious effect of the canon that is exemplified by *Frère Jacques; and* (4) Further experimentation and discovery of possibilities; e.g. Does it sound good with three repeats? What if one voice repeats three times while the others repeat twice? What happens when I transpose a voice or two or three? The possibilities are endless.

This type of problem solving is not very different from what a student of music theory might do to analyze a piece of music. Music Blocks places all of the analytic tools in the foreground and makes them easily accessible with minimal music-specific jargon.

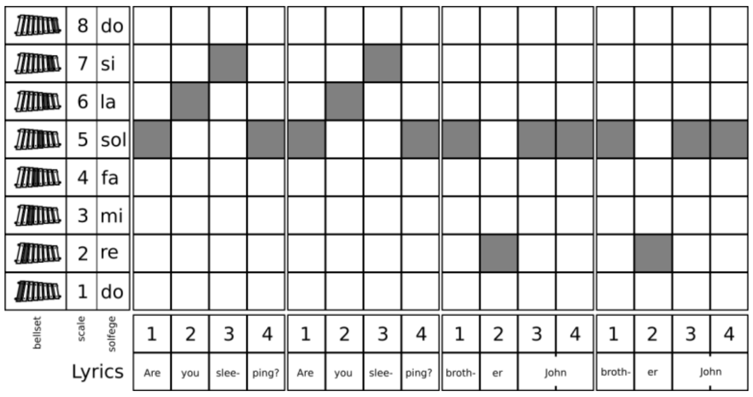
3. Music Blocks

Turtle Blocks (Bender, et al. 2015) is an activity with a Logo-inspired graphical "turtle" that draws colorful art based on snap-together visual programming elements designed to run in a web browser. Music Blocks (http://walterbender.github.io/musicblocks) is a "fork" from Turtle Blocks (i.e. built from Turtle Blocks' code). It also runs in a browser. It expands upon Turtle Blocks in that it has a collection of features relating to pitch and rhythm. Many of the ideas in Music Blocks were inspired by the music-in-education ideas, representations, and practices developed by Lawrence Scripp (Scripp et al. 2014).

In the sections that follow, we describe some of the numerous extensions we made to Turtle Blocks in order to provide affordances to access powerful ideas in music: what makes Music Blocks a Musical Microworld, as opposed to yet another block-based programming language.

3.1 The graphical-notation matrix

We chose to use a matrix (cartesian coordinate grid for pitch value vs. note duration) for two reasons: (1) because not everyone who uses Music Blocks is expected to be proficient at Western musical notation; and (2) the matrix representation allows for flexibility and scalability.



bellset	scale	solfege	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
	8	do																
	7	si			■				■									
	6	la		■				■										
	5	sol	■			■	■			■	■		■	■	■		■	■
	4	fa																
	3	mi																
	2	re										■				■		
	1	do																
Lyrics			Are	you	slee-	ping?	Are	you	slee-	ping?	broth-	er	John		broth-	er	John	

Figure 2: A matrix used for learning how to perform two portions of the melody Frère Jacques *on bellsets. This handout was used at the MusicLaunch community music program at Boston Chinatown Neighborhood Center as a tool to help beginner students collaborate with more advanced students in a performance of this well-known round. The three columns on the left are helpful in understanding how solfege, numbering, and bellset-location are all representations of the same concept.*

If you imagine that the collection of pitches represent a musical bell set, then adding/removing pitches is like adding or removing keys to a bell set. This has the benefit of focusing the musical creation process into a particular pitch-space (e.g. "key" or mode) for their intended sound world. Too often, music software programs present the user with a large keyboard of notes, which provides all of the "wrong" (unintended) notes alongside all of the "right" (intended) notes indiscriminately and side by side. Plus, most singable melodies fall within a register of less than an octave and singing one's musical creations is very good ear-training. The graphical-notation matrix is designed for this type of mindful, creative practice (See Figure 2).

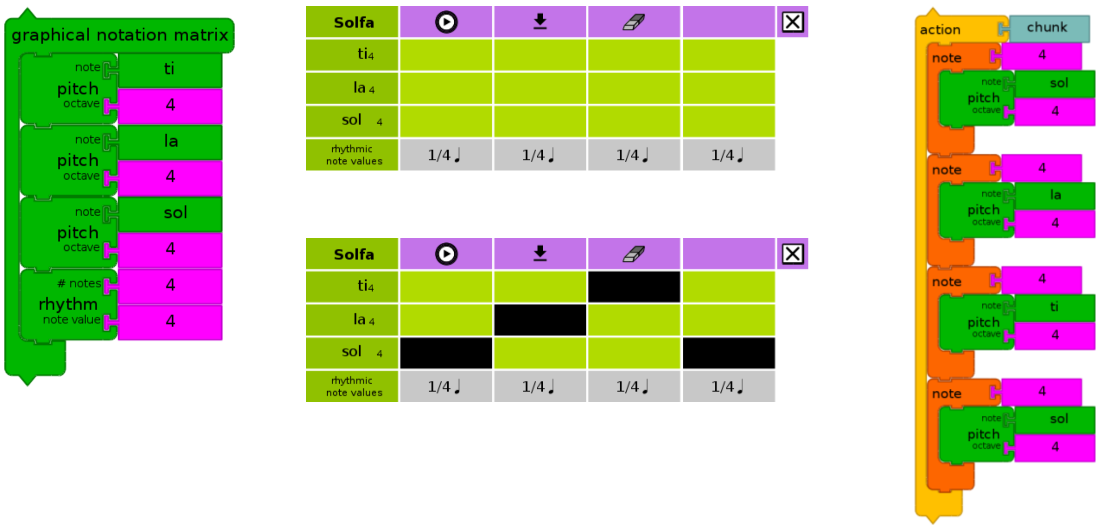


Figure 3: On the left is a stack of blocks used to generate a graphical-notation matrix (three pitches in octave 4 and four quarter notes). In the center top is the resulting matrix. In the center bottom, several notes have been selected in a matrix; On the right, a stack of blocks (chunk) generated from the matrix.

Having launched Music Blocks, a user starts by clicking on the *Graphical-notation Matrix* block (See Figure 3, left). A grid organized vertically by pitch and horizontally by rhythm appears (Figure 3, top). Because the matrix had three *Pitch* blocks, there are three rows, one for each pitch. (A row at the bottom specifies the rhythms associated with each note.) There are four columns for selecting notes because a *Rhythm* block was used to specifies four quarter notes.

By clicking on individual cells in the grid, the user will hear individual notes (or chords if she clicks on multiple cells in a column). In Figure 3 (center bottom), four quarter notes are selected (black cells). If the user clicks on the Play button (found in the top row of the grid), she will hear a sequence of notes played (from left to right, with their respective rhythmic values): sol, la, ti, sol (G4 E4 F4 G4). Once the user has a group of notes (a "chunk") that she likes, she clicks on the Save button (just to the right of the Play button). This will create a stack of blocks (See Figure 3, right) that can used to play these same notes programmatically (more on that below). The user can rearrange the selected notes in the grid and safe other chunks as well. Since she can define as many chunks as she wants, the user is free to experiment.

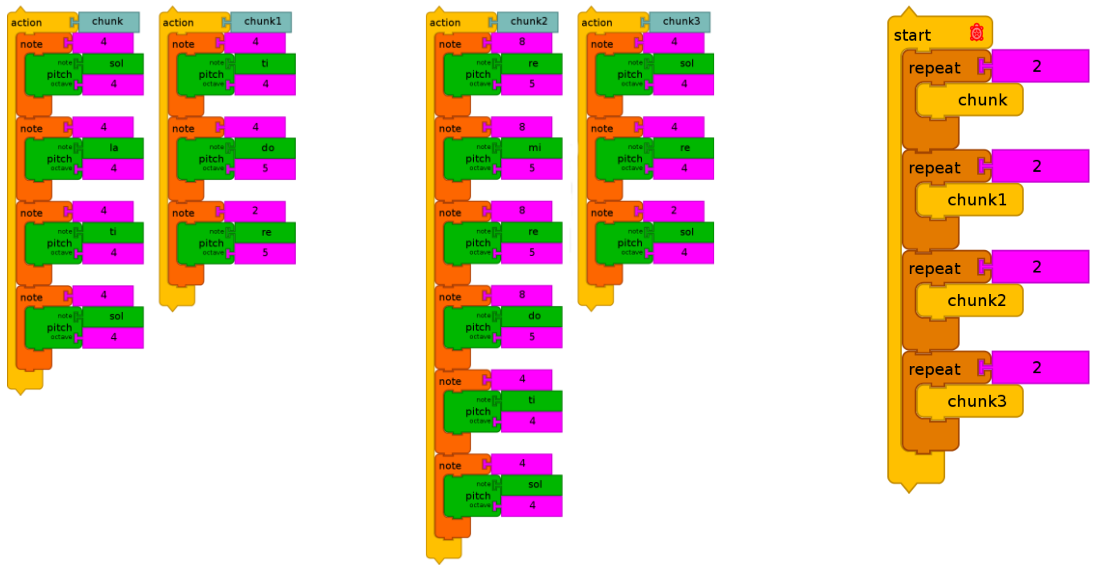


Figure 4: Frère Jacques *programmed in Music Blocks. Note the similarity to the Music Box example shown at the bottom right of Figure 1.*

The chunk created when the user clicks on the matrix *Save* button is a stack of blocks. The blocks are nested: an *Action* block contains *Note* blocks, each of which contains at least one *Pitch* block. The *Action* block has a name automatically generated by the matrix, in this case, chunk (the user can rename the action). Each note has a duration (in this case 4, which represents a quarter note). We encourage users to put different numbers in and see (hear) what happens. Each *Note* block also has a *Pitch* block (if it were a chord, there would be multiple *Pitch* blocks nested inside the clamp of the *Note* block). Each *Pitch* block has a pitch name (Re, Mi, and Sol), and a pitch octave; in this example, the octave is 4 for each pitch. To play the chunk, the user simply clicks on the action block (on the word "action"). She will hear the notes play, ordered from top to bottom with their respective rhythmic durations.

Pitch blocks are used inside the *Matrix* block to indicate pitches to select. They are also used inside of *Note* blocks to specify the pitch(s) to be played. The user can plug different values into the Pitch block slots for name and octave. The pitch name can be specified using a *Solfege* block (e.g. Do, Re, Mi); a *Pitch-name* block (e.g., C, D, E); or a *Number* block (frequency in Hertz). The octave is specified using a *Number* block and is restricted to whole numbers.

Rhythm blocks are used to generate rhythm patterns in the matrix. The top argument is the number of notes. The bottom argument is the duration of the note. Multiple *Rhythm* blocks may be used inside the *Graphical-notation Matrix* block. For more complex rhythmic patterns, *Tuplets* is a tool to create sets of rhythmic values that are not based on a power of 2. The user can mix and match *Rhythm* and *Tuplets* blocks when defining grids. The user can also specify individual notes and chords in the matrix.

3.2 Programming with music

The chunks created from the matrix are used for programming. (The user can also create or modify chunks by hand, without utilizing the matrix). Music Blocks creates a new block specific to each chunk. Clicking on, or running, this block has the same effect as clicking on the chunk. She can repeat chunks either by using multiple chunk blocks or using a *Repeat* block. She can also mix and match chunks. Adding a few more chunks results in *Frère Jacques* (See Figure 4).

3.3 Beyond Music Box

In provisioning our musical microworld, we have added tools that can be used to further explore music's fundamentals. Music Blocks includes many ways to transform pitch and rhythm: (1) For pitch transformation, we have tools such as *Sharp*, *Flat*, and *Transposition*. The *Sharp* and *Flat* blocks modify pitch to *Pitch and Note* blocks, or even chunks. The *Transposition* block can be used to make larger shifts in pitch to individual pitches or to all pitches contained within a chunk. To shift an octave up or down, transpose by ±12 half-steps. (2) For rhythmic transformation, we have tools such as *Rhythmic Dot*, *Multiply* or *Divide Beat Factor*, and *Rhythmic Tie*. To "dot" a note, a common musical transformation, use the *Dot* block. A dotted note extends the rhythmic value of contained notes by 50%. The *Multiply/Divide Beat Factor* blocks multiply or divide contained rhythmic note values by the factor defined by the user. (3) For exploring and transforming larger musical form, we have *Repeat* and *Duplicate* blocks. The *Repeat* block will play a sequence of notes or chunks multiple times in their given order; the *Duplicate* block will repeat each note specified. (4) Each *Start* block runs as a separate musical voice in Music Blocks; when you click on the “Run” button, all of the *Start* blocks created are run (performed) concurrently. This feature can be used to achieve musical counterpoint (multiple voices performed simultaneously with differing pitch and note values) to create complex harmony.

There are exciting possibilities when combining the math and logic functions of the original Turtle Blocks code with the Music Blocks extensions. For example, the user can program music which chooses random parameters—rhythmic note value, octave, pitch in Hz, etc.—to create moments of aleatory and improvisation in their compositions. Students can use math and logic to create the harmonic series and find different ways to program rhythmic dots and rhythmic ties.

Music Blocks also has tools that take advantage of a computer's sensors, which means that interactive performances are possible. For example, we could reuse the chunks from *Frère Jacques* as follows: when the mouse is in the lower-left quadrant, chunk0 is played; lower-right quadrant, chunk1; upper-left quadrant, chunk2; and upper-right quadrant, chunk3.

```
mouse = {
g'4 a'4 b'4 g'4 g'4 a'4 b'4 g'4 b'4 c''4 d''2 b'4 c''4 d''2 d''8 e''8 d''8 c''8 b'4 g'4 d''8 e''8 d''8 c''8 b'4 g'4 g'4 d'4 g'2 g'4
d'4 g'2}
\score { << \new Staff = "treble"
{ \clef "treble" \set Staff.instrumentName = #"mouse" \mouse } >> \layout { } }
```

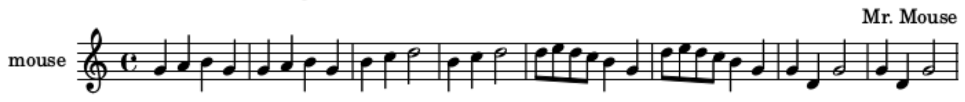


FIgure 5: The output of a Music Blocks program can be saved as a Lilypond format file (“mouse” is one of our default (and make-believe) musical instruments); the compiled Lilypond output.

A *Save as Lilypond* block transcribes a composition (http://www.lilypond.org/) (See Figure 5). Our goal in building a bridge between Music Blocks and Lilypond (which is also Free/Libre Software) is to give the learner both the ability to communicate with the mainstream world of music and access to a rich set of tools that they may use to further augment their exploration of music. Music Blocks, by design, does not confine a user to its tools—rather it is a tool to propel the ambitious learner to other rich and authentic musical discoveries.

4. Preliminary Conclusions

Why a microworld for music? According to Papert, a microworld must be sufficiently bounded and transparent for constructive exploration and yet sufficiently rich for significant discovery. Similarly, Music Blocks aims to be a musical microworld—an open-ended, yet powerful and musically relevant tool—one that invites learners of varying backgrounds to explore fundamental musical concepts that are both intrinsic to music yet transcendent of a specific discipline.

When discussing microworlds, however, Papert also speaks of the need for guidance, both in the microworld itself and in the teacher's facilitating a child's exploration of it. Therefore, workshop projects are being actively developed and refined to provide guidance and inspiration to teachers and learners in their use of Music Blocks. These projects use an approach that builds from the *Power Piece* approach mentioned above. Since *Power Pieces* introduce rich musical ideas that can be studied, analysed, transformed, and re-imagined, they are ripe for open-ended explorations as part of workshops. Students may, for example, be given a power piece to work on, along with guided inquiry questions that lead to investigation about music and music integration. Performances and presentations of musical creations at the end of a workshop should be playful, fun, and motivating. Facilitated dialogue after performances and presentations should be opportunities for critical thinking and reflection. The first examples of student programs written in Music Blocks demonstrate the efficacy of the microworld approach.

Why programming and why Free/Libre Software? Forty years of observation suggests that engaging children in using computers *and in programming software to run on computers* is a powerful means to drive learning. The process of writing and then repairing (also known as "debugging") a program, which Solomon described as "the great educational opportunity of the 21st Century", provides a basis for active learning through heuristic problem solving. Free Software is enabling student contributions to the Music Blocks design, documentation, and code.

References

Bamberger, J.S. (1991) *The Mind Behind the Musical Ear: How Children Develop Musical Intelligence*, Harvard University Press, Cambridge, pp. 104.

Bender, W., Solomon, C., and Urrea, C. (2015) "(More than) Twenty Things to Do in Turtle Blocks", *Proceedings of Constructionism 2015*, Vienna, Austria.

Fredkin, E., and Minsky, M. (1970) *Digital Music Synthesizer*, US Patent 3610801.

Papert, S. (1980). Computer-based microworlds as incubators for powerful ideas. In R. Taylor (Ed.), *The computer in the school: Tutor, tool, tutee* (pp. 203–210). Teacher's College Press.

Papert, S. and Solomon, C. (1971). "Twenty Things To Do With A Computer" *MIT AI Laboratory Memo* 248.

Proulx, V. (2010) Music in Introductory Object Oriented Programming, *Proceedings of Constructionism 2010,* Paris, France.

Sendova, E. (2001) Modelling Creative Processes in Abstract Art and Music, Eurologo 2001, *Proceedings of the 8th European Logo Conference* 21–25 August, Linz, Austria.

Scripp, L., Ulibarri, D., Southerland, S., Gilbert, J., Sienkiewicz, F., Swihart, A. N., and Swihart, E., (2014) *Principal Investigator's Final Report: The Oakland Unified School District's Music Integration Learning Environment (MILE),* Arts in Education Model Development & Dissemination (AEMDD) Project.

Vercoe, B. (1986) *Csound: a manual for the audio processing system and supporting programs with tutorials*.

Acknowledgments

The authors would like to thank Google for their generous support of this work through Summer of Code (GSoC) and Google Code-in programs and the students who participated. Marnen Laibow-Koser helped advise the GSoC intern. Glen Larsen advised on Lilypond. The Sugar Labs community has been very supportive of the project, as has been Eduardo Marturet. We would also like to thank Cynthia Solomon for encouraging us to write this paper.

On the Road to Sustainable Primary Programming

Ivan Kalas[1,2]
[1] *i.kalas@ioe.ac.uk;* London Knowledge Lab, UCL Institute of Education, London, UK
[2] *kalas@fmph.uniba.sk;* Department of Informatics Education, Comenius University, Bratislava, Slovakia

Extended Abstract of the Keynote

In recent years we witness a new intense outburst of interest in educational programming in several countries, including their primary or even pre-primary stages. Constructionist educational programming for everybody has been the main principle in our development and research since our first influential implementation of Logo programming environment for Windows in 1993 (Blaho et al. 1993). However, until recently we have not experienced systematic, nationwide, long term, thoroughly built, sustainable and properly supported integration of educational programming into formal school systems – in spite of the fact that several countries (including Slovakia) have declared such policy materialised in their (sometimes even mandatory) subjects named Computer Science, Informatics, ICT or Computing[24].

Therefore, it is natural and legitimate to ask whether this wave of interest in educational programming differs from the previous ones, which had never managed to reach the level of permanent presence in naturalistic classroom settings – especially in primary[25] classroom settings. In my presentation I will argue why there is currently a chance to achieve this and which factors I think must be thoroughly taken into consideration to build sustainable and broadly accepted educational programming at primary level (referred to as primary programming).

To start with, for primary programming to really work and sustain, it needs to ensure that:

(i) Learning it is relevant for the primary learners and leads to meaningful progress.
(ii) Primary schools (i.e. the school administration, teachers and indirectly also parents) are supportive and supported to adopt it.
(iii) External circumstances do not restrain it.

In my reflection I will focus on the first two requirements which we as researchers and developers have a potentially greater opportunity to influence. I will primarily draw on my current exceptional opportunity to engage with two different primary educational systems – that of England and Slovakia[26].

[24] using the most recent name of the new mandatory subject in England

[25] *According to OECD glossary, primary education (ISCED 1) usually begins at ages five, six or seven and lasts for four to six years (the mode of the OECD countries being six years). Programmes at the primary level generally require no previous formal education, although it is becoming increasingly common for children to have attended a pre-primary programme before entering primary education.*
The boundary between pre-primary and primary education is typically the beginning of the systematic studies characteristic of primary education, e.g., reading, writing and mathematics. It is common, however, for children to begin learning basic literacy and numeracy skills at the pre-primary level, see
stats.oecd.org/glossary/detail.asp?ID=5411

[26] with considerably different history of integrating digital technologies into school classrooms and different histories of computing and informatics subjects

In the presentation I will synthesise the experience accumulated at Comenius University and London Knowledge Lab in:

(a) being involved in developing Slovak curriculum and content for Informatics study programme for Slovak primary and secondary education[27],
(b) taking part in building and conducting corresponding nationwide professional development (PD) programmes for in-service primary and secondary teachers,
(c) being involved in developing software interfaces for primary and kindergarten children[28] to facilitate the development of their computational thinking and problem solving skills through programming,
(d) analysing cognitive requirements and difficulties of the programming tasks which primary pupils aged 8 to 10 are supposed to solve in the phenomenal international contest Bebras[29], see (Gujberova & Kalas 2013),
(e) being a part of the research and development of the ongoing UCL IOE ScratchMaths project, studying how educational programming in years 5 and 6 can contribute to the development of the pupils' mathematical thinking[30]. The project is being presented in detail in this conference by (Benton et al. 2016). In my presentation, however, I will focus on its design principles from the computing and computational thinking perspective.

Based on this I will try to (a) formulate an emerging *conceptual framework for primary programming* and (b) identify five *key principles* which I believe should be respected when developing and implementing new and sustainable primary programming content so that aforementioned requirements (i) and (ii) might be satisfied. Naturally, other researchers face similar questions. Duncan et al. (2014) explore the meaning and potential impact on learning programming for primary pupils and classify early programming learning environments into five different levels of age appropriateness, for children from 2 to 14 years. Meerbaum-Salant et al. (2013) and Armoni et al. (2015) investigate the use of Scratch immediately after the primary stage and report on the extent to which the computer science concepts were successfully learned.

My understanding of what educational programming is and why it is hard borrows from Blackwell (2002) who ascribes it to *abandoning direct manipulation*, *using specific formal notation* and *adopting abstraction as a tool for coping with complexity*. I also gain from Watt (1998) and his recall of Papert's syntonicity (1980), i.e. a strategy of a (young) programmer to identify with the controlled object, which in our ScratchMaths intervention design plays significant role.

In my reflection I will also borrow from Brennan (2015) who addresses *five tensions in supporting constructionist learning approaches* within the ScratchEd PD model, namely tensions between (i) tool and learning, (ii) direction and discovery, (iii) individual and group, (iv) expert and novices, and (v) actual and aspirational. Our ScratchMaths context is close to Brennan yet slightly different as our primary focus are pupils in years 5 and 6, though this naturally incorporates their teachers and corresponding PD development and deployment.

By synthetizing all previous design experience I will identify five key principles for designing sustainable primary programming content, calling them *five areas of respect*. I will present them in detail and illustrate them using some of the emerging design principles from our on-going ScratchMaths development, trialling and research.

1 Respect primary pupils

Most of the educational programming research in 80s and 90s focused on out-of-school, after-school or somehow selective settings, with some exceptions such as (Nikolov & Sendova 1985), (Sendova 1989), (Turcsanyi-Szabo 1997), (Rader et al. 1997), or our own development of a programming curriculum for Slovak lower secondary schools (Blaho et al. 1994). However, in that

[27] in Slovakia this means for pupils aged 6 to 10
[28] aged 3 to 6
[29] see http://www.bebras.org/, http://www.beaver-comp.org.uk/ and http://www.ibobor.sk/
[30] for pupils aged 9 to 11

particular project the content we developed was also deployed by us, Informatics education specialists – and the real and sustainable impact on ordinary in-service primary teachers and naturalistic classroom settings was at the best only trivial and indirect[31].

Recently however we have adopted different approach in our development for Slovak kindergarten children (Moravcik & Kalas 2012) and in our on-going ScratchMaths project in which we conduct the research and development in a rich sample of regular state primary schools – paying close attention to every pupil in the class. When trialling our programming intervention we act as nonparticipant observers of ordinary teachers who work with all pupils in their classes, in their regular school lessons. In our ScratchMaths intervention we strive to support all pupils, of all different achievement levels, facilitating high diversity in activities. For each investigation of each module we specify its learning objectives, check lists of its expected and additional practical achievements and also lists of corresponding assessment tasks.

This approach is still quite rare, if compared for example with a broad sample of current Scratch literature, for example book publications which I analysed and will report about in the presentation. I will also present some of our ScratchMaths design principles, by which we are trying to achieve appropriate learning outcomes for every pupil in the class, such as systematically encouraging the important principle of body syntonicity, including activity extensions to challenge more able pupils, discussing in pairs and reporting back, combining hands-on and unplugged activities etc.

2 Respect primary teachers

In the majority of educational systems primary teachers – not specialists in particular areas – teach all or most of the subjects. And yet in many primary programming research and pilot development projects the new content is being trialled by the developers themselves, often university Computer Science experts, ICT specialists or ICT secondary teachers. Currently, I consider this to be a risk factor and indication of likely unsustainability. In an alternative strategy the subject content specialists collaborate closely with the primary teachers within the iterative development stage already and the trialling and deployment stage is completely in the primary teachers' hands, who are often non-specialists and usually novices in primary programming. Moreover, based on anecdotal evidence from several countries, we cannot rely on future primary teachers' pre-service development as despite multiple attempts to integrate primary programming into this, there is still very little time dedicated to this aspect of the curriculum[32].

However, if we begin the whole deployment of the countrywide sustainable primary programming intervention with existing primary teachers, we have to anticipate and cope with their low initial confidence in elementary programing and potential difficulties to deeply understand most of the underlying computational concepts. As a consequence, this will also significantly influence the initial level of their constructionist pedagogy when running the new intervention – which will, fortunately, quickly increase as they start gaining confidence, experience and success with their pupils in the lessons, when deploying the content. To support this transition, the whole PD process (which will be typically rather short due to the time pressures of teaching) must convince teachers (a) that they will manage to conduct the intervention and (b) that the intervention itself will productively contribute to their teaching goals (in England, for example, will effectively support their pupils' mathematical thinking development).

In our pre-primary and primary developments, both in England and Slovakia, we try to achieve this ambition by properly understanding the educational milieu, setting appropriate learning goals and realistic selection of computational concepts, and by carefully building all possible bridges to other areas of primary education, especially mathematics, and also by applying a consistent constructionist approach when pupils are from the very first minute invited into opportunities to create meaningful products being scaffolded by their teacher.

[31] informing mostly only us, its authors

[32] in 90s and 00s we believed this was only a temporary and untenable resistance

The other three of my five key principles relate closely to what Brennan in (2015) refers to as the tension between the *tool* and *learning*. While Brennan negotiates this tension within the teachers PD context, I treat it within both professional development and everyday class teaching context. Thus, although I fully harmonise with Brennan's perception, I tend to go further and distinguish between *computing*[33] and *computational tool*[34] on the *tool* side of Brennan's tension, and thus reflect on how to negotiate it within the triangle of *computing* (what to learn), *tool* (what to use) and *learning* (how to learn). The following three principles of the sustainable primary programming intervention design result from our effort to negotiate productive balance among these three foci.

In accordance with Papert (1987) I appreciate that diverting towards a computational *tool* may result in unwanted technocentrism, while diverting towards *learning* may lead to shallow or no development of programming at all. However, I also believe that diverting towards *computing* may lead to another form of a technocentrism, a kind of 'computer-science-centrism', which might diminish the ultimate sustainability of the primary programming intervention. On the other hand, diverting from *computing* may easily result in minor or trivial transition within the computational thinking development.

3 Respect computing

Identifying appropriate computational concepts and methods and establishing productive approaches for how to mediate them to and reconstruct with primary pupils is up till now probably the least explored and defined area, although it is recently getting greater attention, see for example Duncan et al. (2014), Guzdial (2004) or Meerbaum-Salant et al. (2013). Most of the current programming environments explored in programming with young or very young learners focus on an actor which children control getting instant, usually visual feedback. And yet, the importance of the *body syntonicity* in the sense of Papert (1980) or Watt (1998) and corresponding pedagogies which support pupils in identifying with the actor they control is almost completely neglected (based on e.g. my Scratch literature analysis).

Although my goal in this presentation is to reflect on sustainable primary programming intervention *design principles*, not on its *specific content*, I find it useful to think about which opportunities are offered to learners from the computational perspective in this context, i.e. the conceptual framework of the most common content. Therefore in the presentation I will propose a vision of how to structure the space of primary programming content along *three distinct criteria*. This may increase understanding of the opportunities and powerful ideas of the primary programming. These criteria are *actor*, *control* and *interaction*:

- The *actor* to control, being either *physical* (e.g. a wooden toy, a programmable toy, robotic car,) or *virtual* (like Lightbot, virtual Bee-Bot, or a Scratch sprite).
- The *style of control*, being *(1) direct manipulation* (e.g. moving a toy by the hand or dragging a sprite by the mouse), or *(2) direct drive*, i.e. taking an action with immediate, visual and unambiguous reaction (e.g. pressing a button atop the Bee-Bot to make it turn right by 90 degrees or clicking an isolated move -50 steps block to make a sprite move backwards 50 steps), or *(3) programming* (i.e. building or working with an external representation of the future behaviour of an actor).
- The level of *interaction*, i.e. either controlling an *isolated actor* with restricted interaction, or controlling multiple interacting actors.

Almost all 'sub spaces' of the primary programming conceptual framework classified by these three criteria offer rich opportunities for thoughtful, well projected and systematic development of different basic computational concepts and methods. For example:

[33] or Computer Science or Informatics

[34] in the wide spectrum from digital programmable toys of the Bee-Bot type to programming languages like Lego WeDo or Scratch

- *Directly manipulating a physical actor* (e.g. navigating a toy in a hand through a toy town or navigating Bee-Bot in a hand atop a regular square mat with toy shops on it) gives rich opportunities to think about, plan, discuss and act a solution of a problem.
- *Directly driving a virtual actor* (e.g. a Scratch sprite by clicking one isolated block in the scripts area) gives unique opportunities to explore its functionality with different input values etc.
- *Planning future behaviour of a physical actor* (e.g. building a linear sequence of paper cut-outs with arrow commands printed on them to plan a trip for a bee on a grid maze mat to get to a flower) gives opportunities to plan that behaviour, analyse its properties like length, compare different solutions, modify it to avoid another obstacle, rerun it several times etc.
- *Planning future interactions of two virtual actors* (e.g. building parallel scripts for two sprites to react to an event of being clicked which should result in asking the other sprite to come along) gives opportunities to plan and explore communication and collaboration between multiple actors.

I will argue that identifying a set of distinctive criteria like the ones I listed above helps to better understand the conceptual framework of primary programming and better appreciate its richness when designing the computing interventions for primary pupils. I will also point out that neglecting the opportunities of different 'sub spaces' of this conceptual framework[35] may deprive primary pupils of constructing meanings of the key important computational concepts and procedures and thus of the sustainable development of their computational thinking.

Although there are different approaches how to structure key issues of computational thinking, see for example Wing (2008), Brennan & Resnick (2012), or the English National curriculum for Computing (DfE, 2013), in the *computing* node of our ScratchMaths development triangle we organize *the big ideas of computing* into five categories, inspired by Abelson and Sussman's *mechanisms for thinking about computational processes* (1986), namely

(1) *actors* and their attributes;
(2) *data* (storing, using and updating values);
(3) *commands and expressions*;
(4) *programs* (their structure and interpretation, including sequences, loops, conditions, conditionals, parallelism, randomness, events and broadcasting) and corresponding design practices;
(5) *abstraction* as a mechanism of giving names to compound structures and manipulating them as units.

When designing the ScratchMaths intervention and corresponding PD sessions in our project we use these five categories as an organisational framework for pupils and their teachers to experience, understand and appreciate the *mechanisms for thinking about computational processes*, for example:

- view individual commands as means to achieve isolated reactions,
- view commands as atomic blocks for building programs,
- view programs as explicit representations of future behaviour(s) of an actor or actors,
- view abstractions (new commands) as names referring to other programs (exploiting the sequence of steps: build a behaviour, debug it, name it, keep it, use its name instead of it),
- view broadcasting as an instrument for actors to communicate and collaborate.

4 Respect the tool

As I pointed out earlier, neglecting opportunities of the different areas of primary programming conceptual framework (as specified above by the *actor*, *control* and *interaction* criteria) may lead to shallow or (almost) no learning of programming, different misconceptions, or even loss of the interest in educational programming altogether. From the *tool* perspective of the *computing-tool-learning* triangle I will argue that discounting the level of the tool's developmental

[35] for example by inappropriate timing from the pupils age point of view – we will discuss this risk factor in more detail in section 4

appropriateness[36] and neglecting its computing affordances may as well result in wasting its potential to support the learning goals of primary educational programming.

What I observe happening in some settings at present is that primary teachers start using one particular tool, for example Scratch, in almost all or literally all year groups of the primary education. However, younger pupils (e.g. aged 5 to 7) may thus miss the opportunity to benefit from affordances of several other more age-appropriate computational tools, including programmable physical actors (like Bee-Bots etc.) or programmable virtual actors (like Scratch Jr. etc.). More than that, teachers often report that pupils prefer not to use the same tool for long periods, which may result in the situation when pupils in the higher years of the primary education – when a tool's affordances would be more appropriate, start losing their motivation to learn with that particular tool. Using any tool for too long may also increase the challenge of transitioning the key computational concepts from one tool to another and thus deepening pupils' understanding.

I will also argue that a similar *wrong-timing effect* may occur if a professional programming language (e.g. Python or similar) is introduced too early, in this case at any point in primary education. Clearly, more research is needed to better understand the concept of developmental or age appropriateness of different computational tools and their computing affordances and how to match them against corresponding computational concepts and different pedagogies (to be discussed in the last of our five 'areas of respect').

In the ScratchMaths development process we structure the elements of Scratch in the organizational framework of the following categories:

(1) *objects* (sprites, stage, data – i.e. anything controlled by blocks);
(2) *blocks* (atomic units of the language, representing either commands or expressions);
(3) *scripts* (compound representation of behaviours);
(4) *project* (its execution and management).

This framework is informing us in our design process when identifying the affordances of Scratch and engaging them in selecting the computational concepts and productive learning strategies to sustain the balance in the triangle. In the presentation I will illustrate some of these affordances, including:

- Explore the functionality of a block by clicking it as a single isolated block in the scripts area (or even in the blocks palette). This is possible even if the block does not represent a command but a condition or a value.
- Give meaningful names to new blocks, sprites, broadcast messages, variables or costumes (not restricted by rules common in most of the programming languages).
- Change the value of a variable using the change <variable> by … block instead of building an assignment command with an expression on its 'right side'.
- Use your own backdrops which contain visual items to support learning, like starting position, colours to inform the sprite, text notes, coordinates or certain positions like grid points etc.

5 Respect learning

The third node of the triangle represents the need to respect the teaching and learning strategies on both teachers and pupils side *"…with aspirations to disrupt technocentrism in the classroom"* (Brennan, 2015). Brennan ibid notes that "…*for some teachers, a lack of content knowledge can undermine confidence, discouraging them from using Scratch with their student"*. In our ScratchMaths PD sessions and intervention classroom trialling this proved to be one of the key factors for all primary teachers. Thus, to keep the proper balance between the *tool*, *computing* and *learning*, in the project we developed the *5Es teaching framework* consisting of five constructs: Explore, Explain, Envisage, Exchange and bridgE and tend to engage them in each activity.

[36] for example by using a tool which by its programming paradigm or its control and data structures does not correspond to the developmental level of the pupils

This framework – presented in more detail in Benton et al. (2016) – is the leading principle for us in the ScratchMaths design process when matching computational concepts with the way they are supported in Scratch and the strategies to support the pupils to construct their meanings. In the presentation I will illustrate some of these learning strategies, including:

- Encourage pupils to explore by proceeding from *direct manipulation* to *direct drive* to *programming*, from simple to compound, from isolated to composite, from single to multiple.
- Encourage pupils to identify with a sprite (to use their body, wooden toy or paper cut-outs in their hands to explore, envisage and plan their scripts).
- Follow consistent rules throughout different activities (such as *use as few blocks as possible*; *have good reason for each block in your script*; *look for the smallest number of repetitions to achieve the task*; or *give a name to certain behaviour, keep the definition and use the name instead*).
- Use strong and persistent metaphors (such as *moving* for each 'turtle geometry' motion command and *jumping* for each Cartesian style motion command).
- Collect 'good mistakes' and exploit them to envisage and discuss the differences.
- Explore and explain unknown or unexpected behaviour by decomposing it into its parts.
- Encourage exchanging solutions between pupils and modifying them.
- Bridge and discuss links with conventional mathematics, re-contextualise learning outside of computing.

Although there are many factors influencing the sustainability of primary programming, the key one, I believe, is the confidence of the primary teachers in its relevance for their learners. Primary teachers are professionals who must be respected for their remarkable expertise and appropriately supported. In our research and development efforts we are trying to achieve this by exploiting the primary education milieu, studying the conceptual framework of primary programming, exploring the richness of its niches, and striving to negotiate productive balance within the 'triangle of respect' among *computing*, *tool* and *learning*.

I also believe that new content and pedagogies for computing or computational thinking can be developed that are developmentally appropriate – for every primary pupil, mediate the potential of programming to education, avoid the danger of focusing on mastering technology itself, and productively contribute to all domains of the pupils‘ development, including mathematical thinking.

Keywords

programming; computing; computer science; Scratch; primary education; design research

Acknowledgemants

I would like to thank my Comenius University Department of Informatics Education colleagues with whom we have been implementing educational programming content and PD programmes in Slovakia. I would also like to thank my ScratchMaths project colleagues. Our project has been key influence for this paper although I express here my own opinions, which they may not completely identify with.

References

Abelson, H. & Sussman, G.J., 1986. *Structure and Interpretation of Computer Programs*. MIT Press.

Armoni, M., Meerbaum-Salant, O. & Ben-Ari, M., 2015. From Scratch to "Real" Programming. ACM Transactions on Computing Education, 14 (4), Article 25.

Benton, L., Hoyles, C., Kalas, I. & Noss, R., 2016. Building Mathematical Knowledge with Programming: Insights from the ScratchMaths Project. Paper submitter to *Constructionism 2016*.

Blackwell, A.F., 2002. What is Programming? In *14th Workshop of the Psychology of Programming Interest Group*. pp. 204–218.

Blaho, A., Kalas, I. & Tomcsanyi, P., 1993. Comenius Logo: Environment for Teachers and Environment for Learners. In *Proceedings of EuroLogo*, Athens, pp. suppl. 1–11.

Blaho, A, Kalas, I. & Matusova, M., 1994. Environment for environments: A new metaphor for Logo. In Wright, J. & Benzie, D. (eds.): *Exploring a New Partnership: Children, Teachers and Technology*. Elsevier, 1994, pp. 153–166.

Brennan, K. & Resnick, M., 2012. New framework for studying and assessing the development of computational thinking. In *Proceeding of the 2012 Annual Meeting of the American Educational Research Association*, Vancouver, Canada.

Brennan, K., 2015. Beyond Technocentrism: Supporting Constructionism in the Classroom. *Constructivist Foundations*, 10(3), pp.289–296.

DfE, 2013. *Computing Programmes of Study: Key Stages 1 and 2*, DfE. Available at: https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study.

Duncan, C., Bell. T. & Tanimoto, S., 2014. Should your 8-year-old learn coding? In *Proceedings of WiPSCE 2014*. pp. 60–69. ACM Digital Library DOI 10.1145/2670757.2670774.

Gujberova, M. & Kalas, I. 2013. Designing productive gradations of tasks in primary programming education. In *Proceedings of WiPSCE 2013*. pp. 108–117. ACM Digital Library DOI 10.1145/2532748.2532750.

Guzdial, M., 2004. Programming environments for novices. In *Proceesings of the Computer Science Education research*. ACM, pp.127–154.

Meerbaum-Salant, O., Armono, M. & Ben-Ari, M., 2013. Learning computer science concepts with Scratch. *Computer Science Education*, 23,3, pp. 239–264.

Moravcik, M. & Kalas, I., 2012. Developing software for early childhood education. In: *Addressing educational challenges: the role of ICT*. Manchester MMU, pp. 1–12.

Nikolov, R. & Sendova, E., 1985. Informatics textbooks for beginners. In *Proceedings of the International Conference Children in an Information Age: Tomorrow Problems Today*. Varna, Bulgaria, pp. 621–629.

Papert, S. 1980. *Mindstorms: Children, Computers, and Powerful Ideas.* New York: Basic Books, Inc.

Papert, S., 1987. Information technology and education: Copmuter criticism vs. technocentric thinking. *Educational Researcher* 16(1), pp. 22–30.

Sendova, E., 1989. Computers as a Stimulus for Generating Ideas. *Education & Computing*, Vol. 4, Elsevier Science Publishers B.V., pp. 151–155.

Turcsányi-Szabó, M., 1997. Designing Logo Pedagogy for Elementary Education. In *Proceeding of EuroLogo conference*, John von Neumann Computer Society, pp. 273–283.

Watt, S., 1998. Syntonicity and the psychology of programming. In *Proceedings of 10th Annual Meeting of the Psychology of Programming Interest Group.* pp. 75–86.

Wing, J.M., 2008. Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 366(1881), pp.3717–3725.

Papert's sort-of-right mathematics

Jean-Francois Maheux, *jfuqam@gmail.com*
Département de Mathématiques, Université du Québec à Montréal

Abstract

In this theoretical paper, I react to a comment made by Papert in relation with the "sort of right" answer he obtained with a logo program he made to solve a mathematical task. I argue that mathematics and mathematical activity are often sort-of-right, despite the usual impression that mathematics is the culprit of "perfection". I give examples of the presence and importance of sort-of-right mathematics in the discipline, and in school. I conclude with a reflection on the possibility to give more importance to sort-of-right mathematics in regard with teachers and students vision of mathematics, and its resonance with Papert's writing about what education ought to be.

Keywords

mathematics education; vision of mathematics; precision and certainty; Papert

There's a crack in everything
That's how the light comes in
Mike Strange, Shanna Crooks & Leonard Cohen (1992). *Anthem*.

Oxygen

In April 2004, Seymour Papert (Papert & Jettinghoff 2004) recorded a talk at the University of Maine (e.g. www.youtube.com/watch?v=4iIqLc0sjjs). Papert presents a Turtle Geometry environment, but a problem occurs with the Logo program prevents him from showing a correct answer to the task they where discussing. Papert then goes on saying:

> We did something wrong with our unit, but you see, that's great because then we go to correct the unit and it's *sort of right* and this is one of the things that is so wrong with mathematics as it is taught in schools: you can't be sort of right.

Developing the thought, he then talks about debugging as a fundamental mathematical process, and the importance of being wrong to be able to learn from a mistake by fixing it. I got stuck, however, with the idea of *sort-of-right*, thinking that it might not (always?) be necessary to "fix" things: what if sort-of-right was right enough? Is this not an important piece of constructionism-inspired mathematics education?

Fuel

Mathematics is often present as the culprit of exactitude and precision, but there are many branches of mathematics in which approximate answers play a very important role. We can think for example about inverse (or ill-posed) problems, for which different kind of sort-of-right solutions can be produced (e.g. Kabanikhin, 2008). Advocates of experimental mathematics (e.g. Franklin, 1987) also argue in favor of giving more importance to empirically justified results, arguing that "insistence on instant absolute rigor is sterile" (Chaitin, 2004, p. 7). Confirming the value of the first 10^{13} zeros of the zeta function (and a few billion more from the 10^{24}-th zero) (see Gourdon & Demichel, 2004) doesn't make Riemann hypothesis right, but certainly corroborates it enough to justify working under the premise that it is sort-of-right. Should it be proven wrong, I am certain many mathematicians will hastily try and find ways to preserve the rightfulness depending on it!

I am deliberately not glossing upon "applied mathematics" here, where exactitude and precision are *always* contingent, should we build a bridge or develop tools for) signal analysis. We all know how, through history, numbers like π and $\sqrt{2}$ were approximated for practical purpose: if you want to know how many one-step-long stones you need to go around a circle with a 10 step-long diameter, 3x10=30 certainly gives you a good enough answer; and perhaps you'll bring an extra one or two, knowing that your sort-of-right calculation always underestimate what you actually need. What we tend to forget, in my view, is how mathematics itself is constructed in similar a way, and develop many branches in which being sort-of-right is all we whish for.

To give one example, lets consider asymptotic comparison: a method to study a relation (e.g. the behavior of a function) through the behavior of another function deemed "simple" or better-understood. A simple case is that of the average order of an arithmetic function the number of divisors of n, $\tau(n)$. The function $\log(n)$ gives that number "on average": it is sort-of-right. Of course, there maybe a better approximation of $\tau(n)$, and perhaps maybe even an *exact* function, who knows… Point is: it is *also* enough to average it to that simple expression, and go on with it, or move on.

Perhaps starting with Brouwer's controversial rejection of the excluded middle principle (e.g. Van Heijenoort, 1967), notions of rigor, proof and certainty have also been remarkably revisited over the last century, making room for what we may call their sort-of-right equivalent: confidence, explanations, conviction, and so on (Kreisel, 1972; Takeuti, 2013; Tymoczko, 1998; Wittgenstein, 1969). With this, for example, humanly verifiable proofs are then one particular kind of explanations, assuredly "righter" than those escaping the grasp of human mind or logical argumentation. As a matter of fact, even the notion of "humanly verifiable" has been put under pressure, with the arrival of proofs variable by at most 4 or 5 people (e.g. Mochizuki's work on the ABC conjecture).

The "fallibilist" tradition also thought us how mathematics as a human practice is intrinsically defective, always subject to revision, always in need of reaffirmation, always filled with unnoticed inaccuracies (e.g. Davis, 1972; Livingston, 1986; Ormell, 1996). We did live a whole 70 years with William Shanks' 180 wrong decimals for π, and even the strongest advocates against the fallibilists conclusions on the nature of mathematics recognize that doing mathematics is precisely about fixing mistakes, extending concepts, clearing up ambiguities, and that what we might call it's sort-of-right-right aspect are heuristic to mathematical freedom to create (e.g. Sherry, 1997). But I think that fully assuming the value of sort-of-right mathematics might require a little bit more. Fallibilism often seems emphasize "doing mathematics" and avoid addressing questions related to mathematics "as a network of ideas". Ernest (2014) thus write about that "maverick tradition": "the certainty of mathematical knowledge is still acknowledged, the concept of certainty is circumscribed and limited to current human knowing" (p.3).

Reading through history of mathematics with a dialectical mindset, I have recently begin conceptualizing what I call the "im|perfect" nature of (doing) mathematics (I use the Sheffer stroke to emphasize the dialectic tension). The idea is to examine the fruitful, mutually constitutive interplay between opposing forces in the development of mathematics as a network of ideas, and in actual practice. At this moment, I have articulated five dimensions, where im|perfection contribute to (doing) maths: ambiguity|clarity, traces|ideas, impressions|convictions, dispersion|discipline, limitation|expansion. The dimension called limitation|expansion, for example, refers to how what counts as mathematics is often jostled by new extensions. There is a drive towards limiting (doing) mathematics to a minimal number of concepts and ideas, but also a constant venture to create new possibilities (the successive controversial apparitions of irrational, negative, complex, and transfinite numbers illustrates this very well). Could such a line of thinking offer a appealing enough basis to both recognize try and embrace sort-of-right mathematics?

Heat

That's a lot about math… What about mathematics education? Well… First off, Papert's sort-of-right reminds of the concept of "good-enough" discussed by Reid and Zack (2003, 2004). Examining students' accomplishments, Reid and Zack came to note:

> Our investigation revealed that much of what we had marveled at was in fact incomplete, tentative, and sometimes inconsistent. Yet at the same time the students were able to continue to work on the problem, to explore, and even to explain things to each other that they seemed not to fully understand themselves. (p. 28)

Students understandings were not perfect, but good-enough to engage, and keep moving forward in relation to the task at hand. Thinking about it, is this not actually happening *all the time*? Was it ever a condition that *all* students would perfectly understand *everything* before moving on to the next topic? Students (and teachers) seems to "make do" with sort-of-right understandings, and this is also how (at least in this case) they can come up with the most interesting insights and ideas. An interpretation reminiscent of Rowland (2000) study on the role of vagueness in students' mathematical talk. The good-enough metaphor has a lot in common with constructivist replacement of truth with viability (e.g. Von Glasersfeld, 1995), recently taken up by Proulx (2013) to analyse students' strategies or mathematical solutions as functional (instead a optimal) within a context. In a way, this is truly the case of Papert's little program in the video: it works well as far as showing how a program can be made to find an answer, even if it is not optimal in doing so (by not actually providing the correct figure). And again looking at things this way, I believe it is not uncommon *at all* for teachers to assess students' sort-of-right answers, making points on partial solution or passing over little mistakes. I am not sure, however, of how much we actually value that kind of sort-of-right mathematics: is it mostly something we wish we could get rid of, or maybe just some necessary evil? Or do we truly believe its irreplaceable significance?

The mathematical ideas encountered by students in school also often have sort-of-right features. First, mathematicians themselves hardly define what "numbers" are (and hopefully nevertheless make progress (Wiener, 1915)). Many concepts are also (officially or practically) defined in mathematically surprising manner, for example what is exactly "a fraction"? Mathematical definitions are always contingent, but that contingency is rarely made explicit. For example, the notion of parallel lines, valid in Euclidian geometry, has to be completely redefined in non-Euclidian geometry. Borasi (1993) illustrates this with a study in which she confronted students with the definition of a circle applied to taxi-geometry. Others did similar things with the concepts (e.g. "slope") and their representations (Zaslavsky, Sela, & Leron, 2002). After all are not all symbols "signs of ambiguity" (Mamolo, 2010)? That is: there are many occasions in which sort-of-right mathematics is offer to the students, but again probably not in a way that emphasizes the productive, creative, or simply interesting aspect of this sort-of-rightness.

Chain reaction?

The idea of sort-of-right mathematics surely has a disruptive power. In the beginning of the 20th century, the mathematical community started to struggle in what we now call the foundational crisis. But as a result, new branches of mathematics where created, new heuristics where brought forth, and richer conceptualizations developed. What if we tried exploring the potential of sort-of-right mathematics for teaching and learning? What if we deliberately set it out, and ask students for approximate answers, good-enough solutions, viable procedures, and so on? What if we tried identifying new mathematical topics to explore with students precisely to bring forth sort-of-right mathematics, like fuzzy logic or Fermi problems?

Back in the 1980-1990, numerous studies showed how students' and teachers' discourse about mathematics at all levels (including the university) where evocative of what Lakoff and Nunez (2000) call the "romance" of mathematics, and found this situation detrimental to teachers and students (e.g. Raffaella Borasi, 1990; Ernest, 1992; Herzig, 2002; Thompson, 1984).I suspect that the kinds of mathematical experiences offered to them have a lot to do with this. In school, mathematical ideas are almost always well defined, well ordered. Problems almost always have solutions (if not "a" solution), conjectures are (shortly) proven, tools are effective, and consistency is never a problem. Sort-of-right mathematics would be in strong conflict with that picture.

Could this warm up a bit what is sometime described as the cold-dead mathematics of school, where teachers and students are restricted to engage with ideas that have been "steeled" for decades or century (what Janvier and Glaymann termed as "des mathématiques toutes faites")? Wouldn't it make sense in a world increasingly populated by technologies increasingly capable of presenting on the spot mathematical information? When key capabilities are that of asking good questions and interpreting answers, it is more than enough to able to feel what it sort-of-right: it is fundamental!

I think also it is an important statement to make, in response to how mathematics has been used to justify normative regimes of schooling primarily aimed at producing clarity, rationality, objectiveness, exactitude rather than pleasure, beauty, sensitivity, compassion and so on. It goes in favour of what Brown (1993) called the necessity of a *pedagogy of the confuse*, where errors, doubts or vagueness are not only brief and temporary obstacles to be overcome, but the very condition of doing mathematics. I see it also a way to *positively* respond to the "challenge" of technology. Let's consider how Bret Victor's project to "kill math" (http://worrydream.com/KillMath) in regard to its requirement of complex, abstract symbolic manipulations. Bret suggests using technology to simplify all these operations, and intuitively find (right or sort-of-right!) answers. In that light, even modeling (for which the sort-of-right mathematics gives an additional boost as to its relevance to what we do in schools) might take quite a new look (fig. 1).

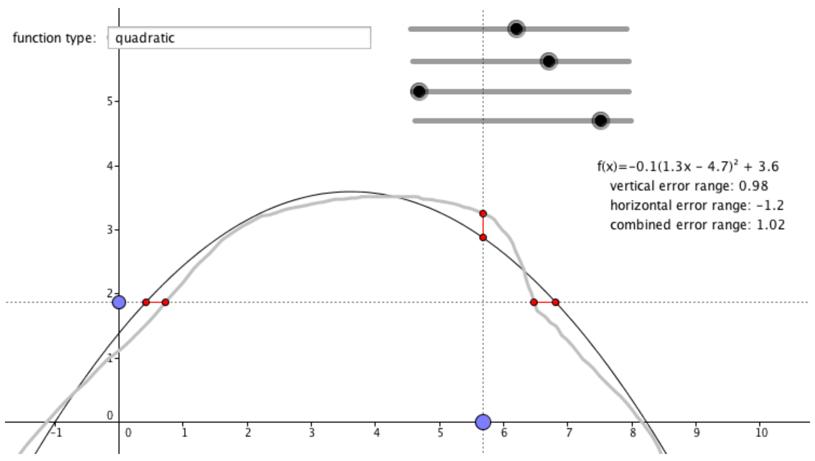


Figure 1: modeling a curve with a quadratic function

But as mentioned earlier, I want to insist in this paper on the sort-of-right possibilities for mathematics even outside of its practical applications where "good enough" solutions have long been recognized as the norm. The public, starting with our students, will certainly benefit from growing in awareness of this situation (see for example Furinghetti, 1993), and I feel the case is easily made! This is also, as one of the reviewer noted, where constructionism will most easily intersect with the ideas presented in this paper. The reader should thus be able to make insightful contributions from that side. On the other hand, bringing into conversation the sort-of-right aspect of more “theoretical” aspect mathematics might not come so easily. And I wonder if recognizing the interplay of perfection and imperfection, im|perfection, in mathematics “as a network of ideas” might incite us to explore different avenues for mathematical activity within constructionism. How about, for example, engaging students in creating programs to test a given mathematical conjecture (and discuss the outcomes), or play with the idea that a program (e.g. for a figure) could be a definition and discuss what follows from one or another (and why), or explore what happen if after creating something using that program, we decide to change the definition! I personally plan on trying some of this things in a near future, starting with Collatz’s conjecture, for example, as suggested by Stagerez (1997).

I cannot conclude, however, without going back to Papert’s comment and react to the idea that there is something “so wrong” with mathematics as it is taught in schools. If it is OK for mathematics to be sort-of-right, maybe it should also be the case of “mathematics as it is taught in schools”. What I mean is that what we generally try to do in school with mathematics is incredibly difficult, if not simply impossible (as Brousseau (1988) contend). What I understand as being my job as a researcher in mathematics education is not to go about finding and fixing “what is wrong” in schools. It is to provide ideas as to what could possibly be done in, out, around, for, with, or even against mathematics educational system. Nothing more… but nothing less. Which also means I very much agree with an another of Papert’s remarkable assertion in this video:

> I like to say there is a big distinction between something that I love and I call mathematics and something called “math”, which is what we teach in schools … and that’s not a mathematics curriculum it’s a “math” curriculum. […] Mathematics is an active intellectual activity, and it means working at things where you’re using the mathematical ideas that you are struggling with for a larger purpose. And, the idea that the larger purpose could be discovering something that the teacher decided you got to discover is not a larger purpose.

Papert always insisted on the playfulness of doing mathematics as a paradigmatic example of what education in general ought to be. It is not always, not for everybody that mathematical rightfulness becomes an obstacle to the joys of mathematics. On the contrary, it is for many one of the core reasons why they love doing mathematics. Similarly, in other domains, the desire to know often comes with an aspiration toward “truth”. It is a noble part of human endeavor! But there are also moments, and people, for which rightfulness comes in the way, becomes a burden, and dispossess life from its liveliness. It is time to see how (mathematics) education can also contribute this other noble part of being human, one in which it is not necessary to be right, to be good.

References

Borasi, R. (1990). The invisible hand operating in mathematics instruction: Students conceptions and expectations. In T. J. Cooney (Ed.), *Teaching and Learning Mathematics in the 1990s* (pp. 174-182). Reston, VA: NCTM.

Borasi, R. (1993). Appreciating the mathematical element of mathematical content: the case of definitions. In S. I. Brown (Ed.), *Essays in Humanistic Mathematics* (pp. 123-139): Mathematical Association of America.

Brousseau, G. (1988). Fragilité de la connaissance et fragilité du savoir. Centre interdisciplinaire de recherche sur l'apprentissage et le développement en education: Université du Québec à Montréal. .

Brown, S. I. (1993). Toward a pedagogy of confusion. In A. M. White (Ed.), *Essays in Humanistic Mathematics Education* (pp. 107-121): Mathematical Association of America.

Chaitin, G. (2004). Thoughts on the Riemann hypothesis. *The Mathematical Intelligencer, 26*(1), 4-7.

Davis, P. J. (1972). Fidelity in Mathematical Discourse: Is One and One Really Two? *The American Mathematical Monthly, 79*(3), 252-263.

Ernest, P. (1992). The nature of mathematics: Towards a social constructivist account. *Science & Education, 1*(1), 89-100.

Ernest, P. (2014). Certainty in mathematics: is there a problem? *Philosophy of Mathematics Education Journal*(28).

Franklin, J. (1987). Non-deductive logic in mathematics. *British Journal for the Philosophy of Science*, 1-18.

Furinghetti, F. (1993). Images of mathematics outside the community of mathematicians: Evidence and explanations. *For the Learning of Mathematics*, 33-38.

Gourdon, X., & Demichel, P. (2004). The first 1013 zeros of the Riemann Zeta function, and zeros computation at very large height.

Herzig, A. H. (2002). Where have all the students gone? Participation of doctoral students in authentic mathematical activity as a necessary condition for persistence toward the PH. D. *Educational Studies in Mathematics, 50*(2), 177-212.

Kabanikhin, S. (2008). Definitions and examples of inverse and ill-posed problems. *Journal of Inverse and Ill-Posed Problems, 16*(4), 317-357.

Kreisel, G. (1972). Informal rigour and completeness proofs. In I. Lakatos (Ed.), *Problems in the philosophy of mathematics* (pp. 138-187): North Holland publishing.

Lakoff, G., & Núñez, R. E. (2000). *Where mathematics comes from: How the embodied mind brings mathematics into being*: Basic books.

Livingston, E. (1986). *The ethnomethodological foundations of mathematics*: Routledge.

Mamolo, A. (2010). Polysemy of symbols: Signs of ambiguity. *The Mathematics Enthusiast, 7*(2), 247-262.

Ormell, C. P. (Ed.) (1996). *New thinking about the nature of mathematics*: MAG - University of East Anglia.

Papert, S., & Jettinghoff , R. (2004). Talk by Seymour Papert & Robin Jettinghoff [YouTube video]. Retrieved on June 2015 from https://www.youtube.com/watch?v=4iIqLc0sjjs

Proulx, J. (2013). Mental mathematics, emergence of strategies, and the enactivist theory of cognition. *Educational Studies in Mathematics, 84*(3), 309-328.

Rowland, T. (2000). *The pragmatics of mathematics education: Vagueness in mathematical discourse* (Vol. 14): Psychology Press.

Sherry, D. (1997). On mathematical error. *Studies in History and Philosophy of Science, 28*(3), 393-416.

Stagerz, G. S. (1997). Logo and Learning Mathematics-No Room for Squares. *Computers in the Schools, 14*(1-2), 153-169.

Takeuti, G. (2013). *Proof theory* (Vol. 81): Courier Corporation.

Thompson, A. G. (1984). The relationship of teachers' conceptions of mathematics and mathematics teaching to instructional practice. *Educational Studies in Mathematics, 15*(2), 105-127.

Tymoczko (Ed.) (1998). *New Directions in the Philosophy of Mathematics*. Princeton University Press.

Van Heijenoort, J. (1967). *From Frege to Gödel: A Source Book in Mathematical Logic, 1879-1931*: Harvard University Press.

Von Glasersfeld, E. (1995). *Radical Constructivism: A Way of Knowing and Learning. Studies in Mathematics Education Series: 6*: ERIC.

Wiener, N. (1915). Is mathematical certainty absolute? *The Journal of Philosophy, Psychology and Scientific Methods, 12*(21), 568-574. Retrieved from http://www.jstor.org/stable/2012740

Wittgenstein, L. (1969). *On Certainty* (D. Paul & G. E. M. Anscombe, Trans.). Oxford: Blackwell.

Zack, V., & Reid, D. A. (2003). Good-enough understanding: Theorising about the learning of complex ideas (part 1). *For the learning of mathematics*, 43-50.

Zack, V., & Reid, D. A. (2004). Good-enough understanding: Theorising about the learning of complex ideas (part 2). *For the learning of mathematics*, 25-28.

Zaslavsky, O., Sela, H., & Leron, U. (2002). Being sloppy about slope: the effect of changing the scale. *Educational Studies in Mathematics, 49*, 119–140.

Preparing teachers for the Digital Technologies Curriculum: Preliminary results of a pilot study.

Elena Prieto-Rodriguez, *Elena.Prieto@newcastle.edu.au*
School of Education, The University of Newcastle

Daniel Hickmott, *Daniel.Hickmott@newcastle.edu.au*
School of Education, The University of Newcastle

Abstract

The Digital Technologies (DT) curriculum in Australia, particularly at primary school level, has been considerably expanded in the recent past from the teaching of digital literacy to the teaching of the skills necessary to create and innovate with digital technologies. The concepts of computational thinking, systems thinking and programming are part of this new curriculum. This paper presents the theoretical perspective, methodology and preliminary results of a research project, currently being run to explore different pedagogies for teacher preparation.

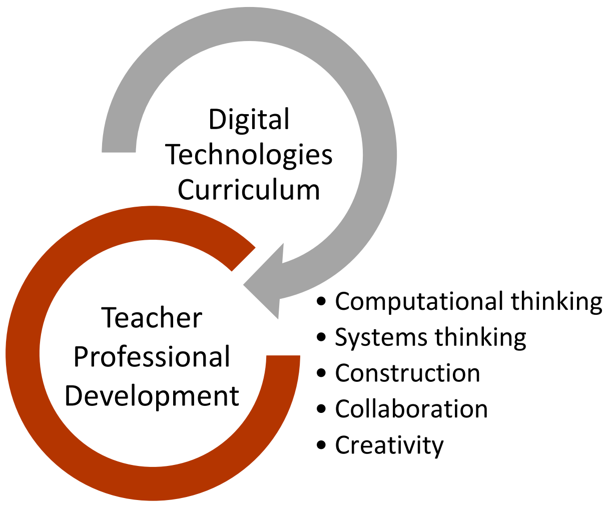


Figure 6: Teacher professional development for the new Australian Digital Technologies Curriculum

The objective of the project is to pilot a research informed strategy for teacher preparation for the DT curriculum and articulate terms of best practice. To do so we are using as a test case a group of teachers participating in a Computer Science 4 High School (CS4HS) initiative at our institution. We hope that by sharing our experience and methodology, we can receive feedback on how to improve on the second phase of this project, and find ways to increase collaboration with other teacher educators.

Keywords

Computational thinking, Digital Technologies curriculum, teacher training, systems thinking, programming education

Introduction

Worldwide, the focus of K-12 Information and Communication Technologies curricula is shifting from the teaching of digital literacy (the use and understanding of digital tools), to the teaching of the skills necessary to create and innovate with digital technologies (Bower & Falkner, 2015). Computational thinking, systems thinking and programming are now skills that are deemed as necessary to allow this creativity and innovation to occur (ACARA, 2014).

A recent report from European Schoolnet (2014) surveyed 20 European countries to discover which had implemented, or were planning to implement, curricula that teach these skills. The report found that 13 of the 20 European countries were in the process of implementing these curricula in their schools. Seven of these 13 countries have made the curriculum compulsory from Foundation to Year 9. Outside of Europe, elective subjects teaching these skills have started appearing in high school curricula in countries such as Israel (Hazzan, Gal-Ezer, & Blum, 2008) and New Zealand (Bell, Newton, Andreae, & Robins, 2012).

The Australian curriculum that aims to teach these skills, the *Digital Technologies* (DT) curriculum (ACARA 2012) has only very recently received final endorsement and will be compulsory for students from Foundation to Year 8. Research into the implementation of the DT curriculum reveals that there is widespread concern about whether teachers will be adequately prepared to teach this curriculum once its implementation commences (Falkner, Vivian, & Falkner, 2014). This recent study revealed that "a consultation with Industry, Community and Education stakeholders in Australia showed that 55% of respondents had concerns about manageability of the implementation of the curriculum, while 45% of respondents did not think that its learning objectives were realistic" (Falkner et al., 2014).

The most common approach to prepare teachers for the DT Curriculum in Australia occurs in the form of short courses, Massive Online Open Courses (MOOCs) such as the one provided by the University of Adelaide[37], or workshops, such as CS4HS (Google, 2015). Globally, CS4HS is organised and run mainly by computer science academics, and there is limited research published about the content of the workshops or what pedagogies are used to convey this content (Blum & Cortina, 2007; Liu, Hasson, Barnett, & Zhang, 2011; Prieto-Rodriguez & Berretta, 2013).

The research we present in this paper originates from three years of CS4HS workshops at an Australian institution, and the realisation that pedagogy, as well as content is essential for professional learning. This paper presents the theoretical perspective, methodology and preliminary results of a study into pedagogies for developing in teachers the necessary skills to impart the DT curriculum.

Theoretical perspective

Computational thinking (CT) is a problem solving approach that has been defined as "the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent" (Wing, 2006). This approach involves characteristics such as problem formulation so that solutions can be run on computers, data analysis, and algorithm design (Stephenson & Barr, 2011). Similarly, systems thinking (ST) is an approach to analysing and understanding digital and physical systems, and their interdependencies and interactions (Davidz & Nightingale, 2008).

The methodology and preliminary results we report in this paper are the product of an endeavour to systematically review workshops with a strong focus on CT and ST imparted at our institution. These workshops have been run annually since 2013 and comprise presentations by university academics, tutorial sessions, and tours of research laboratories. The tutorial sessions have

[37] https://csdigitaltech.appspot.com/course

involved the teaching of a particular programming language (for example, Python), or a particular concept (for example, Data Mining).

In 2013 and 2014 the content of the sessions was delivered using a Direct Instruction (DI) approach. DI refers to the explicit teaching of a skill-set using lectures or demonstrations of material or concepts. It is often contrasted with inquiry-based learning or discovery learning and usually involves an explication of the skill or concept. As such, it may or may not include an opportunity for student participation or individual practice and thus is considered a “teacher-centred” approach (Killen, 2013).

In 2015 the team decided to move away from only using this teaching style, and explore the use of student-centred pedagogies based on constructivist and collaborative learning theories. We were particularly concerned about the varying levels of proficiency with technology that participants to our workshops typically have and decided to choose a framework that would take into consideration all these factors.

The Five Cs framework (Tom, 2015) is a student-centred teaching and learning framework that aims to support students with diverse disciplinary background and learner characteristics. It has been trialled in a Masters of IT program at Central Queensland University, which involves the teaching of complex computer programming concepts, with positive results. As the CS4HS workshops’ participants often have diverse programming backgrounds, and the content can involve complex ideas it was seen as a suitable pedagogy to use during the tutorial sessions in the CS4HS workshops. While DI and other student-centred pedagogies have been used in the past for Computer Science teacher training programs like CS4HS (Liu et al., 2011), the Five Cs framework has not.

The Five Cs framework is centred on 5 constructs identified from the study of constructivist and collaborative learning theories (Derry, 1996; Dijkstra, 1997). These 5 constructs are: Consistency, Collaboration, Cognition, Conception and Creativity. The definitions of the different terms, and the way in which they fit together to support student learning are shown in Figure 7.

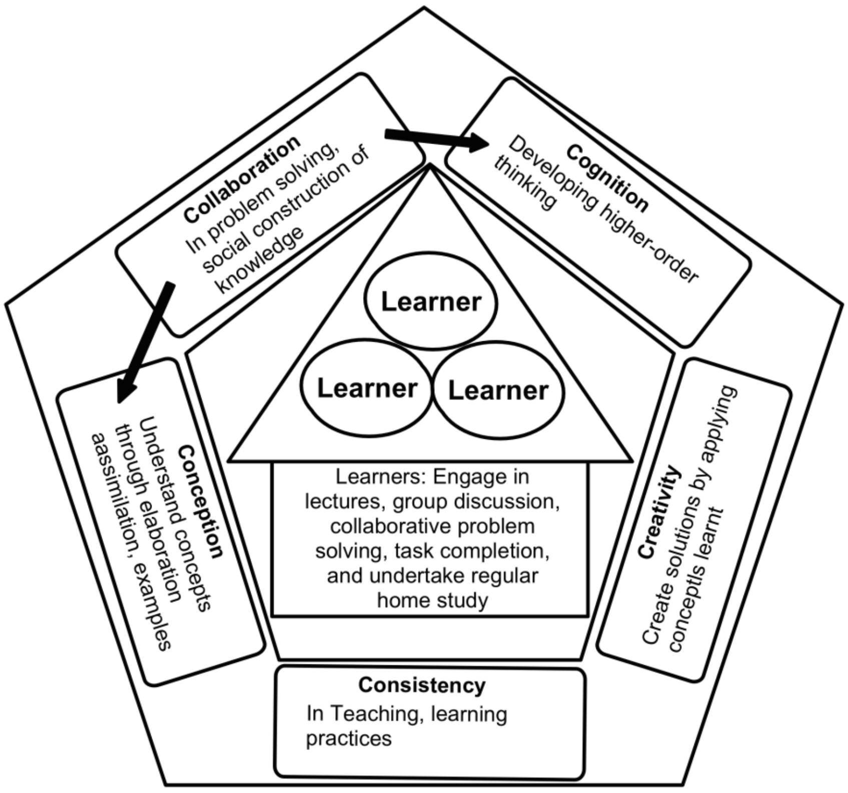

Collaboration
In problem solving, social construction of knowledge
Cognition
Developing higher-order thinking
Learner
Learner
Learner
Learners: Engage in lectures, group discussion, collaborative problem solving, task completion, and undertake regular home study
Conception
Understand concepts through elaboration assimilation, examples
Creativity
Create solutions by applying concepts learnt
Consistency
In Teaching, learning practices

Figure 7: Theoretical framework - Five Cs (Tom, 2015)

These 5 constructs have been used to inform the design the sessions in the CS4HS workshops reported in this paper following the Five Cs framework tutorial structure outlined in Tom's study (2015). The Five Cs framework tutorial structure consists of the following parts for each concept that is presented:

- **Explanation / Elaboration –** the concept is explained emphasising real world examples of where the concept is used.
- **Conceive and Communicate –** the students form groups and discuss questions related to the concept that was explained.
- **Interaction –** the students put forward any questions that arose in the group discussions, and share any insights they have discovered.
- **Collaborative Problem Solving –** the students work together in the formed groups to solve a problem. This problem could be one that has been provided by the lecturer, or one that they have created themselves that applies the concept presented.

Research Aims and Methodology

The aim of this research project is to explore the use of student-centred pedagogies at a professional learning workshop for teachers. The specific research questions of the project are:

1. How do teachers with diverse technical backgrounds experience student-centred pedagogies in the context of professional learning for the DT Curriculum?
2. Do these experiences match the ones outlined by Tom (2015) in a more general context?

To answer these research questions we are conducting a research project over a 10-month period in 2015 and 2016. We will gather quantitative and qualitative data during and after the two professional learning CS4HS workshops we are running in 2015.

The first workshop was held in July 2015 and its participants were high school teachers, in-service (6) and pre-service (3), who had advanced knowledge of computer science and computer programming. The second workshop was held in November 2015 and participants were primary (21) and high school teachers (2) who had very limited knowledge of computer science or computer programming.

As outlined in the Introduction, the tutorial sessions in the two workshops are purposely designed to investigate participant responses to different pedagogies. Each of the two workshops involves three sessions and differed slightly between the Advanced and the Introductory workshops. The pedagogies chosen for each of those are described below.

For the Advanced workshops:

- Session 1 – Web Technologies: Direct Instruction as described by Killen (2013)
- Session 2 – Data Mining: The Five Cs framework (Tom, 2015)
- Session 3 – App Design: Direct Instruction (Killen, 2013)

For the Introductory workshops:

- Session 1 – Visual programming with Scratch: Direct Instruction (Killen, 2013)
- Session 2 – Algorithms with Mindstorms Robots: The Five Cs framework (Tom, 2015)
- Session 3 – App Design: Direct Instruction with some collaboration (Killen, 2013)

The sessions utilising the DI pedagogy began with a short (10 – 15 minute) presentation giving an overview of the key concepts for the covered topic. After the presentation participants worked through a step-by-step tutorial, and asked for assistance from Author 2 if it was required.

The sessions utilising the Five Cs framework also started with a short presentation that gave an overview of the key concepts (Explanation /Elaboration). At the end of the presentation the participants were given a few minutes to discuss ideas for a task they could complete that involved data collection and analysis (Conceive and Communicate). After participants were presented with some examples, they proceeded to create their own tasks to complete. After deciding on a task each of the groups explained their ideas to Author 2 and asked any questions they required clarification on (Interaction). Each of the groups collected data and performed analysis for the remainder of the session (Collaborative Problem Solving), and shared their results and experience with the rest of the groups at the end of the session.

The design of the content in each of the sessions was heavily guided by three overarching aims of the research team when organising the workshops. The first aim is to use the workshops as a way of communicating the applicability and importance of CT and ST to a wide range of domains. The second aim is to provide examples of activities that relate to concepts outlined in the DT curriculum. The third aim is to provide tutorials and materials that the participants can integrate into their practice with limited modifications.

It is important to point out that the content and examples were planned to be relevant to the participants' CT experience, and cover content areas from the stages of the DT curriculum appropriate to individual participants (Foundation to Year 6 or Year 7 to 12, respectively).

To answer the first research question, all participants of both workshops were be invited to complete a survey at the conclusion of the workshops. Three months after their workshop participants are invited to complete a follow-up survey of a very similar nature.

To answer the second research question, participants who completed the follow-up survey will be invited to also partake in one of two focus group sessions to be held in March 2016.

Please note that at the time of writing this paper only the first round of surveys has been conducted, and only the follow-up survey for the Advanced workshop has occurred.

Instruments

Surveys

Surveys were designed to investigate how the participants experienced the different teaching methods used in the tutorial sessions in terms of the Five Cs framework constructs. The surveys aimed to determine how participants experienced these constructs in each of the sessions, not only the tutorial that has been taught using the Five Cs framework tutorial structure.

The *survey* given to participants during the workshops includes questions on the participants' demographics, feedback on the workshop, and three scales with Likert-type items asking about their experiences in each tutorial. A total of 5 questions were asked to participants about each of the sessions, and the scale was designed to ascertain how the participants had experienced the Five Cs framework constructs during the sessions. Each Likert-type item has six response levels ranging from "Very Strongly Disagree" to "Very Strongly Agree". The questions were:

1. The teaching methods used in the session were appropriate for the concepts learned.
2. This session allowed me to improve my higher order thinking skills.
3. I understood the concepts presented in the session.
4. The collaborative nature of the session helped me understand the concepts presented.
5. I was able to find creative solutions to the problems presented in the session by applying the concepts I learned.

The *follow-up survey* is delivered 3 months after each of the workshops. This surveys includes the Likert-type questions from the workshop survey. The questions are asked again to discover if there is any change in the way that participants remember experiencing the 5C framework constructs in each of the 3 tutorial sessions. The follow-up survey also contains other reflective and open-ended questions about the workshop, about how the participants experienced the Five Cs constructs, and about how they approach the teaching of computer science and programming in their classrooms. Also, additional Likert-type questions will be used to discover if the participants would prefer activities that allow for more creativity, more collaboration, and whether they would prefer more concepts covered in less depth, or less concepts covered in more depth.

The surveys are anonymous, but participants are asked to create a personal code so that data can be linked from the first to the follow-up survey. This was done in order to study possible changes in perception of their learning over time.

Focus Group Schedule

The focus group discussions, to be held in March 2016, are designed to collect a variety of information from the workshop participants. The discussions will centre on how participants conceptualise the teaching of computer science and programming in their own practice. The discussion will also allow for participants to reflect on the approaches that they use for learning how to teach computer science and programming, as well as how they find classroom resources for these topics. Emphasis will be placed on the organisational and technical issues that participants may have encountered when teaching computer science and programming, and whether they have participated in professional learning in these topics, not including CS4HS. The proposed schedule for the focus group discussions is shown below and we are wholeheartedly seeking feedback about these questions from participants of **Constructionism 2016**.

1. What pedagogies do you use when teaching programming and computational thinking?
2. Which pedagogies do you think your students prefer the most, and why do you think that?
3. Where do you look for resources for teaching programming and computational thinking?
4. Do you find that the resources need to be modified at all to suit your needs?
5. Is there any support at your school for learning programming and computational thinking?
6. Are there any organisational or technical issues at your school that prevent you from using certain resources or tools for teaching programming and computational thinking?

7. Are there any after-school or extracurricular activities that involve computational thinking and programming available at your school?
8. Have you undertaken any other professional learning (excluding CS4S and CS4HS) for programming and computational thinking?
9. Have you discussed the new Digital Technologies curriculum with other school staff, do you find that their opinions about it are similar to yours?

Preliminary Results

At the time of writing this paper both workshops have already taken place but only the first survey has been completed by participants. An email containing a link to the follow-up survey has been sent to participants in the second survey but only 2 teachers have responded to date, so we are not including its analysis in this paper.

All nine participants of the first workshop (Advanced) completed the survey. Two of those are female and the rest male. One of the females is a preservice teacher. Most participants (5) who completed the survey have been teaching for more than 5 years, and only one of those had been teaching more than 15.

All sessions in this workshop were well liked and all scored quite highly according to the participants in terms of all the constructs in the Five Cs framework (all means except one were above 5 out of 6). The lowest scoring item, however, was the one about collaboration in the session designed using the Five Cs framework (Data Mining). The highest scoring session was overall the one designed following the principles of DI (App design). Interestingly, participants felt that one of the sessions that utilised only DI allowed them the most creativity. No statistically significant differences were found in reference to any of the questions when taking into consideration the gender or professional experience of participants (as measured by the number of years teaching).

Of the 27 of participants in the second workshop (Introductory), only 23 completed the survey. The ratio of female to male was 2:1 which is not atypical in the primary school setting.

It can be seen in Figure 3 that while concepts were understood fairly consistently, for all other constructs, the Scratch session was the one were the teaching methods were preferred by participants and higher order thinking was more present. This session was the only one not involving collaboration and it was not designed using the Five Cs framework. On the other end of the spectrum, the App Design session presented the most problems for participants to the Introductory workshop.

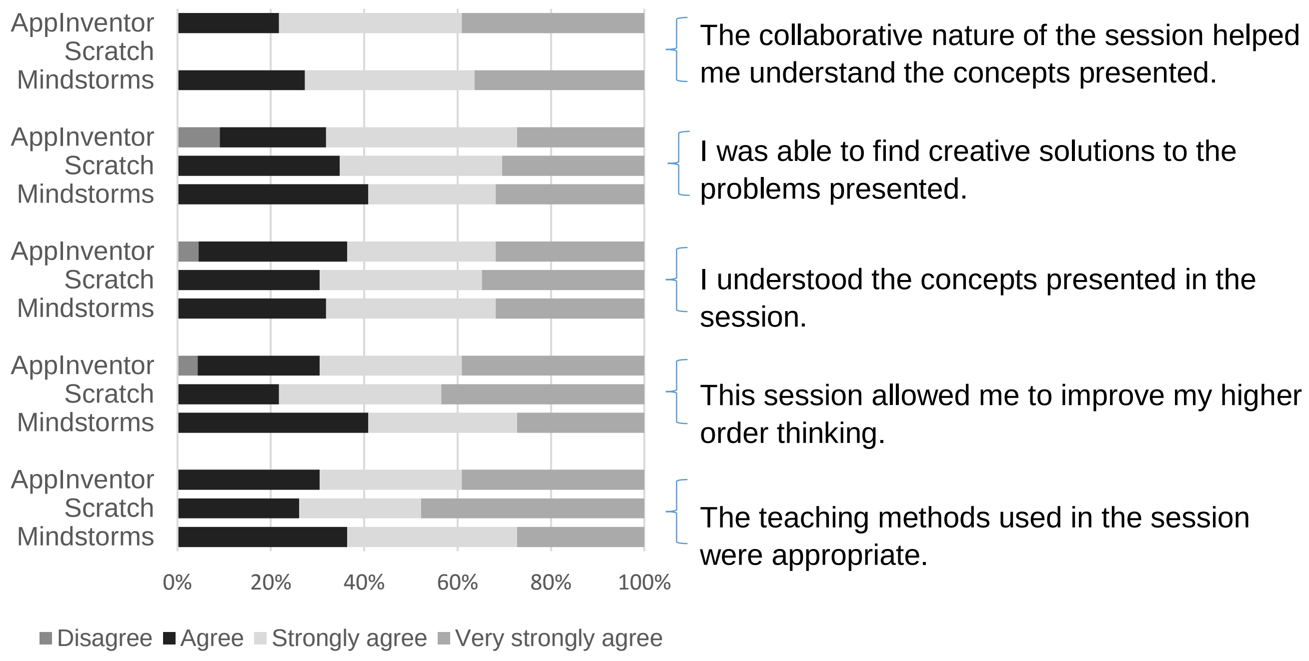


Figure 8: Comparison of sessions in Introductory workshop according to Five Cs framework

Conclusions

In this paper we have presented the methodology and preliminary results of study designed to evaluate how different pedagogies can be utilised in the context of teachers' professional learning. In the design of this project we have focused on the different needs that teachers may have at different stages of their careers while taking into consideration their varying technical background knowledge. To date, we are evaluating a student-centred pedagogy, the Five Cs framework, in contraposition to traditional teaching methods in order to create a research informed strategy for teacher preparation for the DT curriculum and articulate terms of best practice.

At the time of writing this paper, only a small part of the research has been conducted, and the follow-up surveys have not yet been delivered. However, this preliminary work indicates that some elements of the Five Cs framework are perhaps not suitable for teacher training.

In early 2016, the second phase of the project will commence and involve focus group discussions following the interview schedule provided in the Methodology section. We would like to use the feedback we receive at this conference to guide our research endeavours particularly in the second phase of the project, and contribute to the design of this phase and the improvement of teacher training in preparation for the new DT curriculum.

Acknowledgements

The authors would like to thank Google Inc. for funding the 2015 CS4HS workshops held at our institution. We would also like to thank the university academics who presented talks and gave laboratory tours during the workshops, as well as the administrative staff of our institution for their assistance with the organisation and running of the workshops.

References

ACARA. (2014). Digital Technologies. Retrieved 20th May 2015, from http://www.australiancurriculum.edu.au/technologies/digital-technologies/

Bell, T., Newton, H., Andreae, P., & Robins, A. (2012). The introduction of Computer Science to NZ High Schools — an analysis of student work. 5-15.

Blum, L., & Cortina, T. J. (2007). CS4HS: An Outreach Program for High School CS Teachers. *Sigcse 2007: Proceedings of the Thirty-Eighth Sigcse Technical Symposium on Computer Science Education*, 19-23.

Bower, M., & Falkner, K. (2015). *Computational Thinking , the Notional Machine , Pre-service Teachers , and Research Opportunities*. Paper presented at the Australasian Computing Education Conference Sydney, Australia.

Davidz, H. L., & Nightingale, D. J. (2008). Enabling systems thinking to accelerate the development of senior systems engineers. *Systems Engineering, 11*(1), 1-14. doi: 10.1002/sys.20081

Derry, S. J. (1996). Cognitive schema theory in the constructivist debate. *Educational psychologist, 31*(3-4), 163-174.

Dijkstra, S. (1997). The integration of instructional systems design models and constructivistic design principles. *Instructional science, 25*(1), 1-13. doi: 10.1023/A:1002902427253

European Schoolnet. (2014). Computing our future: Computer programming and coding - Priorities, school curricula and initiatives across Europe.

Falkner, K., Vivian, R., & Falkner, N. (2014). *The Australian Digital Technologies Curriculum: Challenge and Opportunity*. Paper presented at the Australasian Computing Education, Auckland, New Zealand.

Google. (2015). Computer Science 4 High School. Retrieved 20th May 2015, from http://www.cs4hs.com/

Hazzan, O., Gal-Ezer, J., & Blum, L. (2008). A model for high school computer science education: the four key elements that make it! *SIGCSE Bull., 40*(1), 281-285. doi: 10.1145/1352322.1352233

Killen, R. (2013). *Effective teaching strategies : lessons from research and practice / Roy Killen*. South Melbourne, Vic: Cengage Learning Australia.

Liu, J., Hasson, E. P., Barnett, Z. D., & Zhang, P. (2011). A survey on computer science K-12 outreach: Teacher training programs. *Proceedings - Frontiers in Education Conference, FIE*, 11-16. doi: 10.1109/FIE.2011.6143111

Prieto-Rodriguez, E., & Berretta, R. (2013). Computer Science : It is not all about programming. *Frontiers in Education*.

Stephenson, C., & Barr, V. (2011). Defining Computational Thinking for K-12. *CSTA Voice, 7,* 3 - 4.

Tom, M. (2015). Five Cs Framework: A Student-centered Approach for teaching programming courses to students with diverse disciplinary background. *Journal of Learning Design, 8*(1), 21-37.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*, 33. doi: 10.1145/1118178.1118215

Programming videogames with models of physical parameters: some examples

Angel Pretelín-Ricárdez, *apretelin@ipn.mx*
Instituto Politécnico Nacional, UPIITA, México
Centro de Investigación y de Estudios Avanzados (Cinvestav-IPN), México

Ana Isabel Sacristán, *asacrist@cinvestav.mx*
Centro de Investigación y de Estudios Avanzados (Cinvestav-IPN), México

Abstract

In the past three years we have been involved in a project (Pretelín-Ricárdez & Sacristán, 2015) with last-year engineering students at the National Polytechnic Institute in Mexico, where they are presented with a sequence (stages) of individual and collaborative activities for constructing (designing and programming) videogames that have mathematical models (e.g. fluids, simple machines, robots, digital circuits, physical parameters, etc.) integrated into game mechanics. With this, we want students to apply their mathematical knowledge in their model and videogame constructions, and in this way learn to situate or contextualize it, as well as relate it to other disciplines, as is required in engineering in real life.

In this paper, we provide four examples of students' designs during the first stage of the activity sequence, which corresponds to the learning or familiarisation with a game engine (in this case, Game Maker Studio), its physics engine and its programming language. In this stage, we ask students to design and implement experiments to characterize a set of basic physical parameters used in the game engine: density, restitution, friction, linear damping, angular damping, velocity, force, impulse and acceleration of gravity. The purpose of this, is that during the design and implementation of experiments, students learn to relate the meaning of a command or piece of code, and other computational objects (sprites, backgrounds, sounds), with a concept (mathematical or physical) or mathematical equation. By assigning objects (Fig. 1) with mathematical or physical properties, they can develop a videogame (Fig. 2), where the correlation of meanings, helps build more complex models.

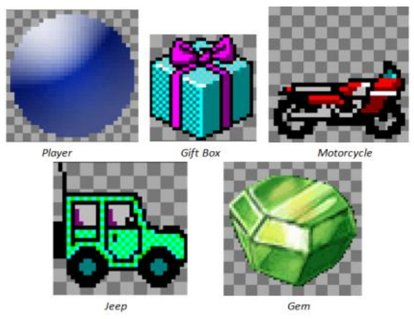


Figure 1. Objects used in Victor's videogame

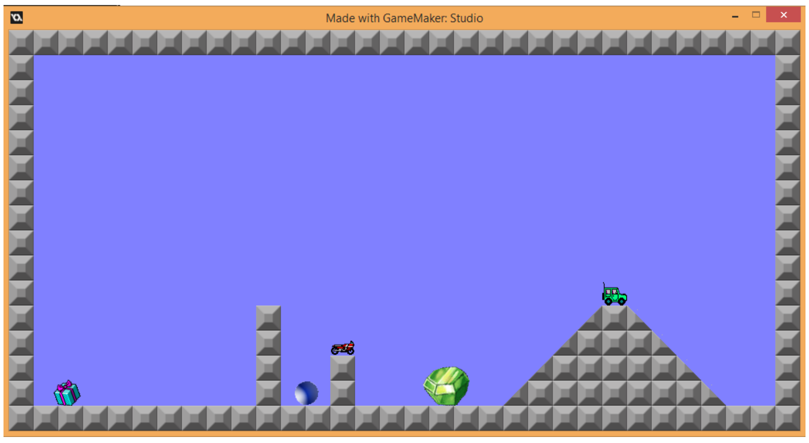


Figure 2. Screenshot of Victor's videogame experiment

Keywords

Constructionism, videogames, physical parameters, mathematical modelling, coding

Overview of the theoretical framework

Our approach of students engaging in the construction (design and programming) of videogames is based on the constructionism paradigm (Papert & Harel, 1991), which considers that learning is facilitated when the learner engages in the active construction and sharing of external objects. We follow in this, works such as Kafai's (1995) learning-by-design. We use this approach to help students implement what they know about mathematical modelling in simulations of physical systems that will be programmed into the videogames. As explained in Pretelín-Ricárdez & Sacristán (2015), the activity sequence is framed in the context of a microworld that takes into account the four components (student, contextual, technological and pedagogical) described by Hoyles and Noss (1987). The sequence of activities was also designed taking into consideration the six principles of Model-eliciting Activities (MEA): defined by Hamilton, Lesh, Lester, Brilleslyper (2008): reality, model construction, model documentation, self-evaluation, model generalization, simple prototype; although in this paper we do not focus on this.

Description of the videogame design introductory stage

As explained above, in this paper we focus on the introductory stage of the videogame design activity sequence. The aim of this stage is for students to learn how to use the videogame engine (Game Maker Studio), through the design and implementation of simulations and games.

To achieve this, we ask students to design and implement a simulation in the game engine, as an experiment or series of experiments, to analyse and characterise the following physical parameters: density, restitution, friction, linear damping, angular damping, velocity, force, impulse and acceleration of gravity. Some students designed only simulations and others implemented their simulations in simple videogames. This preliminary stage of successive experiments creates the foundation for later constructing more complex models and simulations implemented in a complete videogame, as illustrated in Fig. 3.

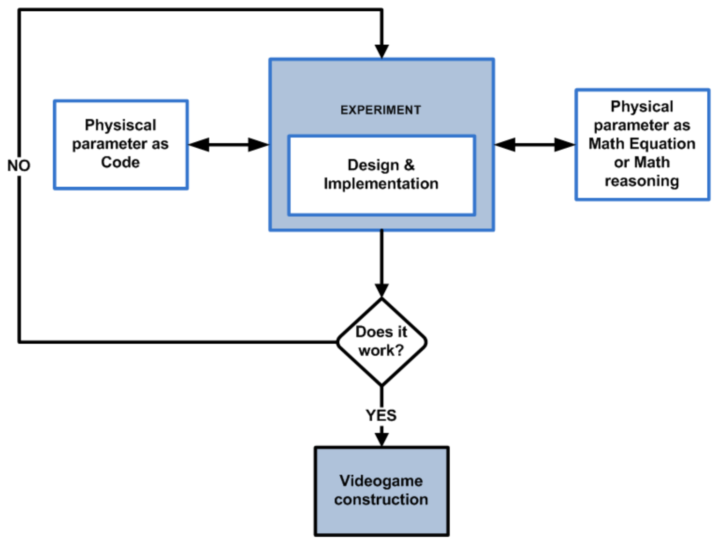

Physiscal parameter as Code
EXPERIMENT
Design & Implementation
Physical parameter as Math Equation or Math reasoning
NO
Does it work?
YES
Videogame construction

Figure 3. Development cycle of the experiments

For the examples given in this paper, the introductory stage was carried out over the course of eight sessions of 1.5 hours each. The examples correspond to four undergraduate engineering students in their final year of studies: Jorge, Víctor, Carlos, and Moisés. During each session, the instructor (the first author of this paper) had a facilitator role, solving specific questions or providing digital resources needed by the students. Students documented their progress through logbooks. Sessions and students' interviews were video recorded.

In the following sections we briefly describe some of the constructions that students created during their simulation or videogame experiments for trying out the parameters that could be used in their final videogames.

Example 1. The Sonic Bombs Videogame: Impulse and Force

This experiment was designed by Jorge as a videogame. In this example, we only show the preliminary design phase, with parts of the storyboard, without giving details of how the physical parameters are implemented into the videogame. The following description is taken from his logbook: The objective of the videogame is that the player must go from a starting point to an endpoint (flag), where both points are separated by a gap of several meters (see Fig. 4). The way in which the player must move from one point to another is by jumping and "flying" using the impulse provided by concussion grenades.

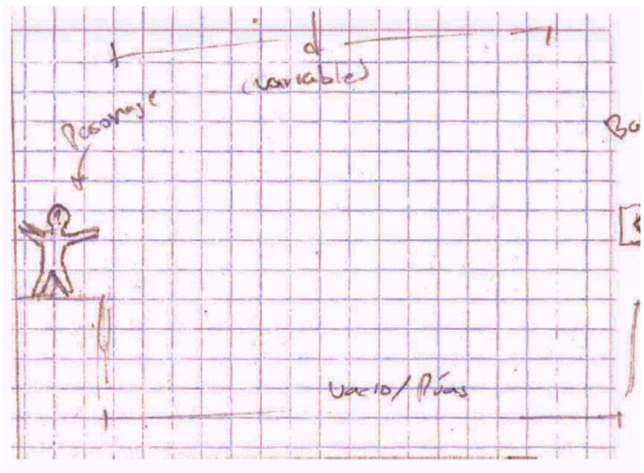


Figure 4. The scenario of the game in Jorge's storyboard

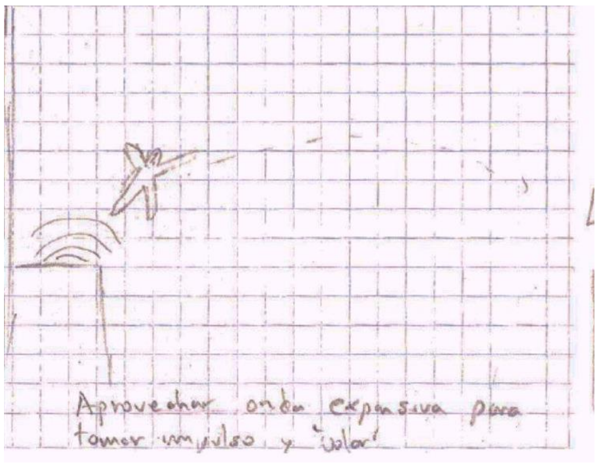


Figure 5. A jump using the grenade in Jorge's storyboard

Concussion grenades are a type of grenade that explodes after a certain time from activation, generating a sonic shock wave (see Fig. 6) that will help the characters increase their velocity, giving them an impulse to reach other places, without generating damage.

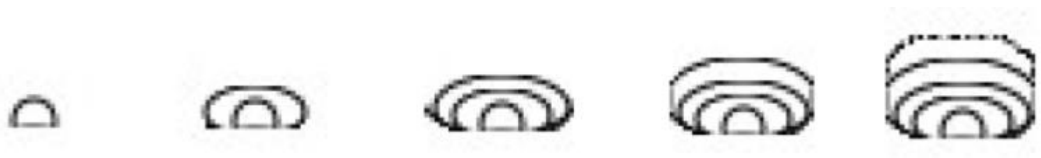
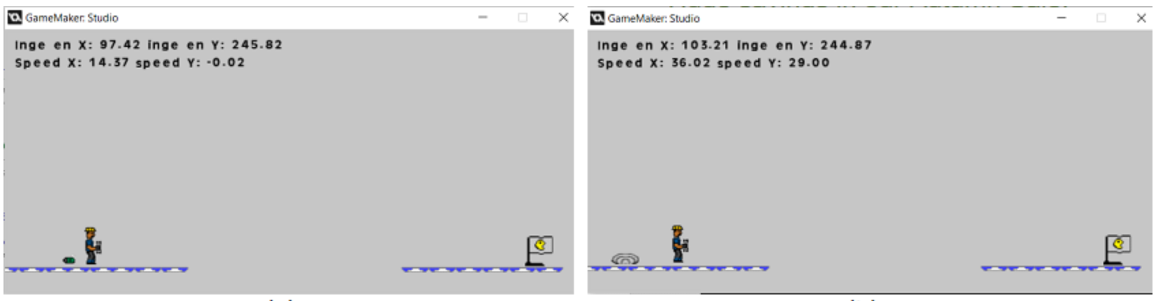


Figure 6. Design of the explosion sequence of the grenade

In Figure 5, we show how the character uses the grenade to increase its speed through an impulse. Figure 7 shows screenshots of the first "draft" or iteration of the actual videogame.



```
Inge en X: 97.42 Inge en Y: 245.82
Speed X: 14.37 speed Y: -0.02
```

(a)

```
Inge en X: 103.21 Inge en Y: 244.87
Speed X: 36.02 speed Y: 29.00
```

(b)

Figure 7. Screenshots of the first iteration of Jorge's Sonic Bombs videogame

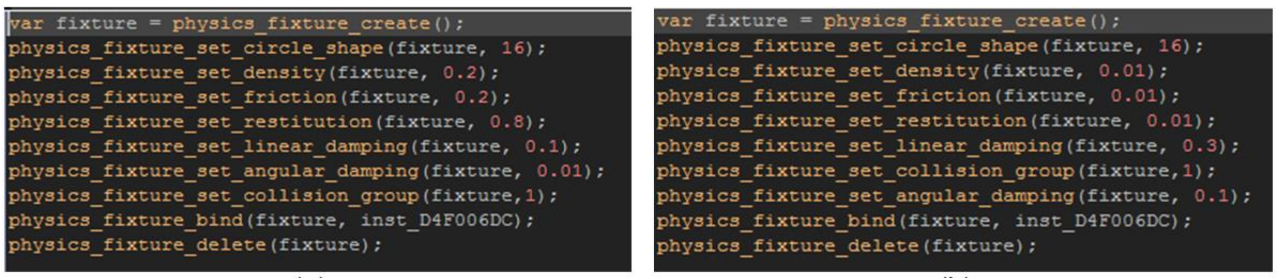
A video of Jorge's videogame experiment can be seen in the following link:
https://dl.dropboxusercontent.com/u/80597462/Constructionism2016/Sonic%20Bombs.mp4

Example 2: Variation of physical parameters on different objects

This videogame experiment was designed by Víctor, who wrote the following in his logbook: "In this experiment, we want to observe the interaction between objects with different physical parameters". Víctor's objects (a player, a gift box, a motorcycle, a Jeep car, and a gem) are shown in Figure 1. And Table 1 shows the physical parameters of each of these objects.

Table 1. Physical parameters of objects

	Player	Gift Box	Motorcycle	Jeep	Gem
Density	0.2	0.01	0.5	0.7	0.8
Restitution	0.8	0.01	0.1	0.1	0.01
Linear damping	0.1	0.3	0.4	0.5	0.9
Angular damping	0.01	0.1	0.5	0.7	0.6
Friction	0.2	0.01	0.5	0.7	0.9



```
var fixture = physics_fixture_create();
physics_fixture_set_circle_shape(fixture, 16);
physics_fixture_set_density(fixture, 0.2);
physics_fixture_set_friction(fixture, 0.2);
physics_fixture_set_restitution(fixture, 0.8);
physics_fixture_set_linear_damping(fixture, 0.1);
physics_fixture_set_angular_damping(fixture, 0.01);
physics_fixture_set_collision_group(fixture,1);
physics_fixture_bind(fixture, inst_D4F006DC);
physics_fixture_delete(fixture);
```

(a)

```
var fixture = physics_fixture_create();
physics_fixture_set_circle_shape(fixture, 16);
physics_fixture_set_density(fixture, 0.01);
physics_fixture_set_friction(fixture, 0.01);
physics_fixture_set_restitution(fixture, 0.01);
physics_fixture_set_linear_damping(fixture, 0.3);
physics_fixture_set_collision_group(fixture,1);
physics_fixture_set_angular_damping(fixture, 0.1);
physics_fixture_bind(fixture, inst_D4F006DC);
physics_fixture_delete(fixture);
```

(b)

Figure 8. Scripts to define the physical parameters of the Player object before a collision (a) and after a collision (b).

For the experiment, Víctor decided that the "Player object" must changes its physical parameters each time it collides with another object; that is, the Player object will acquire the parameters of the object that it touches. This interchange of parameters is carried out through the scripts shown in Figure 8. Figure 8a shows the initial parameters of the Player object and Figure 8b shows what happens when the Player object collides with the Gift Box object.

A screenshot of Víctor's videogame experiment is shown in Figure 2 at the beginning of the paper. And a video of his experiment can be seen in the following link: https://dl.dropboxusercontent.com/u/80597462/Constructionism2016/Densidades.mp4

Example 3. "Donquey Cong": Checking physical parameters of some objects

This videogame experiment was designed by Carlos, who described it in the following way is in his logbook:

> We check the physical parameters of the objects [Fig. 9]. We will put these objects in an inclined plane to analyse how they will behave according to their density, size, shape, and friction generated between each other and with their environment.

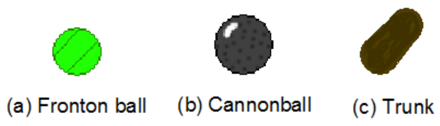


Figure 9. Objects used in Carlos's videogame experiment

> We assigned to each object, physical parameters as in reality. We also designed a scenario [see Fig. 10] composed of four ramps, a starting-point where objects are created and an end-point where objects will be destroyed. Each ramp will be inclined 2° less than the previous one, starting from top to bottom, with the first ramp at 10°; therefore the final ramp will only be at 4°. The scale used 1 m = 32 pixels. The value of gravity acceleration was approximated to 10 m/s^2. […]
>
> In order to estimate [the parameters that will control] the movement of each object, a mathematical model is defined to represent the object's movement in the different inclined planes [inside the scenario]. [In Table 2], we show the calculations for the trunk object.

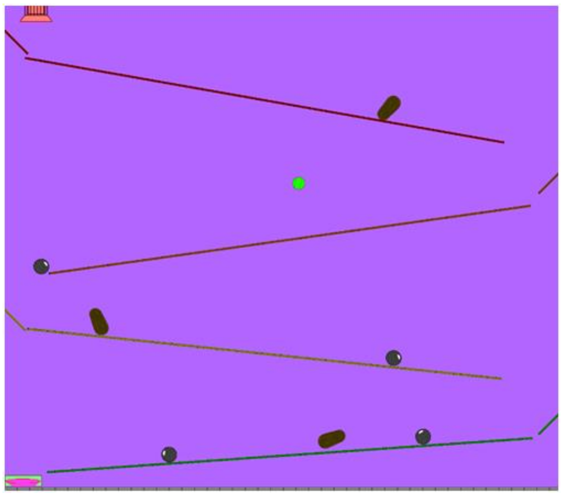


Figure 10. Screenshot of Carlos's videogame experiment

Table 2. Results of the analysis for the Trunk object

Ramp	Inclination	Acceleration (a_x) $[m/s^2]$	Time (t) $[s]$	Final velocity (v_f) $[m/s]$
2	8°	3.37	3.37	11.38
3	6°	3.03	3.55	10.79
4	4°	2.69	3.77	10.17

A video of Carlos's videogame experiment can be seen in the following link: https://dl.dropboxusercontent.com/u/80597462/Constructionism2016/Donquey%20Con.mp4

Experiment 4: Simple pendulum and control of an inverted pendulum

A pendulum is a mechanical system that can be used to experiments with the relationship between several physical parameters and the acceleration of gravity. If the mass (m) at the end of a simple pendulum joins a rigid bar and it is placed upside down, then we have an inverted pendulum, which is a system of unstable nature: this is a classic example for the study of automatic control (Fig. 11).

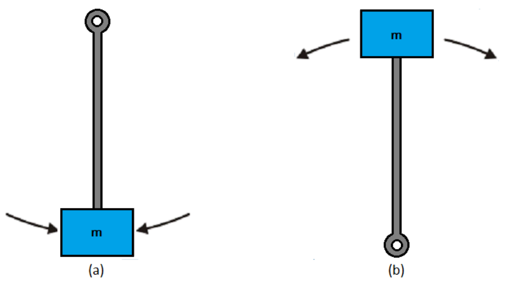
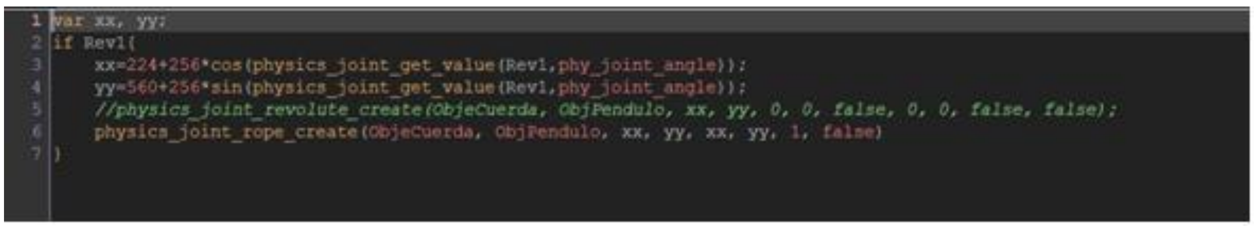


Figure 11. (a) Simple and Pendulum (b) inverted pendulum

This simulation experiment was designed by Moisés. The following description is taken from his logbook.

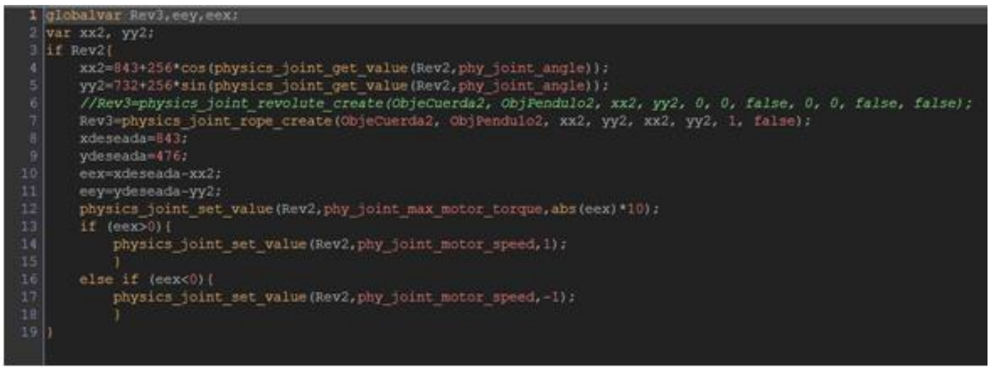


```
var xx, yy;
if Rev1{
    xx=224+256*cos(physics_joint_get_value(Rev1,phy_joint_angle));
    yy=560+256*sin(physics_joint_get_value(Rev1,phy_joint_angle));
    //physics_joint_revolute_create(ObjeCuerda, ObjPendulo, xx, yy, 0, 0, false, 0, 0, false, false);
    physics_joint_rope_create(ObjeCuerda, ObjPendulo, xx, yy, xx, yy, 1, false)
}
```

Figure 12. Code to create joints in motion for the simple pendulum and the inverted pendulum

We work a proportional control to balance an inverted pendulum using the following formula: Out = K_p*e (t), where: Out is the output signal (controller output), K_p is the proportionality constant, which determines the degree of amplification of the control element; e(t) is the error signal with respect to time. Next, we show the code developed for the creation of the joints sprites, for a simple pendulum and inverted pendulum inside the game engine [Fig.12].

Then we made some changes to achieve the swinging of the inverted pendulum, and we seek the proportional control by calculating the error between the desired position and the current position. [Fig. 13].



```
globalvar Rev3,eey,eex;
var xx2, yy2;
if Rev2{
    xx2=843+256*cos(physics_joint_get_value(Rev2,phy_joint_angle));
    yy2=732+256*sin(physics_joint_get_value(Rev2,phy_joint_angle));
    //Rev3=physics_joint_revolute_create(ObjeCuerda2, ObjPendulo2, xx2, yy2, 0, 0, false, 0, 0, false, false);
    Rev3=physics_joint_rope_create(ObjeCuerda2, ObjPendulo2, xx2, yy2, xx2, yy2, 1, false);
    xdeseada=843;
    ydeseada=476;
    eex=xdeseada-xx2;
    eey=ydeseada-yy2;
    physics_joint_set_value(Rev2,phy_joint_max_motor_torque,abs(eex)*10);
    if (eex>0){
        physics_joint_set_value(Rev2,phy_joint_motor_speed,1);
        }
    else if (eex<0){
        physics_joint_set_value(Rev2,phy_joint_motor_speed,-1);
        }
}
```

Figure 13. Code for inverted pendulum swinging

Figure 14 shows a screenshot of Moisés's simulation.

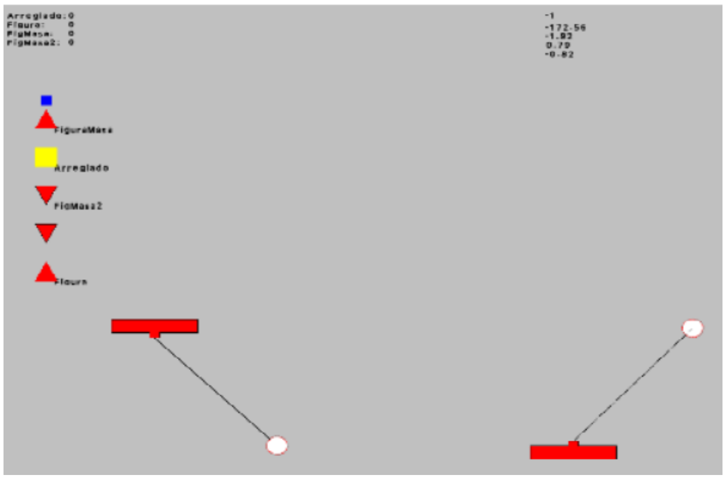


Figure 14. Simulation of a simple pendulum and inverted pendulum control

A video of Moisés's experiment can be seen in the following link:
https://dl.dropboxusercontent.com/u/80597462/Constructionism2016/Pendulo.mp4

Students comments regarding the videogame design activity

Several students mentioned how much they appreciated these experiments, mentioning that they provided the only opportunity in their engineering studies for actually simulating mathematical models as they would behave in real life. Other students' comments regarding their appreciation of these experiments were:

- **Moisés:** When we do these experiments, we become aware of how many errors can occur in the construction of systems: programming errors and errors in assigning values due to the restrictions of the physics engine. I felt a great satisfaction after being able to simulate and control a pendulum, because this shows that we can simulate any type of system in 2D – from a pendulum to robotic arms–, which is helpful to visualise the behaviour of these systems, without the need of deep programming knowledge to create them; even with the [basic] understanding and modelling of these systems we can create quite complex games.

- **Víctor:** These activities were different... here; you have a set of parameters to "play" with... You can perform an experiment, and then another, and then another experiment, with many parameters easily. This is faster, more accessible and also fun to experiment.

Final Remarks

With this activity, we explored and observed how students, through their experimentation, can express a mathematical and physical meaning through the programming codes or commands. We believe that this kind of experimentation can plant the seed of powerful ideas, which in this case will allow the construction (representation and understanding) of the mathematical modelling of more complex physical systems.

References

Hamilton, E., Lesh, R., Lester, F., Brilleslyper, M. (2008). Model-Eliciting Activities (MEAs) as a bridge between engineering education research and mathematics education research. *Advances in Engineering Education*, 1(2), 1-25.

Hoyles, C., Noss, R. (1987). Synthesizing mathematical conceptions and their formalization through the construction of a Logo-based school mathematics curriculum. *International Journal of Mathematics Education in Science and Technology*, 18(4), 581–595.

Kafai, Y.B. (1995). *Minds in play. Computer game design as a context for children's learning*. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Papert, S., & Harel, I. (1991). Situating Constructionism. In: S. Papert & I. Harel (Eds.), *Constructionism*. (p. 1-12). New Jersey, NJ: Ablex Publishing Corporation.

Pretelín-Ricárdez, A. & Sacristán, A. I. (2015). videogame construction by engineering students for understanding modelling processes: the case of simulating water behaviour. *Informatics in Education*, 14(2), 265-277. doi:10.15388/infedu.2015.15

Report from the Future: The Next Generation High School

Brian Harvey, *bh@cs.berkeley.edu*
Computer Science Division, University of California, Berkeley

Abstract

The US government kicked off a "next generation high school" project with a week of events in November, 2015. I attended the National Science Foundation forum "Next Generation STEM Learning for All" November 9 and the White House "Summit on Next Generation High Schools" November 10.

There is a lot of good news for constructionists from these events. The emphasis is overwhelmingly on project-based learning, and at least some of the featured model schools talk about the importance of learner-chosen projects and of public display of the results. Figure 1 shows two screenshots from a film, *Most Likely to Succeed,* that presents this initiative. The two segments compare what the film calls 20th Century and 21st Century learning.

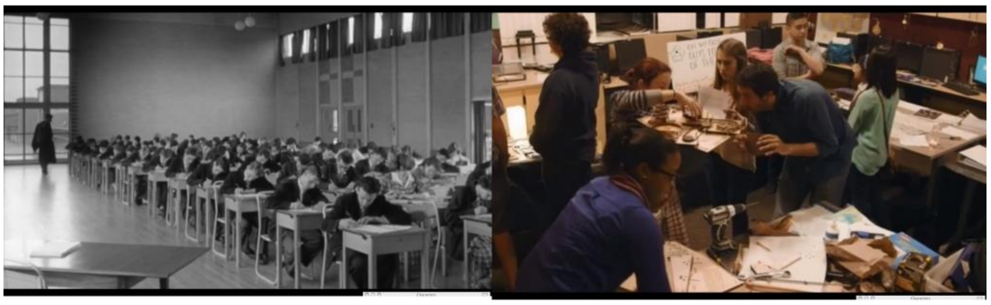


Figure 1. Then and now

There is also some bad news: First, the "next generation" narrative presents the 20th Century as if it were nothing but rote learning, ignoring the strong progressive education movement that started several centuries (or, in some views, millennia) ago. Second, more importantly, the narrative is posed entirely in terms of job requirements—the 20th Century had factory jobs, but the 21st Century has information jobs—as if job training were the sole, or at least main, purpose of education.

Keywords

STEM education; next generation high school; maker spaces; project-based learning; information economy

A Sharp Turn in US Education Policy

The "No Child Left Behind" law was proposed by President George W. Bush in 2001 and signed into law in 2002. It called for a massive increase in standardized testing, and required that the tests be used not only to evaluate students but also to evaluate teachers and schools, with punitive consequences for schools with below-average test scores. From a progressive education standpoint, the results were disastrous: A successful education was defined to mean the learning of facts and the ability to solve exam problems; both the tests themselves and the days of practice tests and similar test-driven activities took over time from real learning; the tested subjects (mostly English and mathematics) drove out other school activities from science and art to recess; the need to race through the tested curriculum eliminated time for learning in depth; teacher morale was destroyed.

When Barack Obama was elected in 2008, progressive educators hoped that he would reverse this test-driven approach. Alas, that didn't happen. Obama appointed as his Secretary of Education the then-CEO of the Chicago Public Schools, Arne Duncan. The highlights of Duncan's Chicago career were his efforts to destroy the teachers' union and to replace public schools with charter schools. Obama and Duncan introduced what they called the "Race to the Top" program that made state education grants dependent on policies promoting charter schools and "performance-based" evaluation (largely based on test scores) of teachers and of schools. In 2014, the two national teachers' unions both called for Duncan's resignation. He did, in fact, resign in 2015, but his replacement, John King, Jr., is a Duncan protégé who was expected to continue Duncan's policies.

It is, therefore, quite a surprise to find this administration, only a month after Duncan's resignation, introducing with great fanfare a largely progressive, constructionist blueprint for the high schools of the future.

Next Generation STEM Learning for All

The National Science Foundation is a leading funder of education reform efforts in science and mathematics. The "New Math" curriculum redesign effort following the US reaction to Sputnik was led by NSF. For 50 years, NSF has championed inquiry-based learning. The current Next Generation program is therefore in keeping with past NSF policy. NSF has also long been interested in educational equity, promoting progressive curricula for all students and not just for an elite.

One of the significant achievements of US education in the past decade has been to attract girls to math and science, formerly almost entirely subjects for boys. Today, girls are half or more of the students taking Advanced Placement courses in these subjects—except for computer science, still around 90% male. The NSF has invested heavily in several initiatives designed to pursue equity for girls and for ethnic minority students in CS. One of those initiatives, called *CS Principles,* funded the development of Berkeley's "Beauty and Joy of Computing" curriculum; that's how I ended up at the NSF forum and the White House summit in Washington.

In addition to government education officials, plenary speakers included the director of the High Tech High charter school, the principal of the public Chicago High School for Agricultural Sciences, the vice president of Education of Battelle Memorial Institute (a nonprofit R&D group), the president of Advanced Reasoning In Education (a private teacher professional development company), and various academic researchers.

The school leaders, in particular, emphasized (without using the word) constructionist activities: hands-on, generally but not always learner-directed, and generally culminating in a presentation to other students, parents, and sometimes outside technology leaders.

I attended a concurrent session about assistive technology for students with disabilities, a topic not central to this paper but important to me because of the need to enhance blocks-based programming languages such as Snap*!* to support visually impaired learners. In the other concurrent timeslot I was one of a panel on “Advancing STEM + Computing in K-12 Education.” Other panelists spoke on

- Integrating “computational thinking” into non-CS science classes
- Diversity in the “Teach for America” STEM initiative
- CS Principles for students with learning disabilities
- Mentoring and broadening participation

(My own contribution was titled “Programming is Cool Again,” in which I argued against the view, enshrined in the CS Principles framework, that girls are turned off by programming.)

What all of these presentations shared (and the other sessions had a similar focus) was a strong commitment to diversity as a *central* goal of STEM education design, not something to be addressed only at the school level.

Other concurrent sessions addressed research experiences for teachers, collaborative learning environments, peer teaching and mentorship, “tomorrow’s cyber workforce,” instructional materials, “authentic discovery,” schools and communities, workplace-focused learning, and engaging students in entrepreneurship.

Next Generation High Schools

Much of the White House Summit on Next Generation High Schools was devoted to funding commitments: a total of $375 million, including $20 million of federal funding, along with 100 or so private commitments ranging from small efforts by individual schools up to $100 million programs from large foundations. Thirty commitment announcements were presented in “lightning talks”; the most interesting were from the individual schools and school districts, many of which showed video presentations of actual student projects.

There were longer presentations from eight White House and Department of Education officials, varying widely in their understanding of educational issues. (The low point, for me, was a remark by the US Chief Technology Officer—who, in fairness, does not deal with education as her main job responsibility—about her visit to a new high-tech school where one of the teachers had invented a great idea: “ask three, then ask me,” really a commonplace technique in Logo classrooms of the 1980s.) The thrust of these presentations, though, was a strong shift toward viewing student learning in terms of real skills evidenced by real projects, rather than by high-stakes testing. Of course these officials couldn’t say straightforwardly “we were wrong about testing,” but it was clear that most of them understood that.

Participants were assigned more or less randomly to small group discussion sessions; mine was about maker spaces. The high point of the discussion was essentially complete agreement that it doesn’t count as a maker space if the teacher chooses the project. The low point was that nobody agreed with me that maker spaces don’t give grades. The others interpreted this as “no feedback” or “no guidance”; I couldn’t convince them that, on the contrary, once you’re not giving grades, you can give *more* feedback, *more* guidance—more tests, even—without students feeling threatened. Everyone at the summit talked about the importance of risk-taking as the way to learn, but it’s really hard to take risks when at the end of the day you’re going to be graded on your success. Even grades based on effort rather than success make risk-taking, and learning, harder.

Conclusions

The good news: If the ideas presented at these events are actually carried out in schools of the future, learning will be quite constructionist: project-based, hands-on, often learner-directed, including assignments that promote metacognition, and culminating in public presentation of the student work. This progressive agenda will apply not only, as often in the past, to elite schools for rich kids, but across the board, with special emphasis on equity and inclusion.

The bad news: This tremendous reform is based almost entirely on rhetoric about jobs and competitiveness. The US is falling behind (#27 in math!), and traditional jobs are quickly being lost to automation. In the old days, Americans designed (and American companies owned) the technology that China mass-produced. But China is now ahead of the US in the education rankings, so if we don't overtake them, they'll design the technology themselves.

The US panic over Sputnik led to some really good educational initiatives, and the current panic over China may have good results also. But rhetoric about job skills doesn't leave much room for a learner's personal growth or personal goals; there will probably still be artists, writers, musicians, psychologists, and comedians in the 21st Century, but they aren't going to help us beat China, and so it's not clear that Next Generation schools will nurture them.

(I should note that the emphasis on jobs is partly also driven by equity concerns. Minorities in the US are most likely to be poor, to be unemployed, even to be in prison. The desire to prepare them for secure, well-paid jobs is not entirely a matter of panic about China.)

Also, most of the speakers at the White House event seemed not to know about the long history of progressive education. Yes, the factory jobs of the 20th Century have a lot to do with the typical factory-like school environment with bells between periods and chairs screwed down to the floor. But the 20th Century also had Montessori, Dewey, Freire, Illich, Holt, Kozol, Neill, and many other pioneering leaders in student-centered education. Oh, and Papert! A decent education can be motivated by things other than the requirements of information-age jobs.

Resituating Constructionism in the Space of Reasons

Kate Mackrell, *katemackrell@mac.com*
University College London, Institute of Education

Dave Pratt, *d.pratt@ioe.ac.uk*
University College London, Institute of Education

Abstract

C*onstructionism* is described by diSessa and Cobb as a 'framework for action', and is built upon the orienting framework of constructivism, together with an assumption, born out of practice, that engaging students in making activity (bricolage) is especially felicitous for learning. It has been further elaborated through the work of Noss and Hoyles to include constructs such as *situated abstraction* and *webbing*, and through the work of Ainley and Pratt to include *purpose* and *utility*. However, constructivism is increasingly being criticised as a model for learning (Roth, 2011): a major issue is that it tacitly endorses the Cartesian mind/world split in which learning consists of constructing knowledge (representations) of the world in the mind. Furthermore, constructionism seeks to add the affective to the account of intellectual development proposed by constructivism without providing theoretical underpinning for this addition. Hence constructivism no longer serves as an adequate orienting framework for constructionism.

We suggest that contemporary work within philosophy (Brandom, 2000; Bakhurst, 2011; McDowell, 1996) provides a more fitting narrative. We argue that the idea of learning as initiation into the space of reasons (Bakhurst, 2011) provides grounds for a synthesis of the affective and the logical. We argue that inferential reasons and reasoning encompass not only the powerful ideas of mathematics and disciplinary knowledge of modes of enquiry but also the extra-logical, such as in feelings of the aesthetic, control, excitement, elegance and efficiency.

Keywords

space of reasons; constructivism; constructionism; purpose and utility; reasons

Introduction: The emergence of constructionism

In *Mindstorms: Children, Computers, and Powerful Ideas* (1982), Papert's vision was of children learning mathematics through deep and personal engagement with powerful mathematical ideas embodied in microworlds, environments in which the child could explore and play with mathematical ideas in their creation of personal self-driven projects. Programming in *Logo* lies at the heart of his vision, promoting the meaningful use of mathematical ideas such as distance, angle, and variable. The central tenet that children could take ownership and control of their own learning stood – and still does stand - in direct opposition to the prevailing approach to school-based teaching and learning of mathematics. These ideas were formalized into constructionism, which shares with Piagetian constructivism the idea that knowledge is constructed by the individual learner, but adds the idea that learning is most effective when experienced as constructing a meaningful product, and stresses the importance of affect.

We claim that the foundation of constructionism on constructivism is problematic and seek to provide a different foundation in ideas drawn from the contemporary philosophy of mind, which will both deal with an intrinsically problematic aspect of constructivism and include affect.

Constructivism and constructionism

Types of theory

diSessa and Cobb (2004), in arguing that theory in design-based research "must do real design work in generating, selecting and validating design alternatives" (p. 80) set out a taxonomy of theories in order to understand the contribution that different types of theory might make towards that aim. They described constructionism as a framework for action: a set of prescriptions for pedagogical strategies, often effective heuristics providing focus and direction to the design of a learning environment, but which fail to separate scientific claims from suggested actions.

These strategies derive from more general grand theories and orienting frameworks, which for the purposes of this paper we will refer to as orienting theories. It is clear from Papert's (1982) own accounts that Piaget's work, the basis for constructivism, provides the orienting theory for Papert's vision of constructionism. One aim in this paper is to argue that this relationship between constructionism and constructivism is problematic.

Connections and discontinuities between constructivism and constructionism

Papert adopted Piaget's idea that knowledge is constructed by the learner but saw his own perspective as more interventionist, involving education as well as understanding. He saw himself as elaborating Piaget's work in order to facilitate the development of different intellectual structures having both logical and emotional form through the design of resonant learning environments. A major aspect was his consideration of the affective component, later elaborated as one aspect of the extra-logical, which also included the aesthetic and the intuitive.

Papert also used the "child as builder" metaphor to consider the materials needed to build, and saw the surrounding culture, and in particular accessible computers, as a major source of these materials, He accepted Piaget's distinction between concrete and formal thinking but felt that objects such as computers could shift the boundary separating the two, enabling knowledge that had previously only been accessible through formal processes to be approached concretely.

We note how, over the years, some issues have been progressively elaborated to place increasing emphasis on the extra-logical. In *Mindstorms,* Logo is described as arising from an interest in enabling the construction of particular mathematical structures, while in *Constructionism* (Harel and Papert, 1991), the roots of Logo are seen more in the search for ways of learning that would deeply engage students. This connects with the Mindstorms idea of the importance of the non-logical in learning, but Papert has moved on from expressing this as in opposition to the logical: it now involves appropriating mathematics in a deeply personal way.

The problem arises: Historical/Philosophical perspective

In 1639, Rene Descartes proposed that the mind, the "I" who thinks, wills, understands and perceives, and the body, a machine that produces representations of perceptible objects for the mind, are distinct. He also proposed that it is by thinking/reasoning that we reach the truth, whereas beliefs based on the senses may be mistaken.

One consequence is the prevailing idea that perception and knowledge are based upon representations. Locke and then Hume countered Descartes' emphasis on reasoning with the insistence that knowledge came from experience and empirical investigation was necessary. Kant, in synthesizing the role of reason and empirical investigation in developing knowledge, held that the world of appearances was constructed by the human mind from sensory matter via a priori cognitive forms in cognition, and that things-in-themselves can only be known through such appearances, which take the form of mental representations (Rohlf, 2014). Piaget in turn derived his "genetic epistemology" from Kant, and Piaget has been recognized as "the great pioneer of the constructivist theory of knowing" (von Glasersfeld, 1990).

Cobb, Yackel, and Wood (1992) stress that constructivism in itself does not involve representationalism and that it can be used to bridge the issue with the mind/world divide. However, although they themselves are careful to talk about "ways of knowing", they point out that many constructivists have uncritically used the idea of knowledge as representation. The idea also persists today. Ernest, well-known for his philosophy of mathematics education says the following in a criticism of radical constructivism:

> Its representations of the world and of other human beings are personal and idiosyncratic. Indeed, the construal of other persons is driven by whatever representations best fit the cognizing subject's needs and purposes. (Ernest, 2010, p. 41)

Ernest criticises the idiosyncrasy of such representations – but seems to be tacitly accepting the idea of "representation-as-knowledge".

Perhaps particularly in considering the learning of mathematics, in which important "objects" (such as functions) are not tangible and hence are commonly conveyed by means of representations (such as the graph of a function), the temptation is to see learning as the construction of representations; perhaps this is even more the case in constructionism, where constructing a real (or virtual) product is seen as a particularly effective means of learning. While it is beyond the scope of this paper to elaborate the important role of representations in mathematics, we suggest that a mathematical representation is a tool which facilitates mathematical thought and action: the ability to visualize and draw conclusions from the mental image of a number line is the mathematical knowing, not the image itself.

The second is the neglect of affect in constructing knowledge. Objectivity in scientific enquiry has prioritized the neutral, objective observer, unaffected by personal responses. The possibility and desirability of such an observer have been questioned for some time and affect is certainly on the agenda in mathematics education research. However, as noted by Roth (2011), there is no way of incorporating emotion into constructivism as it stands: it remains an awkward add-on.

An Alternative to Constructivism

There are a number of ways currently in which the mind/body or mind/world split is being addressed in mathematics education, such as enactivism, derived from biology (see Reid (2014) for a comprehensive account of enactivism in mathematics education research). We will introduce an alternative in this paper. The problem arose in philosophy: are any of the ways in which the problem is currently being addressed in philosophy of relevance in mathematics education, and more specifically to the issue of providing a theoretical basis for constructionism?

The first issue is to show that there is no need for representations in knowing the world. McDowell (1996) uses arguments from Kant and more recent philosophers such as Sellars and Davidson to in effect erase the divide between mind and world. He claims that we do not take in raw data that the mind has to work up but that perception is already conceptual, fit to yield judgement and serve as ground for belief (Bakhurst, 2011). There is hence no need for the mind to construct its awareness of the world.

The second issue is to identify what learning consists of if it is not about constructing knowledge. We first introduce the term, "the space of reasons":

> "in characterizing an episode or a state as that of *knowing*, we are not giving an empirical description of that episode or state; we are placing it in the logical space of reasons, of justifying and being able to justify what one says" (Sellars, 1963, p. 169)

In one sense, this is simply restating the traditional philosophical definition of knowledge as justified true belief, but the image has a much wider scope. McDowell (1996) would include any phenomena, such as behaviour, beliefs and desires which can be made intelligible as manifestations of a responsiveness to reasons. Behaviour becomes intelligible when seen as intentional action. For example, the action of a person opening a refrigerator is made intelligible when we know that the person is thirsty and wants to get some juice to drink. McDowell contrasts space-of-reasons intelligibility with that of the "realm of law" i.e. intelligibility due to identifying phenomena as describable by laws based on empirical investigation. The question of how the refrigerator keeps juice cold would be appropriately answered by an account drawn from the realm of law.

Brandom (2000) sees standing in the space of reasons as an abstraction from the concrete practices of giving and asking for reasons, and as dependent on social articulation. He gives the example of a thermometer's and a human's response to the temperature dropping below 70 degrees to illustrate what it means to stand in the space of reasons: the human can assert that "it's 70 degrees outside" knowing both what would be evidence (or reasons) for such a claim and what would follow from the claim (what the claim would be reasons for). Another example he gives is the difference between the two year old and the seven year old who state that the house is on fire: the seven year old is making a claim, is standing in the space of reasons, as she knows that evidence for her claim is that the kitchen is full of smoke and flame and that it follows that the family is in danger and must flee. Brandom rejects the priority given to representations and argues that knowing is primitively constituted in the game of giving and asking for reasons. Representations are invented as a result of that game. Human utterances and actions are embedded with intent, which others attempt to interpret, and humans seek meaning in the discourse of others. By giving and asking for reasons, humans keep score of the 'correctness' of their meanings in order to seek out normativity.

Bakhurst (2011) has begun to develop a genetic account of knowing based on the ideas of McDowell and Brandom. He claims that human minds develop through initiation into "the space of reasons" such that our thoughts and actions are increasingly guided by what there is reason to think about or do. The thirsty baby can only cry: the thirsty adult acts on the belief that there is juice in the refrigerator and undertakes a set of actions which will ultimately result in drinking the juice – or alternatively gives the juice to the baby and then gets water for herself.

The alternative to constructivism that we propose is to consider the learning of mathematics as just such an initiation into the space of reasons, with focus on thoughts, feelings and actions that can be considered to be connected to mathematics. This would clearly incorporate both the logical structure of mathematical argument within mathematics itself (in contrast to enactivism, which cannot address the nature and growth of mathematics itself (Reid, 2014)) and also, for individual learners, the reasons why mathematics is considered important, and the personal reasons for engaging (or otherwise) in mathematics. Importantly, the focus on "reason" will also necessitate overt consideration of an ethical dimension in mathematics education. For convenience, we will refer to the "mathematical space of reasons", with the proviso that there is no firm boundary between this and the "space of reasons" as a whole.

An issue, however, is that it is very hard to use the word "reason" without falling back into a number of dichotomies. The first dichotomy is that of intellect and emotion. One of the definitions of "reason" (from the Oxford Dictionary) is "the power of the mind to think, understand, and form judgments logically", which gives no link with emotion. However, if we use reason as "a cause, explanation, or justification for an action or event" (also from the Oxford Dictionary) this difficulty does not arise: reason encompasses both creating a rigorous proof of Pythagoras' theorem and also the decision to not engage with attempting to understand such a proof because of anxiety and lack of confidence.

McDowell's emphasis on persons as "inhabitants of the space of reasons" has been criticised as painting an unduly intellectualistic or rationalistic portrait. Bakhurst (2011) discusses this misconception, portraying reasons in the contexts of moral sensibility, passivity and creativity and also mood and responsiveness to music.

The second dichotomy is the precise "mind/world" split that we are attempting to avoid. It is very easy to picture ourselves as having internal reasons (in our minds) that cause our external actions (in the world), or to consider the "space of reasons" as an inner, mental space. McDowell (1995) is adamant that this is not the case. Recent research in cognitive psychology is also helpful here, assuming that experience of "mind" arises from experience of conscious thought. In a review of the limitations of conscious thought, Masicampo and Baumeister (2013) cite evidence that conscious thought is too slow to initiate behaviour: unconscious processes are much earlier indicators of action, even for complex decisions, and also that unconscious processes are capable of initiating and guiding action. Masicampo and Baumeister also describe research in which participants provide plausible but false reasons for their actions, or explain decisions that they did not in fact make. This is in stark contrast to Descartes' assumption that the mind is in control of the body, or Kant's assumption that our cognitive structure is transparent to us. The implication is that reason, as the explanation of action, cannot be equated with conscious thought, or the mind.

A tension arises, however. Both McDowell and Brandom argue for the importance of the articulation of reasons. Although McDowell does not equate articulation and reason, he considers the development of language as the entry-point into the space of reasons, and Brandom would consider that his seven year old who says the house is on fire is only entitled to her commitment to this claim if she is able to verbally justify it. Bakhurst differs in emphasis, however. In a vivid analysis of Mr. Gradgrind's classroom (from Dickens' *Hard Times*), he contrasts the boy who is considered to know what a horse is because he can list its properties with the girl who could not list these properties but who lived with horses and whose interaction with them showed a responsiveness to reasons, even if she could not articulate these. There is, however, an important aspect of the space of reasons involved in verbal articulation; Bakhurst sees our central task as opening up already-possessed views, however incoherent, to critique.

Further Developments in Constructionism

We have suggested that a consideration of learning as initiation into the space of reasons will alleviate the problem with learning seen as constructing representations and distinct from affect. We turn now to more recent developments in constructionism which have also dealt with these issues and show how these initiatives, while underscoring the disconnect between constructivism and constructionism, align with the idea of participation in the space of reasons.

A major issue in constructivism is the potential lack of correspondence between knowledge constructed by the learner and acceptable mathematical knowledge, well documented in the literature on misconceptions. In responding to this problem within a constructionist (and also constructivist) framework, Noss and Hoyles (1996) challenged the traditional view of mathematisation as moving the mathematics from action to cognition. Although they do not describe it as such, this view may be seen as a product of the mind/world split, in which actions-in-the-world need to be "internalized" as cognitions-in-the-mind. In contrast, Noss and Hoyles implicitly stress the connection between mind and world: "we want to put forward a case for learning as the construction of a *web* of connections – between classes of problems, mathematical objects and relationships, "real" entities and personal situation-specific experiences" (p. 105). A key construct is that of situated abstraction, both process and product, which describes the way in which learners construct mathematical ideas by drawing on this web in particular settings. They stress that abstracting “can be seen as a way of layering meanings, connecting between ways of knowing and seeing rather than replacing one kind of meaning with another” (p. 122).

We note that both situated abstractions and reasons arise from context. To explore further the connections between the two, we introduce Brandom’s idea of the inferential nature of the space of reasons. For Brandom, two propositions such as “there is smoke and flame in the kitchen” and “we must leave the house” are connected if one can be inferred from the other, possibly via the intermediary of other propositions such as “the house is on fire”, “we do not have a fire-extinguisher”, etc. Developing within the space of reasons involves increasing awareness of the inferential connections in the world. We can take this beyond propositions: we hug a friend who is feeling low in the belief that this will comfort her. Situated abstraction is likewise about connecting: learning is about increasingly knowing connections. However, the nature of such connections is left unexamined, which is problematic; it is tempting to see a situated abstraction as an inferential proposition drawn on the basis of what is known about a context and what this would entail in action, but this is not yet warranted. Brandom gives us a way to begin to discuss which connections may be important, which are incidental, and which, of the huge number possible in any moment of lived experience, may not be made at all. We hence think that there may be very fruitful comparisons made between the space of reasons and the web of connections – once “connection” has been elaborated.

The second development in constructionism focuses on the extra-logical aspect of constructionism. Ainley, Pratt and Hansen (2006) proposed the terms, ‘purpose’ and ‘utility’. *Purpose* refers to the perspective of the pupil. A purposeful task is defined as one that has a meaningful outcome or solution. Purpose may be quite distinct from the teacher’s objectives for mathematical learning. *Utility* refers to the sense of power for a mathematical idea that a child might appreciate when they see how mathematical concepts or methods enable them to make progress in their task. Purpose and utility is an attempt to capture the role of engagement in learning, so important in constructionism and yet such motivation is not intrinsic to constructivism. The importance of these ideas in the mathematical space of reasons is clear: “purpose” is the child’s reason to engage in an activity, while “utility” might involve both the reason that learning the mathematical idea is seen as desirable and the child’s growing responsiveness to the reasons embedded in the mathematical idea through its relationship to her own reasons for engaging in the activity.

A brief re-interpretation of constructionism in action

We have argued that the space of reasons provides a better theoretical basis for constructionism than does constructivism. We can gain a better foothold on this proposition by re-presenting some reported learning episodes as taking place in the space of reasons.

Pratt and Noss (2002) described a situation in which students were engaged in trying to mend a computer-simulated “broken” die. The students (10-11 years of age) simulated the tossing of the die and made the observation ‘the more times we throw the die, the more uniform is the pie chart’ (the pie chart displayed the number of occurrences of each side of the die). This statement is both a situated abstraction and, in Brandom’s terminology, a *commitment* to an inference about the die’s behaviour. The question of the students’ *entitlement* to this commitment arises: can the students explain why this is the case and what might follow from this? Brandom considers that interaction takes the form of *giving and asking for reasons* during which entitlement to commitments may be queried. In this case the on-screen behaviour of the die enabled the students to query their own entitlement to this commitment.

These reasons however include purpose and utility as might be better understood by a re-presentation of an episode from Mindstorms in which a child was shown how to control a cybernetic turtle, using commands such as FORWARD 100 and RIGHT 90. Being able to make the turtle move and then draw was sufficiently engaging to carry most children through the initial learning process. In our terminology, the child was being offered a reason to engage with the turtle, a purpose that the child readily adopted. In the terminology of Ainley et al (2006), this purpose may not in itself reflect the learning objectives of the teacher or indeed those of the designers of Logo. Papert described how the child needed to extensively explore these and other related commands to gain mastery, a process that involved the child in a sustained enquiry as to the reasons for the turtle's behaviour. The child had a clear purpose to activate the turtle and to do this the child needed to be able to explain how it operated.

The initial activity typically developed, according to Papert, into early programming, when the child was introduced to the metaphor of teaching the turtle a new word. Papert mentioned how the child might program new words such as SQUARE or TRIANGLE but crucially appended the phrase 'or whatever the child wishes'. The sense of control over the turtle was supplemented for the child by a sense of control over her own learning. For example, the child, who decided to teach the computer HOUSE, exploiting SQUARE for the walls and TRIANGLE for the roof, initiated her own interaction with the turtle involving asking for reasons that explain the turtle's behaviour and observing the feedback as the turtle giving reasons. The child began to appreciate the utility of programming to create movement.

Learning mathematics involves becoming inculcated into the practice of mathematics. From the perspective of a space of reasons, mathematical knowledge includes the modes of enquiry that are prioritised by the discipline and articulate the nature of the subject. Papert's approach places emphasis on these reasons. In the above example, the child's inquiry process involved experimentation, testing, accounting for behaviour and debugging, all targeted at inferring commitments, making them open to critique, and testing entitlement to hold these commitments.

In another episode from Mindstorms, Papert described the creation of a spiral by starting with the SQUARE program but making the side of the square a little longer at each step so that the shape spiralled outwards, and ultimately writing a short and elegant program using variable and simple recursion. The child recognised the beauty that mathematicians recognise in the recursive program when, with a little help, she created such a program for herself. The spiral program also offers an example of how variable can be a powerful means of communicating instructions to a computer. Mathematics teachers typically struggle with the teaching and learning of algebra because they lack resources to enable the child to give and ask reasons for variable. The desperate question so often asked by the child in the conventional maths classroom after several lessons of early algebra, "But what is this 'x'?", betrays the lack of utility that the child is feeling for variable. In contrast, making use of variable in order to create a spiral imbues algebra with utility. The mathematical knowledge about variable is made meaningful because it has become part of the space of reasons during the child's activity to create a spiral on screen. This episode shows how reasons can involve beauty, elegance and utility.

These episodes are intended to illustrate the diversity of reasons, given and asked for, fusing powerful ideas in mathematics, mathematical methods of enquiry, logic, utility, generalisation and abstraction, with purpose, aesthetic appeal, elegance and excitement.

Conclusion

In the preface to Mindstorms, Papert conveyed his own deep love of gears and how the turtle was an attempt to offer a general-purpose tool for mathematical thinking that performed a similar role to that undertaken by gears for him. However, Papert was unable to elaborate his sense of the affective aspect within the theoretical compass of constructivism. The link between constructivism and constructionism became increasingly tenuous, with the introduction of purpose and utility, webbing, and situated abstraction. Constructivism has also allowed a tacit, but questionable, identification between "knowledge" and "representation",

In this paper we have followed some contemporary developments in the philosophy of mind and conceptualized learning as initiation into the space of reasons, which incorporates all types of reasons: personal engagement, dialogue and critique, the embedding of reasons in artefacts and in the logical structure of mathematics itself. By stressing the importance of the inferential rather than the representational, we can move away from the metaphor of "constructing knowledge" to seeing mathematical knowing as participating in a web of interconnecting reasons.

These ideas connect well both with Papert's aims and with later developments in constructionism. We hence propose to replace constructivism with the space of reasons as an orienting framework for constructionism.

***Acknowledgement**: This work was supported by the Swedish Research Council (VR) for educational science*

References

Ainley, J., Pratt, D. and Hansen, A. (2006). Connecting engagement and focus in pedagogic task design, *British Educational Research Journal*. 32(1), 23-38.

Bakhurst, D. (2011). *The Formation of Reason*. Oxford, UK: Wiley-Blackwell.

Brandom, R. (2000). *Articulating Reasons: An Introduction to Inferentialism*. Cambridge, MA: Harvard University Press.

Cobb, P., Yackel, E., & Wood, T. (1992). A constructivist alternative to the representational view of mind in mathematics education. *Journal for Research in Mathematics Education*, *23* (1), 2-33.

diSessa, A. and Cobb, P. (2004). Ontological innovation and the role of theory in design experiments, *The Journal of The Learning Sciences*, 13(1), 77–103.

Ernest, P. (2010). Reflections on theories of learning. In B. Sriraman & L. English (Eds.) *Theories of Mathematics Education* (pp. 39-47). Berlin: Springer-Verlag.

Harel, I. and Papert, S. (Eds.) (1991). *Constructionism.* Norwood, New Jersey: Ablex.

Masicampo, E. & Baumeister, R. (2013). Conscious thought does not guide moment-to-moment actions – it serves social and cultural functions. *Frontiers in Psychology*, retrieved September 22, 2015 from http://journal.frontiersin.org/article/10.3389/fpsyg.2013.00478/full

McDowell, J. (1996). *Mind and world*. Cambridge, MA: Harvard University Press.

Noss, R., & Hoyles, C. (1996). *Windows on Mathematical Meanings: Learning Cultures and Computers*. London: Kluwer Academic Publishers.

Papert, S. (1982). *Mindstorms: Children, Computers and Powerful Ideas*. London: Harvester Press.

Pratt, D. & Noss, R. (2002). The micro-evolution of mathematical knowledge: The case of randomness*, Journal of the Learning Sciences,* 11*(*4*)*, 453-488*.*

Reason *[*Def. noun 1, 1.2, 2, verb 1, 1.1*]. (*n.d.*).* In *Oxford Dictionary Online.* Retrieved April 7, 2015, from http://www.oxforddictionaries.com/definition/english/reason

Reid, D*. (*2014*).* The coherence of enactivism and mathematics education research: a case study. *Avant, 5(2)*, 137-172.

Rohlf, M. (2014), Immanuel Kant, In E. Zalta (Ed.) *The Stanford Encyclopedia of Philosophy*. Retrieved September 22, 2015 from http://plato.stanford.edu/archives/sum2014/entries/kant/

Roth, W-M*. (*2011*). Passibility: at the limits of the constructivist metaphor.* Dordrecht, the Netherlands*:* Springer*.*

Sellars, W*. (*1963*).* Empiricism and the philosophy of mind, in W. Sellars (Ed.) *Science, Perception and Reality* (pp. 127-196), London: Routledge & Kegan Paul.

von Glasersfeld, E. (*1990*). "*An exposition of constructivism*: *why some like it radical*". *Journal for Research In Mathematics Education – Monograph* ***4****, 19–29.*

Restructuration in Practice: Challenging a Pop-Culture Evolutionary Theory through Agent Based Modeling

Ümit Aslan, *umitaslan@u.northwestern.edu*
Learning Sciences and Center for Connected Learning, Northwestern University

Uri Wilensky, *uri@northwestern.edu*
Learning Sciences, Comp. Sci., and Center for Connected Learning, Northwestern University

Abstract

An ordinary, non-scientist person's exposure to science through pop-culture is ever growing. However, science is still believed to have a high threshold of entry. For most people, doing science means going through rigorous training of literature, specialization, and learning formal mathematics. Many people feel that they have no means to figure out which idea is scientifically accurate and which one is inaccurate. More importantly, they only have the trustworthiness of the source as heuristic for believing or not believing in a particular scientific message. In this paper, we argue that agent-based modeling has the potential to lower the threshold of entry to science and to empower people against the flood of scientific messages in pop-culture. To demonstrate our theory in practice, we conduct a thought experiment in which we extract a much debated scientific theory on the evolution of sexes from a BBC Earth documentary and show how one can easily recreate and explore the assumptions of such a theory using the NetLogo agent-based modeling environment. Then, we compare traditional formal mathematics based scientific analysis approach with our agent-based modeling approach and show how the latter affords people with little or no training in high level formal mathematics or evolutionary biology literature to challenge scientific ideas.

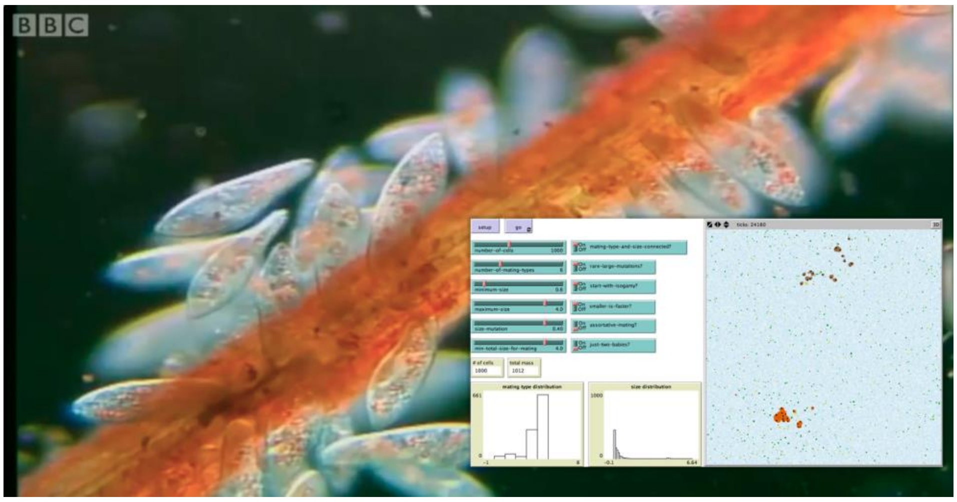


Figure 1. BBC Earth's documentary and the agent based model we developed based on it.

Keywords

Science, literacy, agent-based modeling, restructuration.

Introduction

Research shows that TV and the Internet are by far the main scientific information resources for just plain folks (shortly JPFs), a term coined by Lave (1988) (Brossard and Scheufele, 2013; Horrigan, 2006). JPFs hear about topics like human evolution, depletion of fisheries or the disappearing of honeybees on a daily basis. We begin by asking the following question: How are the JPFs supposed to sift through all the scientific messages that they get through media and examine them? Given that JPFs do not have the time to deeply learn the literature, is there a way to facilitate this process while not becoming an expert in science? Unfortunately, it seems like JPFs currently only leverage heuristics such as trust in scientists or trust in resources when forming opinions about on scientific information, such as believing or not believing in the climate change (Hmielowski, Feldman, Myers, Leiserowitz and Maibach, 2013). This situation shows that JPFs are dramatically underpowered against the scientific enterprise and finding ways to lower the threshold of entry to science is an important challenge for learning scientists. In this paper, we reiterate a constructionist ideal, democratizing science through restructuration (see Wilensky and Papert, 2010), and argue that agent-based modeling has the potential to empower JPFs in their casual interactions with scientific ideas through pop-culture.

So far, a common discourse in the science education literature has been justifying the need for formal science education based on a need for scientific literacy (e.g., Holbrook and Rannikmae, 2007; NRC, 1996). Many researchers argued that school is *the* place for learning scientific practices and preparing for a life dominated by scientific messages. Falk and Dierking argue that this “school first” paradigm is so ubiquitous that most educators or policy makers take it for granted (2010, p. 486). Yet, research has shown that JPFs learn about science mostly out of school (Feder, Shouse, Lewenstein and Bell, 2009). Besides, retention of even basic scientific knowledge learned through formal schooling has always been problematic (e.g. Custers, 2010). Based on such evidence, it becomes increasingly clear that learning about the scientific method and memorizing scientific facts is not enough; facts are easily forgotten and JPFs neither have tools nor time to practice the scientific method. They need better ways to deal with the scientific information thrown at them in their daily lives. We believe that studying JPFs’ out of school interactions with science can inform us in restructuring science education in formal settings. Therefore, we focus on JPFs interactions with science through pop-culture.

Although we agree with the calls for scientific literacy, we believe that we need to rethink its framing. According to Papert, “becoming literate means thinking differently than one did previously” (1993, p. 10). Similarly, diSessa (2001) argues that small changes in learnability can have far greater impact on the eventual impact and theorizes that computers can be *the technical foundation* of a new and enhanced literacy. Lastly, Wilensky and Papert (2010) argue that we need to recalibrate the focus of learning from the means to the object of it. Wilensky and Papert particularly show how agent-based representations are superior to equational representations of science by analyzing both major representational shifts in the history, such as the effects of switching from Roman numerals to Arabic numerals, and strengths of agent-based restructurations of modern scientific phenomena such as kinetic molecular theory and materials science.

Building on these theories, we claim that agent-based modeling has both the power and the potential to play a critical role in lowering the threshold of entry to science and facilitating democratization of scientific knowledge. In addition, we coin the term *agent-based literacy* as a promising approach to rethink JPFs’ everyday interactions with scientific ideas, as well as reframing the very construct of scientific literacy. In particular, we focus on an analysis of how agent-based modeling could empower JPFs to challenge scientific ideas presented in pop-culture. To illustrate our position, we run a thought experiment and demonstrate how a hypothetically agent-based literate JPF can react to a still disputed evolutionary biology theory from a TV documentary and easily challenge it using the NetLogo agent-based modeling environment (Wilensky, 1999) without needing to become an expert in the domain. We also discuss how agent-based modeling differs from the traditional scientific method in challenging the same evolutionary theory in terms of threshold of entry for non-scientist people.

Evolution of Sexes: Challenging a Pop-Culture Theory using Agent-Based Modeling

Let's conduct a thought experiment and explore the learning of an imaginary JFP named Alex, who enjoys following BBC Earth's channel on YouTube. In April 2015, Alex would come across a short documentary titled "Simple sex and the birth of gender" (BBC Earth, 2015), which presents ideas on the evolution of the sexes. Following is the full excerpt of the documentary:

> "The sex is a force to battle because females have a score to settle. And it all started about a billion years ago when simple single celled animals like paramecium rocked the world. Only individuals virtually identical in shape and form but they differ genetically. Only different genetic types can come together and reproduce. They fuse their membranes and exchange some of their cell contents. In this form of simple sex, they part after mating as new individuals. There can be dozens of sexes in these animals, highlighted as different colors, and there are no battles.
>
> But that all changed long ago when one of them *cheated*. A *quirk of evolution* made one of the sexes smaller and this turned out to be one of the most significant events in life on earth. The smaller types were more mobile and so monopolized the matings. But the mates of the little cheats could only produce viable offspring if they grew fat to compensate for their partner's lack of provision. Now one sex was on a path to obesity, the other to athletic swimming. The presence of these sex midgets drove all the other sexes to extinction, except for the huge fatty cells. The small sex, those that cheated, became sperm and the large sex eggs. Gender as we know it was born."

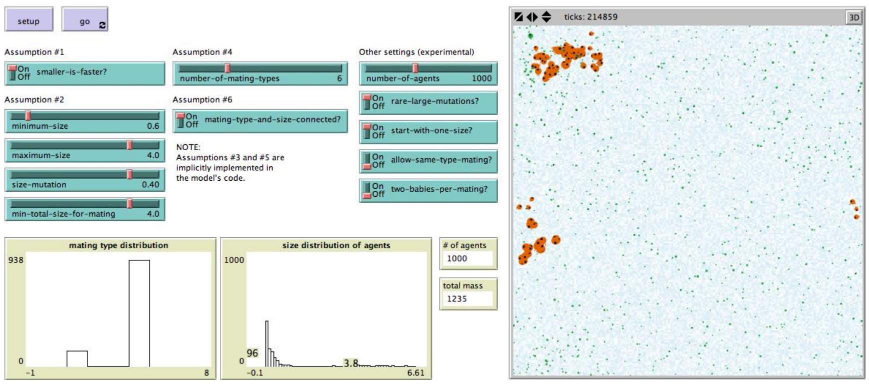
Now let's assume Alex is not literate in agent-based modeling. She would feel this theory compelling and probably believe in it because BBC Earth is a trustworthy news source. However, directly accepting the BBC Earth's theory (BET, in short) could result in Alex developing some misconceptions about the process of evolution. Yes, BET sounds scientifically accurate initially, but some words used by the narrator such as *cheated* and *quirk*, might have led Alex into believing that (1) the process of evolution is one of miracles, and (2) individual agents are consciously trying to gain selective advantage. Conversely, if we assume Alex is literate in agent-based modeling, she would be able to test BET by developing a model in an agent-based modeling environment. In the following section, we describe how one can replicate BET in the NetLogo agent-based modeling environment without any exposure to the evolutionary biology literature.

Modeling BBC Earth's Theory in NetLogo

In order to explore how Alex might go about exploring BET, we went through the process of creating a model that can illustrate how an agent-based literate JFP would approach such a question. Due to the fact that we wanted to think through the lens of a JPF, we did not collaborate with any evolutionary biologist in this process. We developed a simple model of BET and conducted a simple, non-rigorous analysis of the model output. In this section, we begin by explaining our modeling decisions and then describe how the model works.

First step in modeling BET is extracting the assumptions from the documentary. There are six main assumptions of BET: (1) smaller cells move faster, (2) a minimum total size is required for two organisms to reproduce, (3) the evolution of sexes happened among simple single celled organisms, (4) there were more than two mating types initially, (5) mating type is a single gene/trait and it is represented by different colors in the documentary, (6) mating type of an organism and its size are somehow connected. On the other hand, BET gives no explanation on the mechanisms of evolution. Therefore, we have to fill in the blanks with our best guesses. For instance, there is no mention of life cycles of these organisms and the population size so we need to figure out how many agents needed in the system and how would they die. Because we have limited computing power, we decided to have a range of 0-2500 agents in the model. We also decided to have a carrying capacity for the system so we killed agents randomly if the number of agents exceeds a certain number. However, we did not put any natural death mechanism to the system at all. Next, BET assumes that two organisms need a certain amount of material so that their offspring survives but it does not mention what happens if this criterion is not met. Thus, we decided not to allow two organisms to initiate mating if their total size does not exceed a threshold. Third, BET talks about fusion, but it does not say anything about how these organisms are reproducing. In our model, two mating agents hatch new baby agents and then die immediately.

In addition, we wanted to create a model in which the user can change parameters and test different assumptions freely. For some assumptions, we placed an on/off switch (see Figure 2). For example, we wanted to see if speed is really a selective pressure in such a system, so we created a "*smaller-is-faster?*" switch. When this switch is turned on, smaller agents are faster than bigger agents. When this switch is turned off, all agents move with the same speed. Moreover, we did not have any idea on how big or small organisms would be. We also did not know what was the range of size in this organisms, so we had to go with arbitrary values for organism size and size mutations. Still, we wanted to have some sort of tinkering space so we created sliders for each variable such as *minimum-size* and *minimum-total-size-for-mating*.



Assumption #1
On Off smaller-is-faster?

Assumption #2
minimum-size 0.6
maximum-size 4.0
size-mutation 0.40
min-total-size-for-mating 4.0

Assumption #4
number-of-mating-types 6

Assumption #6
On Off mating-type-and-size-connected?

NOTE:
Assumptions #3 and #5 are implicitly implemented in the model's code.

Other settings (experimental)
number-of-agents 1000
On Off rare-large-mutations?
On Off start-with-one-size?
On Off allow-same-type-mating?
On Off two-babies-per-mating?

# of agents 1000
total mass 1235

Figure 2. The evolution of sex model based on BBC Earth's theory.

After some tests, we decided to set the variables as shown in Figure 2 for our demonstrative experiment. In terms of defining agent behavior, each agent in our model represents a single-celled organism and follow a very simple set of commands. In each time tick, each agent changes its direction randomly and moves forward. Then, it checks if there is another agent on the same place with a different mating type (color) and enough size to meet the minimum size requirement for reproducing. If it can find such a partner, it initiates mating with its partner. In the mating process, the mating type and the average size of prospective babies are picked randomly from the parents' trait pool. Next, if *two-babies-per-mating?* switch is turned off, multiple babies are hatched based on total size of parents and the size of prospective babies. Otherwise, parents hatch just 2 baby agents after each mating. Finally, both parents die.

Results

We ran the model from a state where all agents start with the same size and with 4 different setting combinations: (1) smaller is faster and more than two babies can be produced, (2) speed is constant and more than two babies can be produced, (3) smaller is faster and just two babies are produced, and (4) speed is constant and just two babies are produced. At each run, we waited for the population to come to an equilibrium point and took screenshots of the graphs. Due to space limitations, we will not be able to dive deeper into the results of the model. However, we share typical final states of all four combinations in Figure 3:

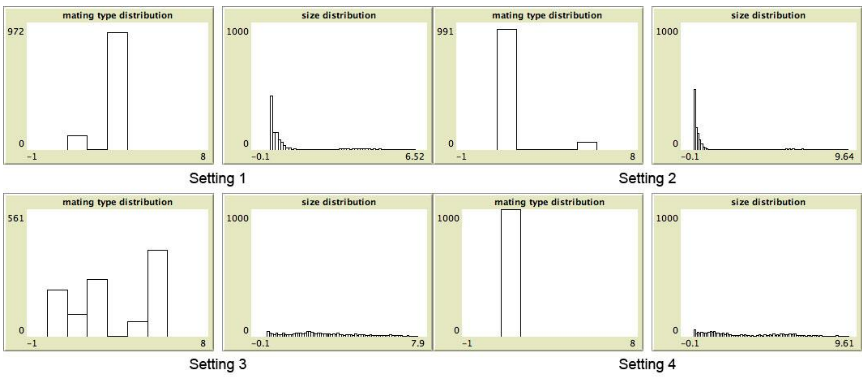


Figure 3. Typical results of four parameter different combinations.

We believe if our plain folk Alex developed such a model, or just even ran it, she would see the emergence of a dichotomous state with numerous extremely small cells and a few very big cells in first two settings. Therefore, she would realize that BET has some explanatory value if the system is setup in this particular way. However, she would also see that there is neither cheating nor a quirk in this process. It all happens through stochasticity and selective pressures in the system. When she took a closer look at the difference between graphs, she would get the chance to test the assumptions of BET, too. Interestingly, there is no substantial difference between smaller cells being faster or all cells being the same speed when we compare the outcome of setting 1 to setting 2. Plus, there is no evolution of small-big dichotomy in settings 3 and 4 at all. It seems like producing more agents in a mating is more important than the speed of individual agents. Particularly, no significant difference between setting 1 and setting 2 shows that the difference is not actually in speeds of the cells. Small agents dominate matings simply because there are many more of them in the system. When this advantage is taken out in settings 3 and 4, big-small dichotomy does not evolve and the model does not end up with a population with two final mating types.

Discussion

Although our model was able to produce a dichotomous state, these findings should not be considered as the proof of how sexes evolved. Instead, our agent-based exploration should be interpreted as a modest exploration of BET and disproof of its assumption about the selective role of agent speed in the hypothetical environment described by it. Likewise, one can rightfully argue that our model is scientifically incorrect because we developed it based on BBC Earth's narration, not on a scientific literature review. However, it still allowed our imaginary JFP Alex to start from somewhere. If Alex had no tools like NetLogo, she would have to first start reading the literature on evolutionary biology. Consider the following excerpt from a paper written by Bulmer and Parker (2002) on this topic:

> "In the ancestral unicellular state the gametic and zygotic survival functions, $g(m)$ and $f(S)$, are likely to be similar in shape and location, leading to isogamy. The development of multicellularity may leave $g(m)$ relatively unchanged, but will push $f(S)$ to the right as the need to provision the zygote increases, eventually leading to anisogamy. This is most clearly seen when the survival functions are sigmoidal, exemplified by the inverse exponential Vance function in equation (2.2). The situation is more complicated in the less likely case when the survival functions are concave, exemplified by the complementary exponential function in equation (2.12)."

Unfortunately, the literature on the evolution of sexes is very challenging for outsiders. In the first place, Alex has to know that this particular phenomenon is called anisogamy, "the occurrence within a population of two gamete types of different size" (Bell, 1978, p. 73), in order to be able to even start a literature review. Then, it is important to realize that there is still no consensus on how anisogamy actually evolved (Blute, 2013). Beyond the literature barriers, Alex also has to know at least the basics of evolutionary game theory (Smith, 1982) to be able to conduct further mathematical analysis in order to challenge BET. Following is another excerpt from Bulmer and Parker's paper (2002, p. 2382):

"We assume that fusions occur between gametes of '+' and '–' individuals (mating types). All + individuals produce n_1 gametes of size m_1, and – individuals produce n_2 gametes of size m_2, with $n_i = M/m_i$, where M is the fixed budget for reproduction. Zygotes are of size $S = m_1 + m_2$, and the survival to adulthood of a zygote of size S is $f(S)$. The chance that a gamete of size m will survive to mate is $g(m)$.

Under these assumptions, the reproductive fitness of + individuals is

$$w_1(m_1, m_2) = \frac{Mg(m_1)}{m_1} f(m_1 + m_2),$$

and the reproductive fitness of – individuals is

$$w_2(m_1, m_2) = \frac{Mg(m_2)}{m_2} f(m_1 + m_2)."$$

This is actually the start of Bulmer and Parker's mathematical analysis and they only talk about how they are going to calculate fitness functions of individuals. Still, even if Alex understands some parts of their paper, it is unfortunately impossible for her to move on from there if she does not have a good background in formal mathematics. Now consider the following code from our agent-based model (gray lines starting with a semicolon are our comments):

```
to maybe-mate
  ; I check if there are any cells here, who is of a different mating type big enough to mate with me
  let potential-partners other cells-here with [
    mating-type != [mating-type] of myself
    and (size + [size] of myself > min-total-size-for-mating)   ]
  ; if I bump into any potential partners
  if any? potential-partners [
    ; I pick one of them randomly and initiate the mating with it
    let my-partner one-of potential-partners
    ; First, we pool our traits so that our offspring inherits the mating type from either parent randomly
    let mating-type-of-babies one-of list ((mating-type) ([mating-type] of my-partner))
    ; Our offspring inherits their average size based on the mating type
    let average-size-of-babies size
    if mating-type-of-babies != my-mating-type [  set average-size-of-babies [size] of partner  ]
    ; based on the total size of me and my partner, the count of baby cells is calculated
```

```
        let number-of-babies ceiling ((my-size + partner-size) / average-size-of-babies) + 1
        ; then we hatch baby agents one by one
        hatch number-of-babies [
          ; each baby inherits the same mating type
          set mating-type mating-type-of-babies
          ; and then mutates its size by adding or subtracting a small random number to the average baby size
          set size average-size-of-babies + random-float size-mutation - random-float size-mutation    ]
        ; after mating, me and my partner die!
        ask my-partner [die]
        die    ]
    end
```

NetLogo code of our agent based model is much more readable and easier to understand compared to mathematical analysis of the evolution of sexes. The primitives of NetLogo such as *one-of*, *hatch* and *die*, require little to no explanation. There is no scientific jargon in our code at all. More importantly, our agent-based model focuses on a concrete process whereas Bulmer and Parker's study focuses on an abstract mathematical analysis. In other words, Alex does not need to be trained to acquire a specific way of thinking.

As a side note, it is worth mentioning that we are not trying to contribute to evolutionary biology literature, at all, while Bulmer and Parker's analysis is built on a very rigorous scientific analysis. Our model, in that respect, is not an effort to do science and it is not comparable to Bulmer and Parker's work. Our goal is to lower the threshold whereas Bulmer and Parker are trying to raise the ceiling. We must also mention that our agent-based code is not a direct equivalent of Bulmer and Parker's equations. The two approaches are distinctly different from each other by nature. In agent based modeling, our focus is on defining the behavior of individual agents and investigating emergent phenomena (Wilensky, 2001), while evolutionary game theory focuses on figuring out strategies that maximize the fitness using equational models (McNamara and Weissing, 2010).

Nevertheless, for our imaginary JFP Alex, who did not study biology formally, challenging BET would have been extremely hard and frustrating if she was agent-based illiterate. She would have no joy in exploring such a domain at all and give up quickly after realizing that she needs much more expert knowledge in order to answer her questions on the theory. However, as an agent based literate person, she can potentially enter the world of evolutionary biology, discover that BET has some flaws, and begin to explore more about the evolution of sexes. It is true that she cannot produce any new scientific theories based on this model, but she can still feel satisfied because this model has potential to empower her against a scientific idea that she is exposed through pop-culture. If she was not literate in agent based modeling, she might have just chosen to believe in the BBC's narrator because it was published by a credible source or she might have rejected it based on her intuitions or superstitions.

Conclusion and Future Directions

In this paper, we proposed agent-based literacy as a way to rethink JPFs' interaction with scientific information in daily life. We argued that people are continuously exposed to a flood of scientific messages through pop-culture but they do not have reliable means to evaluate these ideas other than heuristics about the trustworthiness of resources. The traditional method of challenging these pop-media theories would be diving into scientific literature, which requires understanding heavy jargon and formal mathematical analysis. We argued that agent-based modeling literacy would support a richer participation in scientific practices and conducted a thought experiment based on a pop-culture evolutionary biology theory that we extracted from a documentary. We showed that an agent-based literate person has the power to challenge such a theory and demonstrated how one would develop an agent-based model of this particular theory. Then we compared agent-based modeling with evolutionary game theory analysis.

Based on our current analysis, and also based on decades of constructionist research, we strongly argue that a major restructuration in science is much needed (e.g. Papert, 1993; diSessa, 2001; Wilensky and Papert, 2010; Wilkerson-Jerde and Wilensky, 2010). As part of this restructuration, we need to reconsider the way we conceptualize scientific literacy and teach science in schools. In our opinion, the advent of a computer-based literacy is far more critical than the current operationalization of scientific literacy. We believe that JPFs need more than just heuristics, simple facts, or basic knowledge of the scientific method. We need to lower the threshold of entry to science so that they have better tools and objects to challenge scientific messages thrown at them by entertainers, politicians, or policy makers. In this respect, agent-based modeling is a promising restructuration of science. It is much easier to learn and use compared to the scientific method and complex formal mathematical analyses. It is also much more accessible for JPFs who has neither time to conduct scientific experiments nor access to scientific facilities.

However, we are cautious that we do not propose agent-based modeling as a total replacement for current scientific practices. It would, of course, be absurd to reach in such a conclusion. In addition, we are we are aware of the flaws of our agent-based exploration of the evolution of sexes. In the future studies, we hope to collaborate with evolutionary biologists and develop a better model of anisogamy with correct scientific assumptions and mechanisms. We are also aware of the fact that just one thought experiment has little or no generalizability on the effects of agent-based literacy on lowering the threshold of entry to science. Therefore, our main challenge in future studies is to study the challenges of becoming agent-based literate for JPFs, the potential ways to facilitate this kind of scientific explorations for them, and learning gains of such experiences.

References

BBC Earth. (2015, April 8). Simple sex and the birth of gender - Battle of the Sexes in the Animal World [Video file]. Retrieved from https://www.youtube.com/watch?v=dWa9NmNvChU

Bell, G. (1978). The evolution of anisogamy. Journal of Theoretical Biology,73(2), 247-270.

Blute, M. (2013). The evolution of anisogamy: more questions than answers.Biological Theory, 7(1), 3-9.

Brossard, D., & Scheufele, D. A. (2013). Science, new media, and the public.*Science*, *339*(6115), 40-41.

Bulmer, M. G., & Parker, G. A. (2002). The evolution of anisogamy: a game-theoretic approach. *Proceedings of the Royal Society of London B: Biological Sciences*, *269*(1507), 2381-2388.

Custers, E. J. (2010). Long-term retention of basic science knowledge: a review study. *Advances in Health Sciences Education*, *15*(1), 109-128.

diSessa, A. A. (2001). Changing minds: Computers, learning, and literacy. Mit Press.

Falk, J., & Dierking, L. (2010). School is not where most Americans learn most of their science. *American Scientist*, *98*(6), 486.

Feder, M. A., Shouse, A. W., Lewenstein, B., & Bell, P. (Eds.). (2009). *Learning Science in Informal Environments:: People, Places, and Pursuits*. National Academies Press.

Hmielowski, J. D., Feldman, L., Myers, T. A., Leiserowitz, A., & Maibach, E. (2013). An attack on science? Media use, trust in scientists, and perceptions of global warming. *Public Understanding of Science*, 0963662513480091.

Holbrook, J., & Rannikmae, M. (2007). The nature of science education for enhancing scientific literacy. *International Journal of Science Education*, *29*(11), 1347-1362.

Horrigan, J. B. (2006). *The Internet as a resource for news and information science*. Pew Internet & American Life Project.

Lave, J. (1988). *Cognition in practice: Mind, mathematics and culture in everyday life*. Cambridge University Press.

McNamara, J. M., & Weissing, F. J. (2010). Evolutionary game theory. Social behaviour: Genes, ecology and evolution, 109-33.

National Research Council (Ed.). (1996). *National science education standards*. National Academy Press.

Papert, S. (1993). The children's machine: Rethinking school in the age of the computer. Basic Books.

Smith, J. M. (1982). Evolution and the Theory of Games. Cambridge university press.

Wilensky, U. (1999). {NetLogo}.

Wilensky, U. (2001). Modeling nature's emergent patterns with multi-agent languages. In Proceedings of EuroLogo (pp. 1-6).

Wilensky, U., & Papert, S. (2010). Restructurations: Reformulations of knowledge disciplines through new representational forms. Constructionism.

Wilkerson-Jerde, M. H., & Wilensky, U. (2010). Restructuring change, interpreting changes: The deltatick modeling and analysis toolkit. *Proceedings of constructionism*.

Samba School of the 21st Century: Learning in the Break Dance Community in Bangkok

Nalin Tutiyaphuengprasert, *Nalin@alumni.stanford.edu*
Darunsikkhalai School for Innovative Learning, KMUTT, Thailand

Abstract

International dance culture such as J-pop, K-pop and American street dance spread very quickly among young children and teenagers in Bangkok, Thailand. You can see many dance events two to three times a month. Wandering around the streets in the city of Bangkok, Thailand, at night, you may sometimes encounter a group of young teenagers taking their shirts off and dancing in the dark. Some dances are quite intense as when they put their heads on the floor, flip their bodies upside down and spin on their heads very quickly. People call this group of teens, "street dancers".

The street dancers or so called B-Boys and B-Girls practice intensively 4-5 hours a day, 5-7 days a week. This is comparable to or even greater than the amount of time that these teens would spend in formal school. "Going to school is boring." This is quite a hard fact for some students. But looking at the joy of these dancers in their learning and the amount of time and effort that this group of youth put into those solid 4-5 hours of deliberate practice sparked my curiosity. With inspiration from Papert's "Samba School" (Papert, 1980), I was involved with a dancers' community group in Bangkok as a novice practitioner and participant observer in my ethnography study since 2013.

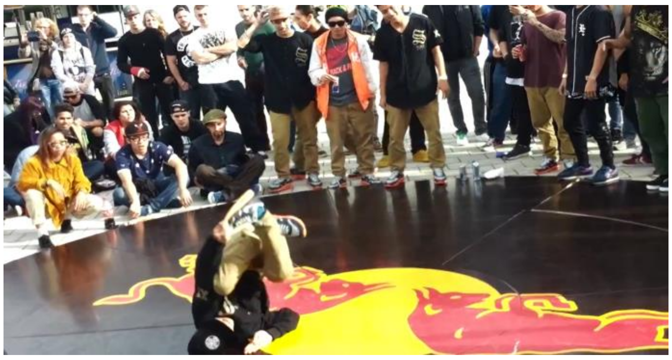


Figure 1. Break Dancer Learning Community in Bangkok

The objective of this study is to observe the mechanics of learning in this dance community. I have also analyzed one case study in depth from observations and an interview of a dancer who found that Bboy dance experiences transformed his life and helped him improve his academic achievement in school. This study might give us some ideas on how to redesign learning and intervention in order to deliver "Samba School" (Papert, 1980) learning experiences in formal school context.

Keywords: informal learning environment, learning community

Methodology

I was a participant observer in real settings in Bangkok from May 2013 to July 2014. I participated in the night practices in Bangkok where I spent 3-4 hours a day and 3-4 days a week as a novice dancer learning how to do Bboy (Bgirl) dance at three different practice areas which are Manorohm Building on Rama 4 Road, National Sport Stadium in downtown Bangkok and Gateway Department Store on Sukhumvit Road. I also participated as an observer in "dance battles" once a month at Fortune Town Department Store, and two battles of the year, (BOTY 2013-2014) and International Battle of the Year (South East Asia) in September 2013.

There are also four different field notes from observing the video clips of the practices and battles which were shared on Youtube. Those video clips were recorded by dancers but also some tourists who stopped by and watched the practice. There was one video clip created by a TV program as a semi-documentary about a Breakdance event in Bangkok with some dancer interviews in the clips. One semi-structure interview was done in April 2015. It was a two-hour interview via Facebook messenger. The informant typed the answer in Thai and I translated all the conversation into English.

Getting to know "Breakdance"

Street dancer or Breaker

I learned from discussions with different people during the practices that what we were doing is called "Bboy" or "Bgirl". Sometimes people called it "breakdance". They dance to American mid-70's funky music and the electronic music genre. B is actually an abbreviation for "break" which is a part of the word "breakbeat" in music. Breakbeat is a technical term where the composer plays with the rhythm part of the song which involves syncopation and polyrhythm. Breakbeat can be found in turntablist's music where they remix the sound from different songs to create a new texture of sounds. This creates the unpredictable and new sound of the rhythm which allows the "break dancer" to express their dance vividly. (Wikipedia, 2015)

Battle and Practice

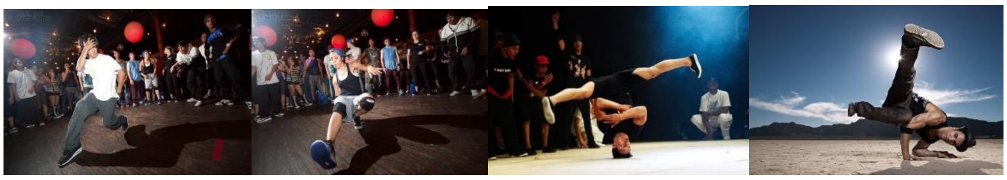
Breakdance can be performed both in a choreographed style or improvisational style. In the battle, break dancers will improvise their dance according to the changes of music or corresponding to the movement of the opponent dancers. In the battle, each dancer has about 30-40 seconds in each round of the dance. One of the battle traditions is that the dancer has to do the same dance movement with an opponent and make it better in order to compete and get a higher score. There were usually 3-5 judges sitting and watching the battle on stage.

> The other dancer came out to show some steps and dance close to the opponent and use the body language saying something like "watch me..". The dancer mimics the other team and tries to make it more fancy. It looked like fighting but it's the dance fight, who is more powerful, cooler, stronger in the dance and design. Each dancer has their own design. They used all parts of the body. Some dancers threw themselves on the floor after jumping in the air. The people giving support will point at the dancer and to the movements that they like. ... The three judges sit on chairs on the stage and watch all the dancers. They have paper in their hands for taking notes. The MC holds the microphone to control the time and clear the stage if the dancers become too intense, in order to avoid fighting. (p.3, field note 2)

Figure 2. Monthly battle in Bangkok (Keep It Real) supported by a department store in Bangkok

In order to be able to smoothly improvise in the battles each dancer must practice a lot to get both perfection and completeness of the movement. They must also practice in order to respond quickly and nicely to the unexpected music. The songs that they use in the battle are usually a mix of different songs and each song will lasts only 25-40 seconds. Sometimes the mood and tone of the song can be changed in the middle of the dance and the dancer must respond quickly to the new song and redesign the performance accordingly.

The breakdance basic movements can be divided into four categories which are Top-rocking (stand posture dance), Footwork (dance by sitting down on the floor), Power moves (body flip upside down and spin movement) and Freezes (Body flip upside down and freeze postures). Dancers will compose all these movements into the 30-40-second set which usually starts with Top-rocking and complete the whole set of movements respectively, finishing with the Freeze.



Top-rocking *Footwork* *Power move* *Freeze*

Figure 3. Four basic break dancing moves

For practicing, dancers in Bangkok usually gather after work in a particular spot that they find convenient for practice. It can be either outdoors or indoors. The dancers look for a wide open space where the floor is smooth so they can spin on the floor. The dance practices are sometimes in abandoned spaces in old or brand new department stores, in the open space in front of the building or at the open space at the Bangkok sky train station.

Figure 4. Practice at National Stadium in Bangkok (Left) and practice in front of Maneeya Building by Rama 4 Road (Right)

The battle in Bangkok is usually organized once a month. In the battle, dancers will compete in solo, duos or in bigger groups of teams of up to 10 dancers. During the battle, dancers often take turns supporting each other. This allows each member to support another with their different strengths.

In the battle venue, dancers dress up and get ready for long hours of dancing. Cypher is a favorite activity for many dancers during the battle day. Cypher is a name given to a circle(s) of practice which will be in the battle venue. One battle may have more than one cypher ring in the area. People gather at the cypher to warm up and stretch before the battle.

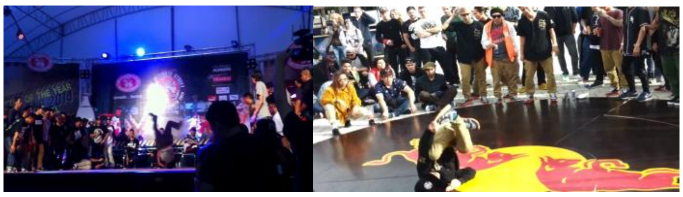


Figure 5. BOTY 2014 Battle Stage(Left) and Cypher Space (Right)

In general, dancers prepare themselves not just for the battle but also for getting ready for the long hours of dance in cypher(s) as well. The battle event usually starts around 4 pm. and finishes by 9 pm. During those five hours, the dancers take turns getting on stage for battle when they are called and will join the cypher(s) when they are not in the battle. The cypher is considered the more forgiving area for mistakes and practice and it is the space where both novices and experts get together and share the dance and ideas and support each other. The cypher stage is active most of the time until the end of the battle event.

Analysis

Challenge is a powerful drive not an obstacle for learning.

Challenge is one of the common drives among many breakers as stated in the interviews. Some Bboys start off with skateboarding or playing other kinds of sports and at least once had seen another Bboy dance during a practice. Body flip and spinning on the head was one of the common movements that triggered many dancers to start their Bboy experience.

MC: Could you guys tell me your inspiration when you first got into bboy dancing?

Dancer A: I started Skateboarding, I scratched skateboard and saw them spin head. I like that..then can you teach me?(The dancer speaks in phrases and words not in a complete sentence. I interpreted that it started off with skateboarding. He liked skateboard but then saw people spin on their heads and he liked it. So he asked them to teach him.)

Dancer B: I like to challenge myself. I think the other people can do it and I like this thing and I think I can do it, too. I can do it and continue doing this. Have fun with the music.

Dancer C: It is self-proofing.

Dancer D: I would like to share with other kids who like bboying, heart came first. when your heart really feel like it, you put your heart into it and then you do it. Keep doing your best no matter how people may scold or criticize you, you guys keep doing it continuously. Heart must come first. (p.2, Field note 4)

Dawes and Larson (2011) discussed "Developing a Sense of Competence" which is a type of personal connection that drives youth to put so much effort and become highly engaged in the activity. Dancer A, B, C and D illustrated such traits among the dancers in that they enjoyed facing challenges and take them as a "self-proofing" process. Dawes and Larson(2011) explained that youth "Doing well in program activities—and having that acknowledged by others—provided meaningful self-affirmation" (p.264). Bboy activities, both in the practice and battles, continuously provide acknowledgement (waving their hands to show appreciation, supporting each other during the practice and battles, etc.) Moreover the dancer himself/herself continuously achieves small goals in the dance practice. The joy of small accomplishments in daily practice empowers the dancers to put more effort into the practice and develop confidence in themselves over time.

Low maintenance learning but high performance aims through practice

Break dance culture in Bangkok is casual. People meet up and practice at free random spaces such as a supermarket, in front of a building, in the abandoned space of the shopping malls, etc. They gather casually after work or school. You can come and leave at any time. There is not much investment or expense in the practice. Basically, dancers can just grab a bottle of water, dress comfortably and be there. Dancers or the novice practioner can join the practice without any fee or experience expected.

This space is inside the supermarket. There were people passing by and some stopped and watched the dance. Sometimes dancers will sit in groups and play on their phones and chat. Some dancers sit and watch the dance and do stretching at the same time. There was not much conversation among those who were dancing. This dance consumes a lot of energy. Most dancers have very strong arms and bodies. None of them use alcoholic drinks. There were some water bottles by the side of the rings. The floor is not clean but these dancers didn't care. They just laid down or sat on the floor very comfortably. (p.2, Field note 1)

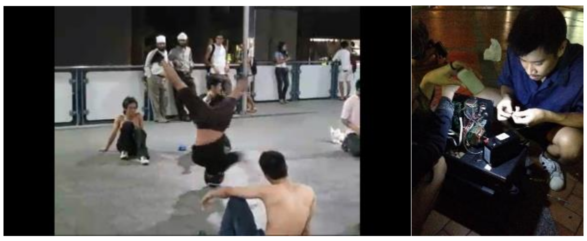


Figure 6. Practice at the Bangkok Sky Train Station. (Left) and the cheap speaker that we used to play the music from our phones. We are trying to fix our old speaker in one of those practice nights. (Right)

The simple and low-cost resources that I saw in the practice such as not needing special dress or materials, the group use of a small cheap speaker, and spaces that are free and open to the public, etc. made it easy to get started. The dancers valued the hard work in deliberate practice as the key to success rather than paid attention to external factors. All these factors created a low barrier to entry that encourages dancers or anyone to start and practice once you have the interest, with the least amount of investment. As one of the dancers said "Heart must come first" and the rest is just putting your effort into the practice.

Community of practitioners as a rich pool of expertise

In the battle or practice, you can see the novices and experts mingle and dance together. They share and learn from each other. Dancers come to practice with a goal in their minds and they also learn from watching and observing other dancers.

> Informant: There was nobody teaching me step by step. I practice this and that all by myself, it's more like sharing knowledge informally rather than real teaching. The more I dance, the more I find it even more interesting, it's hard to explain
>
> Interviewer: What do you think about novice and experienced dancers practicing in the same place? Were there any rules or ways you guys practiced together?
>
> Informant: Not really. We kind of know what we should do. People who came to practice, they will have their own goal of the day. Did you? In general, it' s more like laid back.
> Interviewer: Do the experts learn from novices sometimes? I'm curious.
>
> Informant: Yes, it's more like sharing knowledge. (Interview transcription)

Knowledge-sharing is one of the common interactions in both practice and in the battle between novice and expert. This is also reflected in Lave and Wenger's descriptions of (1991) "near-peer in circulation of knowledge". Learning in this community starts off with creating a relationship with each other and in the open and friendly community create an effective and powerful exchange of knowledge. Everyone in the community can be a learning resource for other learners. In terms of Hull and Greeno's (2006) "Context of Identity", this community has plenty of "figured worlds" where you can see and learn from different examples of many experienced dancers. This is considered a rich pool of expert resources that you can hang around and learn from. As a novice dancer, I was once practicing the basic steps of footwork with five other mentors watching me and correcting my movement in detail.

Dynamic of achievement and equity to success

Sometimes, as a novice dancer, I tried to compose my own dance movements and many experienced dancers watched me and some of them told me that they liked it and iterated my movements further. Papert (1980) stated that in the "samba school" the novice is not separated from the expert, and the experts are also learning. (p.179) This idea still resonates in this 21st century dance community. Learning and sharing in this community is without the status of expertise. Novice dancers with inexperienced minds sometimes come up with fresh ideas and new dances unknowingly. It is common to see a novice dancer paired up or on the same team with an experienced dancer or even in battle with each other. Sometimes the less experienced dancers win in the battle. This creates a huge impact on the novice dancer and empowers them to work hard and practice because there is always a chance to achieve regardless, of how little experience you have.

> This year I won in one of the legend battles in Thailand. I was so lucky. I won the battle in February, 2015. It's called One Man Standing Battle. By the battle's name you know that the person who gets into the last battle is the winner. The rule was all the bboys will be randomly picked to battle one on one and be eliminated until we get to the last one. I felt so good to win this battle. This is my first battle that I win and it's a solo dance. I felt so happy about it. All the winners in the past were very experienced and senior dancers such as Cheno, Tekky, Priwan and Visa (all the bboy's names who are famous in Thailand and all of them have experience in international battles.) I was shocked and could not

> speak on that day. I smiled and I was so stunned by winning this legendary battle of Thailand. This is my glory in my dance experience! This is like the reward for myself and all dance experiences in the past." (Interview transcription)

The culture that values hard work rather than background experience encourages the newcomers to put more effort into practice. The winner is sometimes the less experienced dancer, this depends on who works (practices) more and performs well on the stage.

Know more about others and then yourself.

The Breakdance community in Bangkok has a high level of diversity. There are dancers from other countries joining in battles and in daily practice. There are some dancers who move to other countries just because they would like to try a new practice experience. There are many dancers from European countries such as England, Spain, Turkey and Switzerland in our dance practice group. Occasionally, there are some dancers from other Asian countries such as Japan and Korea joining our practice. There was a Japanese dancer who I talked to who told me that he comes to Thailand to buy food materials for his Thai restaurant in Japan once every two to three months. He takes this opportunity to practice with the dance crews at the National Stadium. There are Spanish dancers who can speak Thai just because they come to Thailand very often to practice. These are experienced dancers who sometimes help arrange the battle and give opportunities for the Thai dancers who win to fly to join battles in Spain.

> "Some foreigners told me that practice in Thailand is more fun though. They said practice with Thai people is much more fun. In other countries, they are quite serious about the practice but I think that's why they are so good at it." (Interview transcription)

This creates an opportunity for dancers to be exposed to international cultures and languages. This supports and encourages dancers to learn more about languages and different cultures and life styles. One of the dancers in the interview told me that socializing in this community changed his perspective about the world and helped him to develop himself to be more mature and get better at English and in his overall academic performance.

> Informant: I was like the average kids in high school who just go to school, get bored, I played a lot. I don't like studying at all. It was during grade 7-9 when I was addicted to games. I went home and played games until I slept, just like other kids, you know. It looked so bad when I looked back. I still feel guilty about it. I am growing up a lot. Dance helps me to learn from the society, from socializing with different people. My world is getting much wider and helps me think about my life and I want to make it better.
>
> Informant: I got much better grades. I'm not very good at language but I can communicate in English better. My GPA is increasing from 2.50 to 3.20 now. Many things are getting better. Now I am studying in IT department, information system. (Interview transcription)

This environment created complex learning opportunities for this dancer. The language barrier became an opportunity for him to put a high level of effort into practicing English in order to strengthen his experiences as well as develop social networking. As Hull and Greeno (2006) stated about language, that literacy practices are "diverse in function and form and purpose". (p. 81). For this circumstance, English and other languages are needed to help dancers accomplish the goal of learning some new dance techniques from each other and sometimes to work together as a team in the battle. Moreover, dance as a self-expression art could help learners to learn more about themselves. Designing your own dance and practice gets the dancers to actually explore and build their identity throughout the process.

> "Informant: I have been changing a lot! I will put it simple like this. Considering, if we live our life normally, your life is just normal but if we try something new, we will have an extraordinary life. I think it's more challenging life. Consider change, I feel like I grew up in another community. The community that all have their own attitude about life, very independent. Dance is like you can express your identity. You can dance at any time during the day. Dance helps me find freedom." (Interview transcription)

Discussion

We can see from the evidence in this study that what learners required to help them learn well is not "easiness" of content. Learners, instead, require challenging tasks regardless of how hard the content or practice might be. The hard tasks on the other hand create pride and confidence for learners in the end. To transfer this powerful street dance learning mechanism to conventional classrooms, I would like to suggest some ideas to restructure learning and assess strategies in the classroom.

In learning, the design of classroom activity should focus on developing challenging task as the first priority. The key success factor is to present topics in more interesting ways to "students". Teacher's knowledge of students' interest would be a good start to meaningfully and tactfully connect a topic of learning to them. For example, dancers develop their language skills in order to communicate to new friends who can teach them how to dance. One right trigger for many learners can motivate them to learn for years. We can see that many dancers who were not successful in school, were triggered by something else which is "very hard" outside of schools. They spent a lot of time and effort into "hard practicing". They were dedicated students but only when they found something that interested them and could meaningfully connect with them. Teachers could spend more time designing "creative challenges" for particular contents to help students learn by themselves rather than planning how to teach content directly to them in the first place.

In the 21st century, digital technology provides us with a variety of learning resources including almost every kind of media available. This creates advantages where learners can access almost limitless content at their level and in their learning styles. Many dancers learn basic movement by themselves from video clips and they can connect to experts in the field by sending online messages at any time. Consider this natural learning interaction these days, how teachers now can let loose the old structure of conventional content knowledge transferring in classroom. Networking becomes an important skills for students to acquire knowledge in new learning culture. Hence, allow space and time for networking among students inside and outside of classroom can help facilitate students to learn how to connect and use of new learning amenities.

Restructuring the assessment mindset and how it should function in the classroom would also be another interesting aspect. Dancers grow up with intense competition, however all those competitions are done in a very creative manner. Studying the way that battles and cyphers coexist in the same environment may give us ideas on how to create such intense but forgiving evaluating atmospheres in the same place. Friendly and entertaining assessments, which do not compromise on quality, help dancers to be ambitious and to develop themselves continuously. In the classroom, assessment is a tool to help teachers keep track of students' progress. But formative assessment can be done in the form of fun competitions or sharing sessions instead of in a formal testing atmosphere. However, the teacher's skill in orchestrating a good balance between competition and collaboration is another key factor where further investigation is needed.

While the "Samba School" of the 1970s has many of the characteristics of the "Street Dance Community" of the 21st century, the core value of learning quality in the informal learning environment is still maintained. They both provide learning experiences where all learners in the community can find joy in learning and developing themselves continuously. Furthermore, considering the modern learning technology environment and a better understanding of informal learning, it is possible to have better opportunities to turn "phobias" or a negative self-identity into a "mania" for deep learning and creating new positive identities. This preliminary study of the street dancers community in Bangkok gives us a big picture of an alternative social and learning interaction in a unique learning environment. However, it also gives rise to more need for further observation and study, both in scale and depth.

References

Dawes, N. P., & Larson, R. (2011). How youth get engaged: grounded-theory research on motivational development in organized youth programs. *Developmental psychology*, *47*(1), 259.

HULL, G. A., & GREENO, J. G. (2006). IDENTITY AND. *Learning in places: The informal education reader*, *249*, 77.

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*. Cambridge university press.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc..

B-Boying. (n.d.). Retrieved June 1, 2011 from Wikipedia, the free encyclopedia

http://en.wikipedia.org/wiki/B-boying

Summer League: Supporting FLL Competition

Pavel Petrovič, *ppetrovic@acm.org*
Dept of Applied Informatics, Comenius University, Bratislava

Richard Balogh, *balogh@elf.stuba.sk*
Institute of Automotive Mechatronics, Slovak University of Technology, Bratislava

Abstract

In the years 2013-2015, we organized three years of a competition in building and programming robots in Slovakia named Letná liga FLL (Summer League of FLL). We invented and designed the competition format. All the tasks are our original ones. The competition has a unique format allowing the teams to compete remotely, eliminating all travel costs. It drops the requirement of time-demanding preparations lasting many weeks that is present in most common robotics competitions, such as RoboCup Junior, FIRST® LEGO® League, and World Robot Olympiad. However, this competition stimulates an exceptional level of creativity and provides an early and manifold feedback in a repetitive fashion. All solutions of the teams are published after the deadline, each being unique and special. After extensive, but well-motivated work on every task, each team compares their own solution with a plethora of other ways of thinking about the same problem. In this way, children learn from each other on a great scale. We experience a series of surprises and unexpected unforeseen approaches to the tasks when evaluating the various solutions after each round. The feedback from the participants has been positive and the dedication of many is beyond our early expectations. This format stimulates a regular and goal-oriented work in the after-school robotics clubs. An additional valuable outcome of our work is the set of 30 creative activities focused on various aspects such as robot design, robot control, and programming, use of sensors, navigation, manipulation, physics, art, and other. Each task has multiple and inspiring solutions. Activities can be solved in 1-3 club meetings. Tasks, solutions and evaluations are freely available on-line.

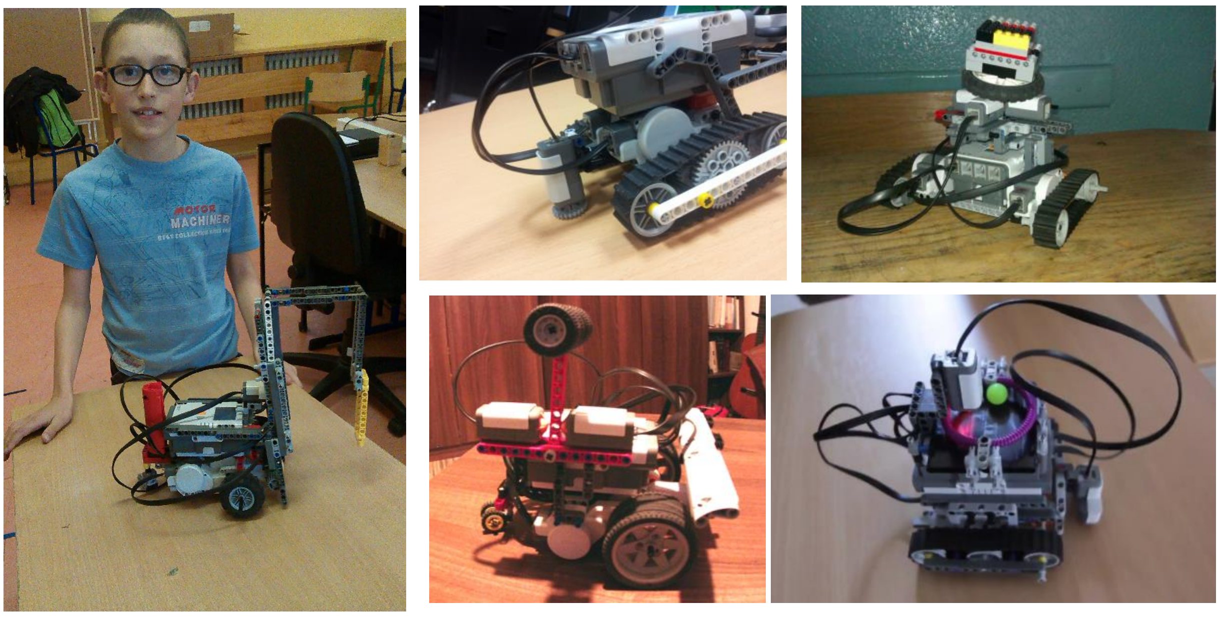


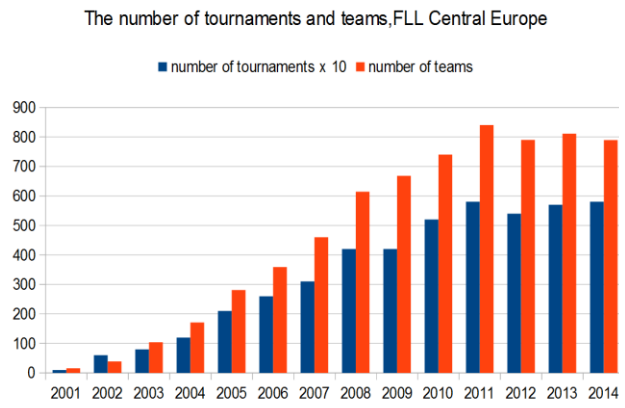
Figure 1: Different solutions of the skier task: from top left: teams Tobias, Amazing Team, Too lazy, LeGorazda, and Gamčabot.

Keywords

robotics competition, constructionism, after-school activities

Introduction

During an approximately 20 years of existence of robotics competitions for elementary and secondary schools (Petrovič & Balogh, 2008, Petrovič, 2011, Petrovič, Balogh & Lúčny 2010), basic types of competitions have crystalized. Some of them are successful in Slovakia too. For instance FIRST® LEGO® League in the Central European region has saturated from the point of view of the number of teams (Figure 2 left), source: Hands-on-Technology, Leipzig. On the contrary, Slovakia still holds a potential for a development (Figure 2 right), source: fll.sk. Even though Slovakia performs better than other counties in the V4 group in this measure, its development is hindered by the financial situation of the schools and the developing regions.



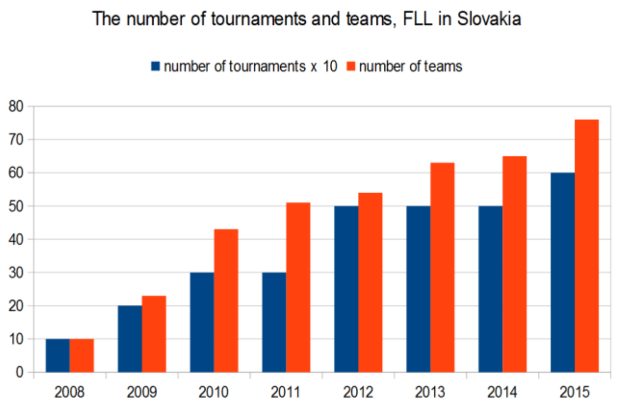


Figure 2: Saturation of the number of tournaments and teams, FLL Central Europe (left) and a potential for a further growth, FLL in Slovakia (R).

A classification of robot contests should take into account at least the following factors:

- *Required time for preparation*: how many weeks, months, or years of work are required to participate successfully?
- *Dependence on external technical support and consultations*: are the average participants able to join the challenges without having experts in their circles? Is the help from the experts needed to help them with constructing and programming the robots?
- *Recidivism*: is the challenge the same – or very similar year after year? Are the teams that already participated in the challenge for several consecutive years already in a particular advantage compared to newcomers?
- *Creativity degree*: are the typical solutions to the task notoriously known? one of the few expected designs with several small modifications? Or alternately, almost every solution is a surprise? In the first case, a steady performance improvement can be observed from year to year. The alternate approach underlines grasping the problem in a creative, original and elegant manner.
- *Team-work focus*: some of the competitions focus on the technical solution only, they are strictly focused on robotics. Other competitions attempt to have a balanced approach, combining teamwork skills, searching and presenting information, solving problems.
- *Narrow focus*: is there a single objective in the competition? Alternately, does the challenge consist of a set of relatively independent little tasks?
- *Platform standardization*: at one end of this axis, the teams are required to use a standardized platform for all aspects of their solution, which is useful to develop their ability to improvise and work within a given set of constraints. The other end drops all constraints on creativity, use of materials, and technologically more suitable solutions.

- *Competing teams' expenses*: how much do the teams need to invest into: registration, material and equipment, tools, software, sets, travel cost, food and accommodation?
- *Organizers expenses*: what are the typical costs that require fund raising and/or sponsor support on the side of the organizers? How difficult it is to organize an event? How many volunteers are needed, how much planning and management in advance is required?

FIRST® LEGO® League is by far the largest and the most popular robotics contest for young people. Among other advantages, its strengths lie in 1) *a common platform* – all children use the same software and hardware platform, which is available and easy to access. It means all participants work with the same set of constraints, which is fair; 2) *a new set of tasks every year* – which allows and motivates newcomer teams to be successful – an important advantage over RoboCup Junior leagues, for instance; 3) *explicitly stimulates team-work* learning and experiences by its format. Team-work is essential in the contemporary world, however often very neglected in school systems. It is an important factor in school reforms in many countries, see for instance (Bjorke et al 2015). More research and didactic work must be done in this direction. FLL provides a suitable research platform on this topic; 4) avoids over-focusing on technology by combining the Robot Game with *Research Project* – making it easier to setup interdisciplinary and gender-mixed teams, plus it builds an entry path to the technological fields for girls. Moreover, its research project provides an excellent platform for learning about how to search, process, and present information (Oppliger, 2002, Timcenko, 2011); 5) a relatively short and well defined *intensive work period* – during this time, teams are able to dedicate extra efforts, produce, and focus. They learn to be goal-oriented. However, large part of the year still remains FLL-free and allows for recovery, motivation-gain for the next season, and leaves space for some other activities as we will show in this paper.

As a regular jury members, organizers, and occasional mentors at the regional FLL contests, we saw a lot of unused potential after the contest finishes. Many teams that are involved in the competition stop their work on robots and programming until the next season. Then they start again from the beginning. We understand the need to work continuously and to keep learning, improving existing skills, gaining new knowledge during the quiet time between the seasons. During the competition period, teams are focused on the specific task. Teams seldom find an additional time for learning new things. They need to play with the robots, make experiments, learn how to work with sensors, find new programming methods, and discover new mechanical concepts also outside the regular competition. In an ideal world, team leaders use this time for learning and improving the skills. In real, coaches don't know what to do with the team in the meantime and lack ideas of what to do more, often dropping the meetings completely. This is the idea behind the Summer League. To give the teams a chance to learn new things, give them additional challenges and some space for more experiments.

Summer League

Organization

In the first year of the SL, we started in May 2013. We published a new task every week, however, the teams had time two weeks to solve it, thus they could already see the specification of another round while they were working on the previous one, and think in advance, let their ideas to develop over time. Alternately, they could divide into dedicated groups, each solving one of the available tasks. We did not advertise the competition in any particular way, but six teams still joined in, and brought interesting solutions. We continued during summer holidays with somewhat longer period for solutions until we reached 10 rounds, but there were very few contestants after the school year was over. We manually extracted all the submitted solutions from e-mails and made them available on a dedicated website, which was part of the homepage of FLL in Slovakia. The main idea was to support the FLL competition, although our initiative was completely independent from FLL. One of the authors is responsible for localization of FLL competition to Slovak language (rules, task descriptions) and maintains the website, while the other author is a member of an association that supports FLL by many different ways.

The second year, we started after the start of our summer semester, in March, and still provided a new task approximately every week, leaving longer time during holiday seasons (spring holidays, Easter, May). In this way, we managed to reach our goal to finish before the long summer holidays. This time however, teams had longer time to submit the solution – usually around three weeks, and thus there always was time of about three weeks to work on each of them. Our idea was to give the teams a selection of tasks to work on all the time, allowing them to skip some of the tasks, or better put, to select those they like or prefer for any particular reason. The number of participating teams tripled in the second year, in total there were 18 teams participating, while five of them kept very active until the tenth round, six additional teams tried at least three different tasks, and the remaining teams participated once or two times only. We were very happy about the increased interest and tried to accommodate the feedback from the teams into the third year – most importantly, decrease the frequency to allow for better quality solutions in longer time.

In the third year, we started even a month earlier. A new task was published every 14 days. Teams could use 3 weeks to work on each task. The total number of participating teams raised to 24, but nine of them solved at least 9 out of 10 rounds, seven additional teams solved at least half of the tasks, and four additional teams solved at least three rounds. In total we have received and evaluated 140 unique solutions (on average 14 per round), however, not the quantity, but the endeavor, and the quality of the solutions was the most satisfying. Both years 2014 and 2015 were organized with the support of a specialized web application implemented using a CodeIgniter MVC framework. It was designed and implemented by a group of students under the supervision of the first author to fulfill their group-project duties in our course on Development of Information Systems (a sort of introduction to Software Engineering).

About the Rules

The contest is run in a friendly, open and trustful spirit, the main focus is to have fun while learning something new. In the first year, there were no particular prizes. We sent the teams a signed and stamped participation certificates and one solution was awarded with a little bag with spare plastic parts and some sweets every week. In the third year, we tried to motivate the teams, and arranged with the association Robotika.SK to sponsor the contest with one LEGO MINDSTORMS robotics set to the overall winner with the highest score after all ten rounds. The rules were formulated as follows:

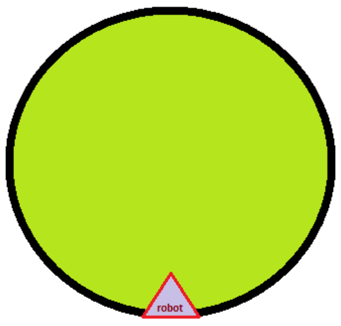
1. competing teams consist of participants in the age of 10-16 years;
2. teams do not need to register for FLL, we are open to other teams as well;
3. in the school, in the club, or at home, team solves the task, and submits their solution to our on-line system;
4. each solution should contain: photograph of the team, one or more pictures of the robot, program, a video showing how the robot solves the task (link to an unlisted YouTube video), and a description of the solution;
5. the hardware must be LEGO MINDSTORMS (as in FLL);
6. the software must be NXT-G, EV3, or Robolab (as in FLL);
7. each solution is awarded by 0-3 points that sum up to the overall score;
8. solutions are evaluated by an independent group of judges (in practice we use 3 or 4 judges and use the average of their evaluations);
9. every team is allowed to make changes to the task specification – to simplify it, if they find it too difficult, earning somewhat reduced score.

The submitted solutions are hidden until the very time of the task deadline when they automatically become visible to all other teams. In this way, the participants receive a very early feedback, and compare their best idea on how to solve the task with all other teams – watch their videos, which is often a lot of fun, and this motivates them to try to understand how other teams were thinking, and what design choices and tricks they used in their programs. Since the teams have already been thinking and working with the task for more than a week, there is a large substance of matter in their mind to connect with this new information, which facilitates their learning in the best possible ways, building on the very principles of constructionism: discovering and learning about the world by doing, and sharing the results with others. We find this contest to be a manifestation and a unique case-study of how constructionism principles extend in an on-line context.

Example Tasks

In this section, we present seven different tasks, each focused on a subset of competences.

1. **Golf (round 4, year 1)**. In this task, the robot is placed on a circular golf green area and its job is to sprinkle it with the fertilizer, represented by tiny plastic parts. The circle is surrounded by a black line, and the fertilizer must be spread evenly, and cover the whole area systematically. The teams must find a mechanical solution to slowly distribute many little parts over time, while the robot successfully navigates of the area. Even though the task focus is on mechanics of the spreading machine, teams had to pay attention to successfully navigate throughout the green, usually using the color or light sensor. Thus a successful solution requires the capability to process and react to input sensory information.



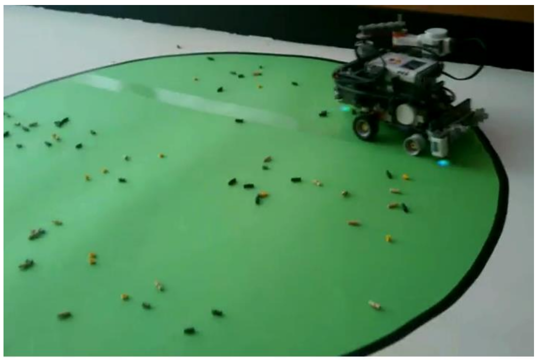


Figure 3: Situation at the start of the task Golf (left) and an example solution of RJMTeam (right).

2. **Rossie has worries (round 6, year 1)**. A family has a personal robot Rossie and leaves for a holiday. The flowers must be watered. In this case, a real water and real flower pots should be used, and thus the task is an example of connecting the learned knowledge to everyday life situation in the real world. Teams must solve two main challenges: localizing and navigating towards the flower pots, operating a watering mechanism – for instance by holding a cup, and tilting it to pour the water. Please see Figure 4 for an example solution.

3. **Skiing robot (round 1, year 2)**. Robot "skier" is placed in the middle of a skiing slope represented by a tilted board in a random orientation. It does not know whether it is facing uphill or downhill or in any arbitrary direction. It may not use accelerometers, gyroscopes or inclinometers. The task is to turn until the robot is facing up, climb the hill to the top, turn around, and "ski" all the way down. This task stimulates thinking out of the box. The teams must realize that what they need is to detect the direction of the gravity, and then find some way how to do that. Figure 1 shows different solutions of five teams: Team Tobias installed a loose hanging plate about 15cm in front of an ultrasonic distance sensor. While turning, the plate moves nearer or farther of the sensor. The moment it is closest to the sensor is the moment robot is facing uphill. Solution of Amazing Team has a simple touch sensor at its back, which is pressed by the robot's own gravity due to a special design of the caterpillars. Teams "Too lazy" and LeGorazda used a weight on top of the robot that tilts in one of the directions, pressing one of the attached touch sensors – the one corresponding to the instant inclination of the robot. Finally, the most impressive solution of Gamčabot team uses a ball rotating around a circle on top of the robot as it changes its tilt in various directions. A light sensor detects the wished direction.

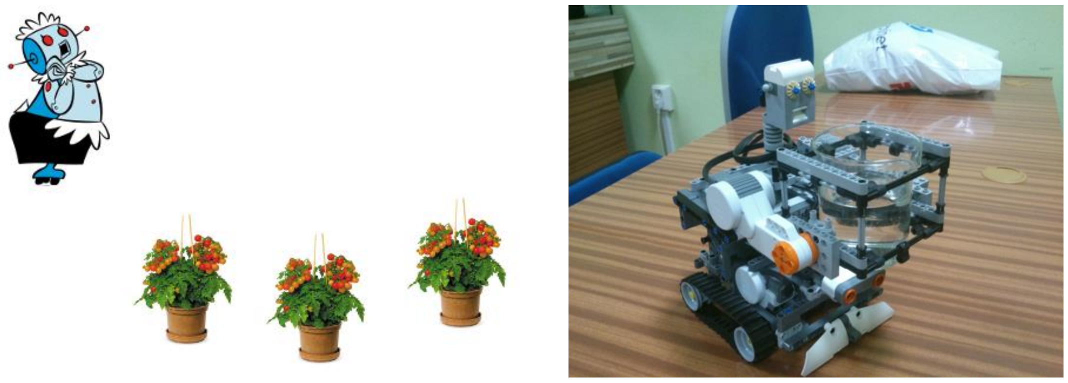


Figure 4: Situation at the start of Rossie has worries task (L), and a solution of RJMTeam (R).

4. **Safe car overtaking (round 6, year 2)** is an example of a motivating task for less advanced learners – they learn the interaction with the environment utilizing the light sensor to detect solid or dashed line, ultrasonic or infrared distance sensor in a closed loop with the motors to control the speed of the robot and a proper navigation to successfully take over the slower car.

5. **Secret hiding place (round 7, year 2)** is an advanced task that requires the robot to memorize a pattern, and to classify further patterns shown as valid or not. The teams built their secret treasures that can only be open using a correct card. Cards contain various patterns of white and black stripes (see figure 6 left). The robot must be able to learn a new pattern that can then repeatedly open the treasure. Incorrect patterns must be rejected and the treasure remains closed. Some teams simplified the task and detected the black/white sensor readings into 4 different variables. The team Benders (figure 6 right) used 16 different positions and encoded the card pattern using binary notation into a single 16-bit integer. The most advanced solution used timers to measure the widths of the black and white stripes. The widths were recorded to a file. The stored and read pattern were compared. If their difference summed under certain allowed error threshold, the opening card has been detected.

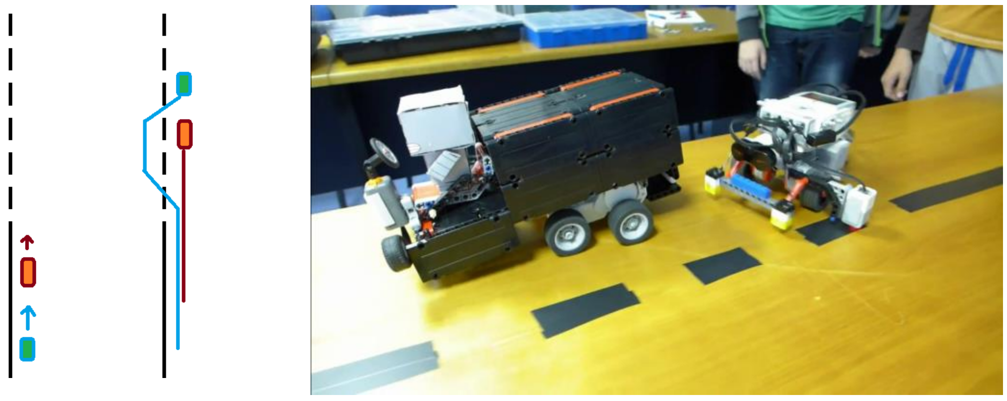


Figure 5: Safe car overtaking task, situation (L), an example solution of Benders Team (R).

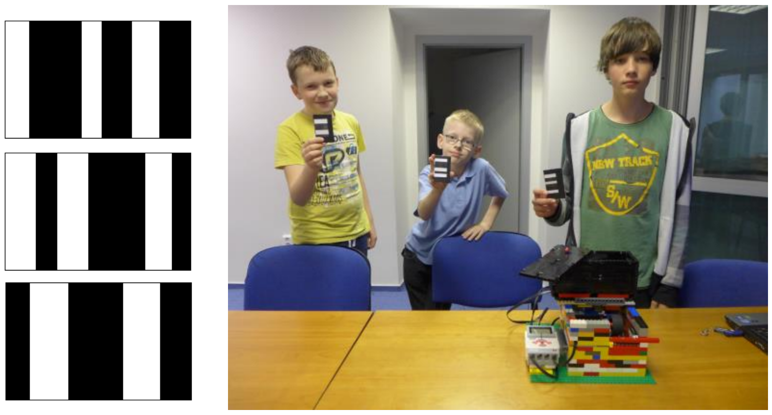


Figure 6: Various card patterns (L) and the best secret treasure – team Benders (R).

6. **Swing (round 1, year 3)**. Robot is sitting on a hanging swinging seat, its task is to start swinging, and then, when it enjoys swinging, to stop. In addition to the design challenge, the interesting part of this task is the changing time pattern of the required movements depending on how much swinging is already taking place. Some teams used distance sensor to detect the swing-position of the robot. Team Ladybirds (figure 7 right) approached this creative challenge with files – they have made a robot that can be trained to swing using a touch sensor. It records the correct timings and then performs the learned sequence. In fact, this task would be suitable even for students of control theory at university.

7. **Art (round 5, year 3)**. In the middle of the season, one task is usually dedicated to a creative artistic topic. In the second year, we focused on music, while in the third year on fine arts. Teams have made: beautiful T-shirt that was painted by a robot, 3D drilling machine, various artistic plotters, 3D printer with LEGO Bricks, Easter eggs painting, and even machine that can draw with glowing light in the dark, see figure 8. We found solutions to this task inspiring.

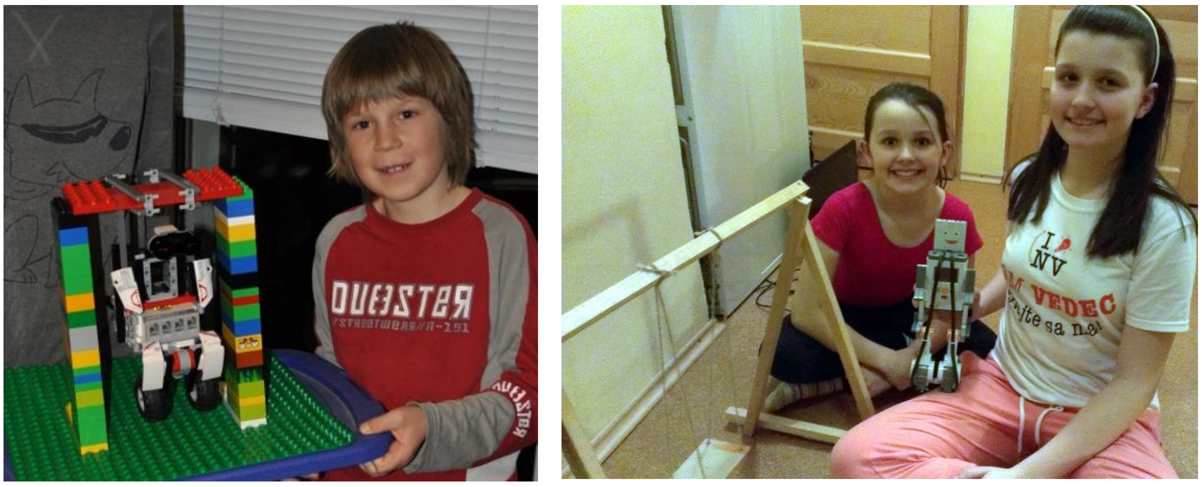


Figure 7: Two swinging robots (teams Dunajská Lužná and LadyBirds).

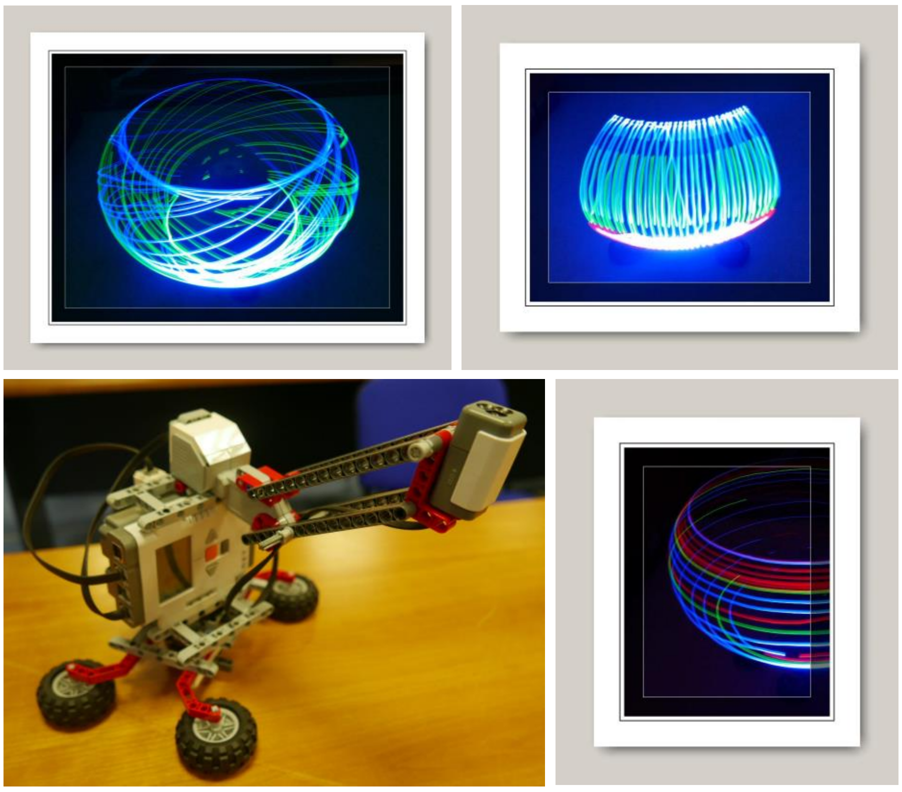


Figure 8: Drawing in the dark – team Benders.

Supporting Software

In the first year, we have manually published the submitted solutions on-line, which even for 6 teams turned out to be a too demanding and time-consuming task. Therefore, a student project at the faculty of the first author has been assigned. The students have successfully created a web application in a usual LAMP environment using CodeIgniter MVC framework. The application allows the team leaders to register a new team at any time, view all passed and current task specification, submit their description, pictures, attachments, video links, and edit their solution before the deadline. Their solutions are nicely formatted and integrated into the whole website of FLL in Slovakia that runs under a version of MODx CMS. Judges can view the solutions, assign scores, write their textual evaluation, select the best team in each round, and make their evaluation public. The resulting scores are computed automatically and shown in a table on the website. Organizers can enter their task descriptions with pictures and maintain a pool of tasks to be used later in the contest, providing a starting and ending dates when the task description becomes visible and when no more new solutions are accepted. This system has simplified the organizers work and made it more convenient for the teams to adjust the presentation of their solution as they wish. Another group of talented students is currently working on a complete rewrite of the web application, this time allowing for multiple languages and better user interface to prepare the system for our future goals.

Discussion

Should the rules be limited in the same way as the original FLL competition rules? It seems to open creativity doors when we allow to use additional components, not strictly limited to LEGO parts. Also programming experience can be wider when not stuck on the graphical programming environment only. On the other side, wider base here can later be limiting when teams start to prepare for regular FLL competition. Children can easily forget some important limitations (this sensor was allowed in SL, so why we can't use it now?). Is there an "optimal" time interval for solving the tasks? Are three weeks sufficient? Usually the clubs meetings are organized once a week, so they may need more time to discuss and solve the task.

Conclusions

We have organized three years of creative constructionist robotics on-line competition for young people in Slovakia aged 10-16, with 10 tasks each year. Its main aim is to prepare the pupils for the FIRST® LEGO® League competition, but it proved to have even larger impact. We have experienced a growing interest and a positive feedback from the participants. We are opening the competition to a global international participation. Participation in the contest is free and easy, emphasizes learning, and creativity, it does not include stress and frustrations from failures at tournament day – which are inevitable in on-site single-day events. The amount of sharing ideas that takes place in this contest is among its strongest advantages. We efficiently utilize modern media – such as YouTube videos and a dedicated web application that allows the participants, referees, and organizers to maintain the contest automatically on their own without any other assistance from a system administrator.

References

Bjorke, S.Å., Lazareva, A., Mayende, G.and Nampijja, D. (2015) Together we can. Team and online collaborative work, PULS; http://grimstad.uia.no/puls/Groupwork/main.htm, accessed 11 May 15.

Petrovič, P., Balogh, R. (2008) *Educational Robotics Initiatives in Slovakia*. In Proceedings of Teaching with Robotics Workshop, SIMPAR 2008, Venice, Italy.

Petrovič, P. (2011) *Ten years of Creative Robotics Contests.* ISSEP 2011, Bratislava.

Petrovič, P. Balogh, R., Lúčny A. (2010), *Robotika.SK Approach to Educational Robotics from Elementary Schools to Universities*, In Proceedints of the 1st International Conference on Robotics in Education RIE2010, September 2010, Bratislava. Jones, A. B. and Smith, W. (1984) *Statistical Methods for Scientists*. Wiley, New York.

Oppliger, D. (2002) *Using ® LEGO League to enhance engineering education and to increase the pool of future engineering students (work in progress)*, Frontiers in Education, 2002. FIE 2002. 32nd Annual , vol.3, no., pp.S4D-11,S4D-15 vol.3, 6-9 Nov. 2002.

Timcenko, O., Friesel, A. (2011) *Competition-motivated teamwork and narratives to motivate girls for engineering,* EAEEIE Annual Conference (EAEEIE), 2011 Proceedings of the 22nd , vol., no., pp.1,6, 13-15 June 2011.

Teachers' Constructionist and Deconstructionist Learning by Creating Bebras Tasks

Valentina Dagienė, *valentina.dagiene@mii.vu.lt*
Vilnius University Institute of Mathematics and Informatics, Lithuania

Gerald Futschek, *gerald.futschek@tuwien.ac.at*
Vienna University of Technology, Institute of Software Technology & Interactive Systems, Austria

Gabrielė Stupurienė, *gabriele.stupuriene@mii.vu.lt*
Vilnius University Institute of Mathematics and Informatics, Lithuania

Abstract

The constructionist theory of learning offers useful ways of thinking how computers and digital tools can be included into education. Deconstructionism as an extension of constructionism is a natural way of children' s behaviour while exploring new tools. Constructionism and deconstructionism can be applied both in the classroom and in outreach school activities. To support constructionism and/or deconstructionism, we need to focus on supporting teachers. This paper explores the main question: How teachers may learn in a constructionist and deconstructionist way. We discuss this question from the perspective of the Bebras challenge in informatics and computational thinking. A contest and supplementary activities have been performed for more than ten years by many countries. We focus on one side of the challenge – the development of tasks. We present our experiences from several years of Bebras teachers' workshops. In this paper, we provide two task development stories focussing on teachers' constructionist and deconstructionist learning.

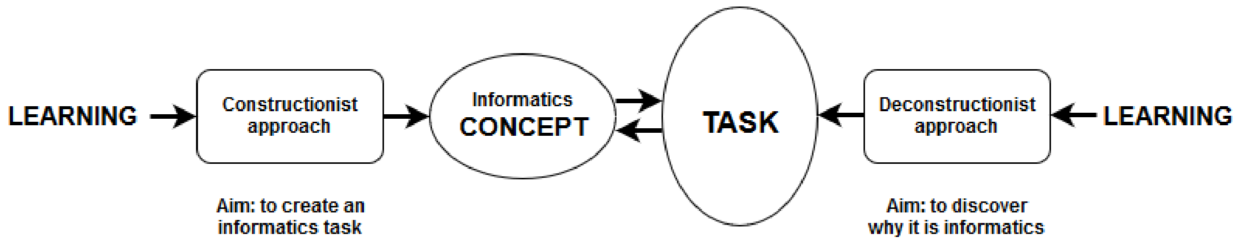

LEARNING
Constructionist approach
Informatics CONCEPT
TASK
Deconstructionist approach
LEARNING
Aim: to create an informatics task
Aim: to discover why it is informatics

Figure 1. Constructionist and deconstructionist teachers' learning by developing tasks

The task is the output, the informatics concept is usually the learning goal and task content, the constructionist and deconstructionist ways are the learning models that are applied to accomplish the task (Figure 1).

Teachers could learn in both ways: 1) through developing (constructing) tasks, and 2) through analysing them solving and explaining the essence of tasks why it is informatics. Furthermore, the presented stories allow us to argue that development of tasks gives teachers an opportunity to learn informatics concepts deeper. Both presented stories demonstrate the constructionist and deconstructionist approaches and focus on the teachers' learning process.

Keywords

Constructionism, deconstructionism, gamification, informatics education, task development, teachers' learning.

Introduction

Seymour Papert extended constructivism learning theory in a way that applies to practical construction and named it by constructionism (Papert, 1987). During several decades the ideas of constructionism have been applied to different activities in education and the results are promising (Brennan&Resnick, 2013; Bruckman, 2006; Resnick, 2014; Rogoff, 1994).

Constructivism advocates learner-centred (in our case the learners are teachers) discovery learning where learners use information they already know to acquire more knowledge (Aleksandrini&Larson, 2002).

Constructionism (Papert&Harel, 1991) and constructionist learning is inspired by the constructivist theory specifying, how individual learners construct mental models in order to understand the world around them. Constructionism provides us the basic idea of an appropriate learning object. Such an object should support learner's step-by-step understanding of the materials and concepts it represents, allowing a user to self-construct his or her knowledge.

Constructionism is focused on the personal construction of ideas and relations through the construction of real-life artefacts. Special attention is paid to using computers and technology (Ben-Ari, 2001). A new approach – deconstructionism as a complementary phase of constructionism was introduced by Pavel Boytchev (2015). Deconstructionism is focused on the personal understanding of ideas and relations of the artefacts. This is important for learning informatics (computer science) concepts. When students cannot relate a new concept to their previous knowledge, they actually fail to decompose that new knowledge.

To support constructionism and deconstructionism in the learning process, we need to focus on supporting teachers, who serve as the agents of classroom innovations (Borko, 2004). How can we support teachers to engage in constructionism? How can they start to think beyond a technocentric view?

We recognise that development of tasks for the Bebras contest as an activity for promoting learning informatics (computer science) have been done using both phases: a constructionist approach and a deconstructionist approach. A constructionist approach is clearly defined by the task development procedure: from the informatics concept or phenomenon to an interesting story with pictures and/or interaction. A deconstructionist approach is based on task analyses: identifying and understanding the structure, content, and concepts hidden in the task. The task analysis identifies the actions and cognitive processes required for a learner to solve a task or achieve a particular goal. Many teachers' task developing workshops, held in Austria and Lithuania as well as at international events (Greece, Kenya, Poland, Slovenia), have been organised using both these approaches.

The history of the Bebras contest-challenge in informatics began in Lithuania on September 25, 2004, when an experimental trial was held. Its aim was to check selected technologies of the contest and to evaluate the level of complexity of the presented tasks. After a month, on October 21, the first Bebras contest took place in Lithuania (Dagiene, 2006).

The crucial point of the contest is good questions (tasks): they focus mainly on informatics concepts; they are short, attractive (expressed by short stories with pictures and sometimes with animation), and answerable in a few minutes.

The Challenge aim and structure

The Bebras International Challenge in Informatics and Computational Thinking (Bebras, 2015) is a motivation challenge in informatics that addresses all school students divided into age groups: Little Beavers (age 8-10), Benjamins (age 10-12), Cadets (age 12-14), Juniors (age 14-16), and Seniors (for upper secondary level). By using a computer pupils have to solve from 15 to 18 tasks of different levels within 40-45 minutes. Thus, each task takes between 1 and 4 minutes to be solved. So far three general types of tasks have been used: (1) interactive tasks, (2) open ended tasks, and (3) multiple-choice tasks (with four choices).

The aim of the Bebras challenge is to promote children's interest in informatics from the very beginning of school and lead them to develop computational thinking abilities. Actually, the main idea is to involve students into informatics task solving activities and to use computational thinking and modern technologies more intensively and creatively (Dagiene&Futschek, 2013).

The Bebras challenge focuses on understanding informatics concepts and phenomena. Understanding and handling the basics and foundations of informatics or computing is more important than knowing many technical details. The use and interpretation of results comes prior to being able to prove them. Controlling computations, calculations, and estimations is more significant than being able to do computations by oneself.

Of course there is a need to learn the basic computer managing techniques very well, but computers have to be understood at many levels, namely: as a fundamental culture and not as a collection of buttons and instructions; as a development of ideas not a finished work; and as an explanation of concepts.

The Bebras challenge should encourage students starting from very early age, kindergarten, to understand computers deeper and to use modern technologies in their learning activities more creatively.

Cognitive, social, cultural and cross-cultural aspects are very important while using technologies – the challenge puts a strong emphasis on culture and language (Fischer, 2009). The Bebras challenge helps the educational community to figure out school students who can use information technologies in a very creative and profound way.

The Bebras challenge is based on solving tasks. A Bebras task is a piece of question on some informatics concept or phenomenon, presented in a short, playful way with a unique story, pictures and/or interactivity.

It is clear that students' (and teachers') interest in participation essentially depends on the task quality. Of course, advertisement of the challenge, belonging to the Bebras international network as well as activeness of schools and teachers also play an important role. Nevertheless tasks remain the crucial issue, especially if we wish not only to motivate students to learn informatics and develop their computational thinking, but also to reveal them a wide range of matters in informatics and to bring out the topicalities.

Tasks should capture the attention of teachers so that they could get interested and encourage students to participate. It is important that teachers would be stimulated to discuss the tasks with learners when the contest is over. Tasks could be part of informatics lessons for introducing new concepts or making easier to understand hard topics.

The Bebras tasks are divided into three levels of complexity. The tasks are being assessed by the following schema: the easiest level of complexity (in each age group) - by 6 points, the medium – by 9 points, and the hardest level – by 12 points. The correct answer adds as many points as indicated for the task, and the incorrect one – subtracts some points (*i.e.* −2, −3, and −4 points, respectively), unanswered problems – 0 points. To avoid negative results, each participant must start having the amount of points equal to the maximum number of points that may be subtracted (*e.g.* 54 points, if there are 18 tasks provided).

Developing of the Bebras Tasks

Attraction, invention, tricks, and surprise should be desirable features of each task presented to students. The Bebras tasks have to be created carefully, taking into account the different aspects of each task, i.e. what educational power and informatics concepts it contains and how to interpret its attractiveness to students (whether it stimulates the motivation of learning).

Since the very first Bebras challenge task developers were searching for more exciting task formulations and distinguishing particular task groups. Consultations on the topics to be involved were continually held. Some of the principles on the structure of tasks and the selection of topics were discussed in research papers (Dagiene&Futschek, 2008; Opmanis, Dagienė&Truu, 2006).

The selection of topics for tasks is related to teaching of the basics of informatics, therefore common elements of education in informatics in all countries (at least which are more active at the international level) are taken into consideration when developing tasks.

A task developing process (spiral cycle) begins with a chosen informatics concept, which is the key idea what we want to teach the students. By adding a funny and interesting story, we create a text with the visual components. By using gamification (application of game principles in non-game contexts), and by adding interactive components (dragging, dropping, etc.) we create a task for the Bebras challenge. The task developing process is based on this spiral methodology (Fig. 2). A Bebras task is usually modified several times, simplified in text, better explained and presented or changed in its story or the question is changed and sometimes even the type of task is changed as well (interactive, multiple choice, open-ended).

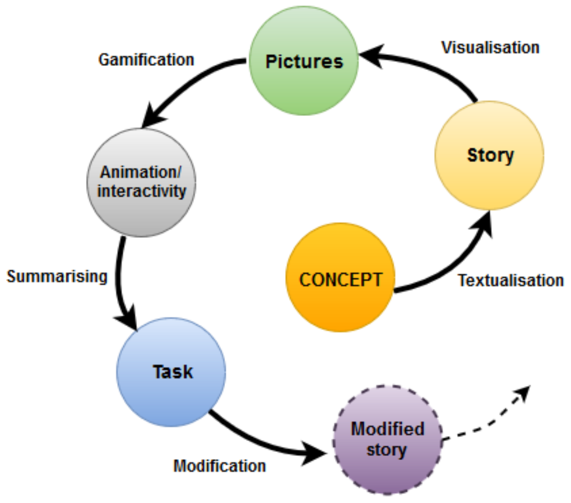


Figure 2. Task developing process

In this paper, we provide two stories with a detailed explanation of the development of tasks that constitute effort to show how teachers can learn informatics concepts in a constructivist way. The aim of this effort was to provide a learning model to empower informatics learning by developing tasks. Developing the Bebras tasks has more than ten-year history, involving more than 40 countries and many teachers as well as researchers.

At the Bebras challenge, tasks are an important source for introducing kids to informatics concepts and procedures. The most important goal of the Bebras challenge is to present informatics concepts in an understandable way and an attractive format so that everybody could learn these concepts and would like to learn informatics.

Tasks are highly valued at the Bebras challenge and their development has been used by the authors in teacher training courses and workshops. Task development workshops were used as a means to help teachers be involved in teaching informatics and support them in reflecting on and revising what informatics is.

Many teachers are involved in creating new Bebras tasks. Each of about 40 countries prepares several tasks that are submitted to the international task workshop. At the workshop some of these tasks are further developed and recommended for use in national Bebras challenges, others are rejected.

During the preparation of tasks for the international workshop, depending on countries' groups of teachers, university classes or local workshops are involved to create tasks. Usually only a selection of these tasks is submitted to the international workshop. At the international workshops (e.g. Dagiene&Futschek, 2012), more than hundred teachers were involved in a further development of tasks. So, a large amount of teachers is involved in task creation. The task development process is constantly improving, while the involved teachers are learning by doing how to create better and better tasks.

While creating Bebras tasks teachers can learn:

- what essential informatics concepts can be conveyed,
- how to present good informatics tasks,
- how to create good stories for informatics tasks,
- how to describe the relation of tasks to informatics concepts,
- how to integrate the Bebras challenge in their teaching activities.

The creation of tasks is a constructive way of learning. Teachers have the freedom to create any task that is useful for the Bebras challenge. In creating tasks a deconstructionist way of learning also takes place: an informatics concept is analysed and deconstructed in its main aspects, some of these aspects are chosen for the task creation where these aspects are constructed to a suitable task. Very often a suitable story has to be invented that enables us to convey the aspects of the informatics concept in an easy way.

Such learning outcomes are very important for the Bebras challenge, but the learned is also transferable to daily teaching at schools. Good tasks are the core of good teaching. So the knowledge about tasks and skills in creating tasks is also valuable for teaching in general.

While the goal of tasks is student's learning, the teachers learn a lot by creating the tasks. The teacher's learning process is more of a constructionist style than the student's learning by solving the task. However, more and more interactive tasks are created by the teachers that allow some aspects of student's constructionist learning.

First example of task development: From a binary operation to a funny story

Binary numbers and binary operations are main informatics concepts that students should learn. So the original idea was to produce a task about the binary XOR (exclusive OR) operation: it has two inputs and one output, it is like the ADD operation which takes two arguments (two inputs) and produces one result (one output).

The formal definition: the binary XOR operation will always produce 1 output if exactly one of its two inputs is 1 and will produce 0 output if both its inputs have identical values (0 or 1).

The teachers started to make a story about this operation that would be attractive to kids. It should be an operation similar to XOR with operands that are from daily life, so that the binary operation is easily understandable. Since XOR is the basic operation in binary adding, they thought of adding something

They had the idea of using tinted glass. Some pieces of glass are either clear or lightly tinted. When one square of glass is placed on top of the other, the result is a double piece that is either clear, lightly tinted or darkly tinted, as shown in Figure 3.

Then they started to construct a question: Assume we have two panes consisting of square glass windows, as shown in Figure 4. We stack them on each other. What colour can we see when looking through the corresponding squares? (This story was presented by J. P. Pretti, Canada).

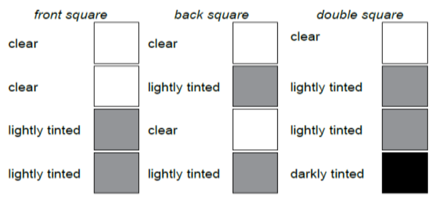


Figure 3. Results of combination of glasses

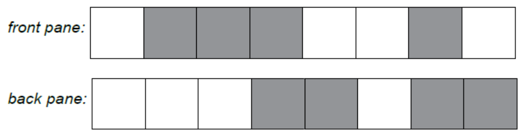


Figure 4. The front and back panes

During the international workshop the task was discussed by several teachers. They thought that the task was still too "technical". They have discovered a real-life story that makes the technical stuff easy to comprehend and changed the story totally (due to J. Vanicek, Czech Republic).

Boat glasses are either clear or lightly tinted. When looking through two such glasses, one will see either clear, lightly tinted, or darkly tinted glass, as shown below.

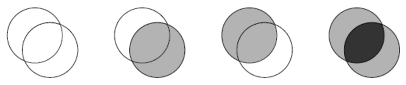


Figure 5. The front and back panes

Captain Black installs circular windows with either clear or lightly tinted glasses into the lower deck of his new yacht, as shown below. When standing at appropriate places on the land, one can see through two corresponding windows on the opposite sides of the yacht. What colours does one see when looking through the corresponding windows?

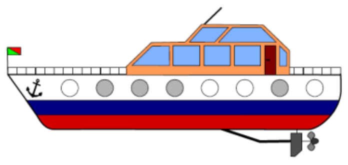


Figure 6. Left side of the yacht

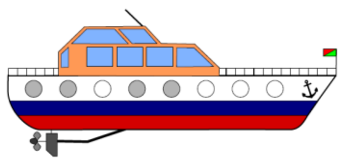


Figure 7. Right side of the yacht

The final version of this task was made interactive, so that the students have just to click on the windows of the resulting ship to change the type of its glasses.

The pictures are now nice and the panes of glasses are embedded in a natural story. We would like to point out that the corresponding glasses in Figure 6 and 7 are in reverse order due to the natural view on the boat. To emphasize that, the anchor in front of the ship was invented in an intermediate version of the task, where just the windows were presented.

An important question in assessing the quality of a task is how well the task is related to informatics concepts. It is a usual habit that during the task workshop a section "It is informatics" is written by teachers for each developed task. So this task has to be analysed and its informatics aspects discussed. Also, the potential solving procedures have to be analysed and broken into its stages (a deconstructionist way as well).

For this task, a further question was raised: is this task still connected to the binary XOR operation? The answer is yes: if we equate clear and darkly tinted glass to 0 and lightly tinted glass to 1, then the above operation corresponds to the XOR operator.

The task has even more power: if we equate clear glass to 0 and any tinted glass (both light and dark tinted) to 1, then the above operation corresponds to the OR operator. And if we relate the dark tinted glass to 0 with one carry bit then we have the relation to adding binary bits. But, anyway, to solve this task, a sequence of binary operations has to be performed with correctly associated pairs of operands (the windows).

Second example of task development: Placing luggage on a conveyor belt

Teachers decided to create a scheduling task. First, they had to decompose the scheduling concept into its essential aspects. Scheduling is an algorithm by which processes are assigned to (limited) resources to complete the work. Scheduling is fundamental to computation itself, and an intrinsic part of the execution model of a computer system; the concept of scheduling makes it possible to have computer multitasking with a single central processing unit.

A scheduler may aim at a specific goal, for example, maximizing the total amount of work completed per time unit, minimizing the response time or minimizing latency. In real-time environments the scheduler must also ensure that processes can meet deadlines.

The teachers started from a simple structure – a sequence of squares. This structure represents a limited resource.

In the first story, letters A, B, C, D, E needed to be allocated in the squares by the rule: put the next letter in the third next empty place, if the end is reached, count continuously from the beginning.

A D B C E

The task is not attractive: neither connected to reality nor making fun. Teachers started to think about various stories. Finally, the airport luggage conveyor belt was proposed and considered.

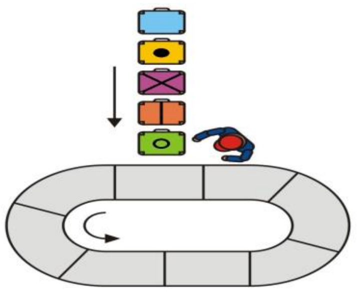


Figure 8. The moving luggage belt

A lazy airport porter is loading the passengers' bags on the moving luggage belt. He always puts the next bag on the third next empty place until all five bags are placed on the luggage belt. How does the luggage belt appear at the end of his work?

(The story was proposed by Stefan Mannsbart, Austria).

In writing the section "It is Informatics", the teachers have to relate the moving luggage belt to realistic informatics applications. Scheduling mechanisms are much more complex in practice as in the given task, where this way to fill the luggage belt is not the most efficient one. Scheduling problems often occur, e.g. the operating system of a computer must perform several tasks quasi-simultaneously on a single processor. The scheduling mechanism then may assign around chunks of computing power to each task or program.

While developing interesting tasks teachers are learning various things: they need to understand the essence of the involved informatics concepts very well, to recognise them in various situations, to choose a good story, and to check whether other people understand the story in a correct way. This process results in a better understanding of their thinking and in the improvement of their expression.

Changes in Task Development

The learning of task creating teachers is also reflected by the change of the applied task developing procedures and techniques. Bebras tasks have been created since 2004 until now. At the beginning of the Bebras initiative the tasks were mainly started from scratch during the yearly international task creation workshop. This procedure was not very time-efficient and some tasks did not achieve the expected quality. To improve that, the tasks were next prepared by the participating countries before the workshop and at the workshop they were selected and further developed. Since this fact works best, if the tasks submitted to the international workshop are already of a high quality, a pre-workshop review process has been installed recently. These changes in task development procedures are a clear sign of progress in teacher's learning of task creation.

Also, the change of type of the tasks, used in the Bebras challenge, shows a progress of learning of teachers. While at the beginning, most of the tasks were of the multiple choice type, now more and more interactive and open-ended tasks are produced and selected. In France, for example, none other than interactive tasks are used now.

Conclusion

The Bebras, an international challenge in informatics and computational thinking, is based on the expression of informatics concepts in attractive, interesting, and funny tasks. Teachers are encouraged to create the Bebras tasks: construct and deconstruct them.

We presented here the task developing process, how teachers learn to better understand certain informatics concepts through their involvement in creating tasks. Their efforts to develop interesting task stories based on some informatics concepts, resulted in a deep understanding of the concept under study and better abilities in explaining the concepts of informatics (what teachers really need).

Creating (constructing) tasks has the same importance as deconstructing the given task – to find out what concepts are hidden in the task and to make a conceptual bridge to informatics as science. Teachers should be involved in both activities - constructionist and deconstructionist approaches for solving tasks - allowing and supporting them to use the experience and to reflect their practice.

References

Alesandrini, K., & Larson, L. (2002). Teachers bridge to constructivism. *The Clearing House*, 75(3), 118-121.

Bebras - International Challenge on Informatics and Computational Thinking (2015). Internet website: http://www.bebras.org

Ben-Ari, M. (2001). Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching*, 20(1), 45-73.

Borko, H. (2004). Professional development and teacher learning: Mapping the terrain. *Educational Researcher*, 33(8), 3–15.

Boytchev, P. (2015). Constructionism and Deconstructionism. *Constructivist Foundations*, 10(3), 355-363.

Brennan, K., & Resnick, M. (2013). Imagining, creating, playing, sharing, reflecting: How online community supports young people as designers of interactive media. In *Emerging technologies for the classroom* (pp. 253-268). Springer New York.

Bruckman, A. (2006) *Learning in online communities*. In: Sawyer K. (Eds.) Cambridge handbook of the learning siences. Cambridge University Press, New York: 461-472.

Dagiene, V., & Futschek, G. (2013). Bebras, a contest to motivate students to study computer science and develop computational thinking. In: *WCCE13*, pp. 139-141.

Dagienė, V. (2006). Information technology contests – introduction to computer science in an attractive way. *Informatics in Education*, 5(1), 37-46.

Dagienė, V., & Futschek, G. (2008). Bebras international contest on informatics and computer literacy: Criteria for good tasks. In *Informatics Education-Supporting Computational Thinking* (pp. 19-30). Springer Berlin Heidelberg.

Dagiene, V., Futschek, G. (2012) Knowledge construction in the Bebras problem solving contest. Conference: Constructionism, Athens, Greece. Available at: http://bebras.org/sites/default/files/documents/publications/Dagiene-2012.pdf

Fischer, G. (2009). End-user development and meta-design: Foundations for cultures of participation. In *End-user development* (pp. 3-14). Springer Berlin Heidelberg.

Opmanis, M., Dagiene, V., & Truu, A. (2006). Task types at "Beaver" contests. *Infomatics Education*, 7-11.

Papert, S. (1987). Computer criticism vs. technocentric thinking. *Educational Researcher*, 16(1), 22-30.

Papert, S. & Harel, I. (1991). Situating constructionism. *Constructionism*, 36, 1-11.

Resnick, M. (2014). Give P's chance: Projects, peers, passion, play. In: Futschek G. & Kynigos C. (Eds.) *Constructionism and creativity*. Proceedings of the Third International Constructionism Conference. Austrian Computer Society, Vienna, (pp. 13-20).

Rogoff, B. (1994). Developing understanding of the idea of communities of learners. *Mind, Culture, and Activity*, 1(4), 209-229.

Teaching computer science teachers: a constructionist approach to professional development on physical computing

Mareen Przybylla[1], Ralf Romeike[2]
[1]University of Potsdam, Didactics of Computer Science
[2]Friedrich-Alexander-Universität Erlangen-Nürnberg, Computing Education Research Group
[1]przybyll@uni-potsdam.de, [2]ralf.romeike@fau.de

Abstract

With the integration of physical computing into computer science curricula in K-12 schools, there is a need for professional development of teachers in this topic area. This article discusses design principles for professional development workshops and reports about practical experience. In particular, the paper describes the development of six principles that were successfully used in planning a workshop on physical computing. The workshop's aim was to empower teachers with wide interests to implement physical computing in computer science classrooms. The workshops' design allowed each teacher to follow his or her personal objectives, to gain hands-on experience in a constructionist learning environment and to reflect on their experience with their colleagues.

Keywords:
physical computing, computing education, professional development, workshop design

1. Introduction

Until recently, software development dominated the creative and design-oriented parts of computer science education. In most schools, the desktop computer is the entry point into the virtual world of computer science. Outside schools, however, traditional desktop computers are increasingly often replaced by ubiquitous computing systems, often containing embedded systems that are hidden inside various intelligent devices (e.g. [1], [2]). The creation of such systems is supported by the large availability of suitable hard- and software tools for all purposes and experience levels, which is also reflected in the maker movement [3]. Makers are engaged in do-it-yourself projects that often include computing skills besides crafting and using electronics. Those activities, where interactive objects are designed and created, can be referred to as 'physical computing'. Interactive Objects are programmed, tangible media that communicate with their environment – be it humans, their surroundings or other interactive objects – through sensors and actuators (cf. [4]). Examples for such interactive objects and installations range from interactive jewellery and clothes over intelligent toy pets to room-filling installation arts. Within the physical computing community tinkering is very popular. This includes two basic activities: exploring existing systems and expressing ideas in creating new systems. This way, constructionist learning [5] takes place: guided by their own interest and for a personally relevant purpose, learners actively construct knowledge. In physical computing activities, students learn with and about interactive computing systems by creating concrete, tangible products of the real world that arise from their own imagination and that they can show around and be proud of in a constructionist sense. Recently, physical computing was integrated into computer science curricula, e.g. England's CAS curriculum [6] or the new computer science curriculum of Berlin/Brandenburg, Germany [7]. This paper therefore examines strategies for introducing computer science teachers to physical computing. After a pilot workshop series, the concept was improved and a two-day professional development workshop was provided for computer science teachers in Germany**.** The aim of this article is to explain the design principles of this workshop, to reflect on the experiences and thus to draw conclusions that can be helpful in planning future professional development workshops with similar foci.

2. Constructionist Learning with Physical Computing

Physical computing has a comparatively long tradition. Blikstein's historical overview of physical computing devices dates back to the 1980s when the LEGO/Logo platform was developed [8]. With physical computing, constructionist learning is raised to a level that enables students to gain haptic experience and thereby concretizes the virtual [9]. Students create real interactive constructions applicable for the purposes of embedded and cyber physical systems and thus learn in authentic contexts [3], [10]. Such learning is described as highly interactive because both, digital media and the real object, immediately reflect learning success and problems and thus allow each learner to learn at his or her own pace based on individual learning goals. Ackermann [11], in reflection of the Piagetian constructivism and Papert's constructionist theory of learning, highlights Papert's view that "'diving into' situations rather than looking at them from a distance, that connectedness rather than separation, are powerful means of gaining understanding". This is exactly what happens in physical computing activities: in designing and realizing their interactive objects, learners dive into the role of inventors. They are connected with their artifacts, even physically, as they can see them, but also touch them, play with them and share them with their peers. Finally, through making their objects, they construct and constantly reconstruct knowledge. As Stager [2] puts it, in physical computing projects "knowledge is constructed and the best way to ensure learning is through the deliberate construction of something shareable outside of one's head".

3. Professional Training on Physical Computing

With the aim of bringing physical computing to schools, we have offered workshops for computer science teachers over the last two years. The idea was to give them an introduction to physical computing that empowers them to design their own lesson plans based on their particular needs. The approach, objectives and experiences of the workshops are described in this section.

3.1. Approaches to constructionist professional development on physical computing

Many of the more recent publications on professional development in computer science describe long-term professional development of pre- and in-service teachers, particularly in countries where computer science is introduced as a new subject in schools or existing computer science curricula are reformed (e.g. [12], [13]). In those examples, various topics are introduced to the participants. Professional developments with a more focused content area are described e.g. in the Bridge21 Activity Model. It was developed to "help teachers in their preparation for delivering Raspberry Pi activities using methods designed to encourage students to take a more active role in their learning" and to "empower teachers to use delivery methods that enable students to learn by themselves" [14]. These goals are very similar to the ones pursued in our workshop series. The B21 model follows a clear structure described in seven components through which groups are formed, divergent thinking is encouraged, the participants are guided through activities and then invited to collect project ideas and present and discuss them in a larger group. Stager [2] describes a more open-ended format of professional development in which he introduces teachers to robotics as it provides a "terrific context for educators to explore the power of learning technology in a playful, tactile and non-threatening fashion." He uses similar settings for teaching children and educators and suggests four factors to develop settings for successful projects: a good gender-neutral prompt or challenge, appropriate materials, sufficient time and a supportive culture. In his professional development, participants were encouraged to share their experiences with each other, to think about thinking and to discuss their ideas and possible implications for their classrooms. This approach contains an assumption: if teachers shall bring constructionism to their classrooms, they should experience constructionist learning in professional development. This impression is backed in the B21 model, where data shows that teachers incorporated methods used during the training into their classrooms [14]. Brennan [15] describes another phenomenon of educating teachers in fields that involve tools or technology: "The 'learning' is focused on learning *about* the tool/technology or the effects of the tool/technology itself, rather than learning *with* or *through* the technology. The questions that are asked about impacts and outcomes strive to isolate the technology in question as the source of change. We still seem hopelessly stuck in a technocentric view." A key aspect of the model she develops to overcome this technocentrism is that teachers should have the same learning experience as their future students. This aspect is pervasive in the workshops described in this paper.

3.2. Objectives in constructionist professional development on physical computing

The approach of teaching educators in physical computing described here implements the idea of constructionist learning combined with the goals of teaching principles and the ideas of embedded systems, ubiquitous computing and physical computing. Therefore, the professional development affects three dimensions: *content*, *tools* and *pedagogy*. First, teachers have to become acquainted with new content: sensing and actuating technology, hardware in general and embedded systems (feedback and control, concurrency, trade-offs, etc.) are no longer discussed on a theoretical level, but actually experienced in class. Physical computing integrates many methods and underlying ideas of embedded systems, cyber physical systems and smart objects (e.g. [16], [17]). Second, teachers have to familiarize themselves with new tools that allow for creating interactive objects, both on the hard- and on the software side. And finally, they have to reflect their pedagogy, abandon traditional teacher roles and acquire new methods of constructionist teaching: they become learning facilitators rather than knowledge transmitters. They are exposed to practical situations when accompanying students in their individual learning processes and fulfil their teaching role in helping students to solve complex problems. They supply learners with adequately prepared learning materials at the right time and provide learning environments that encourage constructionist learning.

3.3. Experience from workshops on physical computing in computer science education

Based on the ideas described above, a constructionist workshop series was planned, conducted and evaluated with the aim of introducing teachers to physical computing. During the last few years more than 250 teachers participated in these workshops, which were usually conducted in local conferences in blocks of 60 to 90 minutes each. The participants work in various contexts, such as primary, vocational or high schools. The data collection method to evaluate the professional developments consisted of three parts. At the beginning of each workshop, we asked all participants about their motivations for participation, their personal learning objectives and prior experience with physical computing. Then, during the workshops, we observed how they coped with the challenges and provided materials. After each workshop, the teachers were surveyed either by filling in a feedback questionnaire or in taking part in a semi-structured group interview, to find out about their impressions of the workshop, their ideas for classroom use and what they see as opportunities or barriers towards integrating physical computing into their computer science classes. The findings from the quantitative and qualitative analysis of those data are described in this section.
Physical computing has made it to schools: There is a clear trend in the data: while in 2013, the first year when workshops were conducted, only few teachers had heard of physical computing, today most of our participants are aware of it and come with clear expectations. However, because of the various school forms, age groups and diverse prior knowledge, the teachers' motivations and personal interests and learning objectives took very heterogeneous shapes.
Physical computing is new to most teachers: Only few teachers reported of prior experience in physical computing. More recently, teachers stated that they had worked with Raspberry Pi or Arduino, but rarely used it in class. In the beginning, working with sensors, actuators or microcontrollers is intimidating to many. Caution and the fear to break something often outweigh curiosity and joy in experimentation. Teachers regularly told us about preconceptions of physical computing as an activity that involves fumbling with tiny wires and requires broad knowledge about electronics.
Time is a very important issue: Many teachers stated that they needed more time to explore and experiment. Observations support this impression. Most teachers get a feeling how hard- and software work together and reach the point, where they could start working creatively. However, most of them did not reach this point early enough to create any interactive objects in the workshop.
Tools are important: In the workshop series, the participants were introduced to *My Interactive Garden (MyIG)* [4], a prototypical construction kit and learning and programming environment based on Arduino [18], as an example for implementing physical computing in computer science lessons in schools. MyIG allows for immediate tinkering without the need of elaborated skills in physics or principles of electrical engineering, such as soldering (Fig. 1). The aim of learning with MyIG is to collaboratively create an exhibition of interactive objects as they could be found in a futuristic interactive garden. After the workshops, the teachers often explained that they refrained from doing such projects in school because the MyIG toolbox is not available for sale. We realized that teachers do not buy several physical computing kits, experiment with all of them and then decide, which one is good for classroom use.

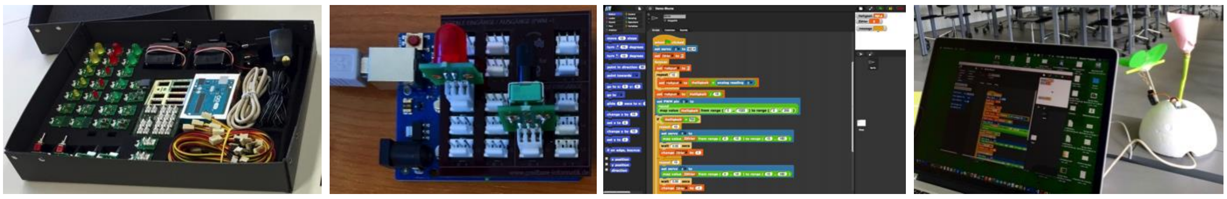


Fig. 1: MyIG Toolbox, Snap4Arduino and example project 'Color-changing blooming flower and butterfly'

Technical failure leads to frustration: Teachers want their classrooms to be free of technical hurdles, such as are common with beta versions of software or that just happen when working with hardware. They are quickly frustrated when things work differently than expected and claim that they were not suitable for classroom use. It is therefore important to have easy to follow manuals and to give explanations why things happen and how they can be avoided in the future.

Teachers take home concrete ideas and examples: From our workshops, they usually take all the work sheets and lesson plans and are delighted when offered editable digital versions. They also enjoy sharing their creations with colleagues, which is in line with the creative thinking spiral introduced by Resnick [19]: in creative processes people imagine, create, play, share, reflect and then start over again with imagining. However, when analysing the data we realized that the approach of MyIG was either useful to teachers in their specific context or not and that hence they would either use it or not. Many teachers did not take the underlying concepts home, but the concrete example.

Hands-on activities are valued: Teachers often stated that they enjoyed taking the role of students and that they perceive hands-on experience as much more valuable than any explanation. They also like working in groups and enjoyed meeting colleagues who have similar interests and problems.

Summarizing, if it is the aim to empower a heterogeneous group of teachers to develop lesson series based on their own ideas in this topic area, they need enough time and various tools and materials to experiment with. It is important to keep the barriers low to get them started. Building on Papert's low floor/high ceilings approach for the Logo programming environment, Resnick [20] suggests that technology for learning should make "it is easy for novices to get started (low floor) and possible for experts to work on increasingly sophisticated projects (high ceiling)" while at the same time providing wide walls: "[…] technologies that support and suggest a wide range of different explorations". When observing these guidelines, both different target groups can be addressed and individual levels of knowledge and learning progress can be considered. It also seems reasonable to let teachers work in groups of interest to create ideas for lesson plans. A similar model is described by Demo [12] as "look – seeInside – modify – share". Here, the focus is on having teachers develop their own materials and on adapting existing approaches to their personal needs. As was shown by Brennan [15], it is also important for teachers to exchange their experiences when they get to know new approaches and put them into practice.

3.4. Design Principles for constructionist professional development on physical computing

From the objectives and considerations described above, several key strategies have emerged that were decisive in developing a framework for constructionist professional development on physical computing. The design principles of this framework are not restricted to physical computing but might easily be adapted to other workshops with similar objectives (Fig. 2).

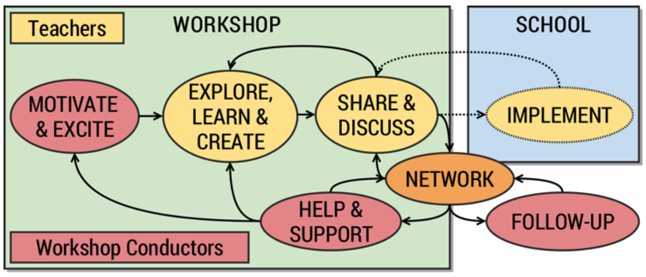


Fig. 2: Design principles for professional development workshops

- *Motivate and Excite:* Give examples and let teachers experience what possibilities physical computing brings and how it enriches computer science classrooms, let them experience the tangibility of computer science, introduce different tools and give only as much input as needed.
- *Explore, Learn and Create:* Let teachers work in groups on interactive objects and ideas for curriculum activities, exhibit the workshop's interactive objects, allow for hands-on experience.
- *Help and Support:* Empower teachers to solve technical problems and help before they give up.
- *Share and discuss:* Let teachers share and discuss their ideas and experience and provide a platform for sharing materials.
- *Network:* Give opportunities for networking with colleagues during and after the workshop.
- *Follow-up:* Help with the implementation of pilot projects, support with hardware and software issues, give advice when needed and interview teachers and students about their experiences.

Those principles were used to develop a two-day workshop that had the same objectives as the previous workshop series, but incorporated the findings from our evaluation and thus improved the workshop setting to better suit the participants' needs. This workshop was supported by Google's CS4HS program [21]. In the following section, the design principles will be explained and illustrated with examples and experiences from the workshop.

4. Implementation

In our workshop *Physical Computing in Computer Science Education: Creative Design and Development of interactive Objects* we had 18 participants, the majority coming from Berlin and Brandenburg. Some participants were from institutions in Hamburg, Lower Saxony and Bavaria. The teachers received credit for their participation in the workshop by the ministry of education as part of their annual professional development obligations. Based on the principles described above, the workshop was planned and conducted on two consecutive days. Basically, it was intended to provide teachers with the necessary skills to successfully teach physical computing in the context of computer science and to develop teaching materials suitable for their particular needs.

4.1. Motivate and Excite

The aim of the workshop was that in the end teachers would be able to create inspiring learning environments that foster students' motivation and creativity with physical computing. They were supposed to creatively engage in hands-on-activities while creating an exhibition of interactive objects in a constructionist environment. We further wanted to display the different ideas pursued by different construction kits and programming environments. Thus, after a short introduction of the aims and didactical concepts of physical computing with the example of *MyIG*, the teachers got the opportunity to tinker and experiment with different combinations of hard- and software. They took the role of students and gained basic knowledge about the *functionality of different kinds of sensors and actuators* as well as suitable *programming environments* such as S4A [22] and Snap4Arduino [23] (Fig. 1). The workshop participants were provided with diverse learning and crafting materials and not instructed how to reach a certain goal. They could follow manuals, solve a jigsaw puzzle, use work sheets or explore the tools on their own. In small groups they then designed various interactive objects with the aim of creating an exhibition of a futuristic interactive garden, e.g. an interactive reaction game that has to be solved to unlock a garden door and a smart fridge for party drinks (Fig. 3). In the discussions later, participants reported of very motivating early success, such as blinking LEDs and positively stimulating aha-effects.

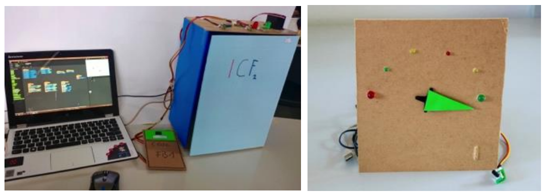


Fig. 3: "Smart Fridge" and "Reaction Game" projects

Fig. 4: Work in teams, sharing and discussion of ideas

4.2. Explore, Learn and Create

The workshop was designed to let teachers work in collaborative groups of interests on their ideas for curriculum activities suitable for their needs. In contributing to the workshop by developing material the teachers were supposed to see themselves as part of a team rather than as learners who are told how to teach. We wanted them to profit from their colleagues' ideas and experience and to share their ideas: we encouraged them to discuss and develop lesson plans, to create interactive objects, to think of possible project themes, etc. In the workshop, the participants worked together in groups of two to five on topics that were relevant and interesting for them (Fig. 4). For instance, one group developed a "smart home" learning unit; another group collected ideas and built prototypes to the theme "Games with Makey Makey". Many teachers also used the opportunity to get to know their favourite tools better and wrote short user manuals for teachers.

4.3. Help and Support

In terms of technical issues, the main aim was to empower teachers to solve technical problems and to offer help before they give up desperately. During the workshop, a technical assistant was always present. Currently, we assist our teachers with the implementation of pilot projects and provide support with hard- and software issues. Advice is given when needed and teachers and students are interviewed about their experiences so that it is possible to reflect on the workshop's impact.

4.4. Share and Discuss Ideas

Considering the creative thinking spiral (see section 0), the groups shared their interactive objects with each other in a gallery exhibition, where every group presented their work. In addition, the outcome of the group works were presented, shared and discussed with all participants. We provided online space for cooperation and sharing materials. In the final part of the workshop, a lively discussion reflected the outcomes of the workshop. It was discussed, if the amount of tinkering and crafting might overstrain students and take too much time from learning computer science or if this is exactly what makes the approach interesting: to balance activities between crafting, technical issues and programming. The groups presented several ideas on methodology and topics and stressed the benefit that with physical computing it is possible to really see what the device does.

4.5. Network

The workshop was meant as an initial platform to build a new community. Throughout the weekend there were many opportunities for networking with colleagues. In order to allow for the teachers to stay in contact with each other, all contact information were shared. A workshop to connect back to each other in person and to continue the collaborative efforts will be conducted after one year, so that everyone has time to make his or her own experience that can then be discussed with the group.

4.6. Follow-Up Activities

To support teachers in actually implementing their ideas, they are provided with the necessary construction kits to be used in their pilot projects if they report their experience, give insights into their particular projects and provide research data. Currently six projects run in parallel, more teachers have announced to start soon. Thanks to the concept of accompanying them in their pilots, the teachers keep contact to the workshop conductors and are eager to give interviews, let their students fill in questionnaires and even permit classroom visits. Additionally, in a follow-up workshop the groups will continue working on curriculum activities based on their experience.

5. Conclusion and Future work

To conclude, the implementation of a constructionist workshop based on the design principles described in this paper was a starting point to empowering teachers in two ways. The participants started to integrate physical computing activities into their computer science classes and developed and shared new ideas and materials that can also be used by other teachers. It is encouraging to see how many of the participants actually put their projects to practice. They give a lot of positive and constructive feedback and make use of the idea of community building, although so far mainly in keeping in touch with the workshop conductor. In order to connect them back to each other, a follow-up workshop is planned for next year. Further, it might be useful to set up an online platform that eases communication among each other. From our experience we conclude that by offering constructionist workshop environments in professional development that allow teachers to follow their personal interests and needs, the chance of affecting their teaching are a lot higher than in instructional training. They are particularly well encouraged to prepare environments for constructionist learning, if they themselves have experienced learning this way. By this means, in creating their own interactive objects, students now actively create knowledge with and about interactive and embedded computing settings. They do so in constructionist settings, just as their teachers did in the workshop.

6. References

[1] N. Rusk et al., "New Pathways into Robotics: Strategies for Broadening Participation," *J. Sci. Educ. Technol.*, vol. 17, no. 1, pp. 59-69, 2008.

[2] G. Stager, "A constructionist approach to robotics," *9th IFIP World Conf. Comput. Educ.*, 2009, pp. 1-12.

[3] S. Libow Martinez and G. Stager, *Invent to Learn: Making, Tinkering, and Engineering in the Classroom*. Torrance, CA: Constructing Modern Knowledge Press, 2013.

[4] M. Przybylla and R. Romeike, "My Interactive Garden – A Constructionist Approach to Creative Learning with Interactive Installations in Computing Education", in *Proc. Constructionism*, 2012, pp. 395-404.

[5] S. Papert and I. Harel, "Situating Constructionism," in *Constructionism*, S. Papert and I. Harel, Eds. Norwood: Ablex Publishing Corporation, 1991, pp. 9-20.

[6] Department for Education, *Computing programmes of study: key stages 1 and 2. National curriculum in England*, no. September. England, 2013, pp. 1-2.

[7] Landesinstitut für Schule und Medien Berlin-Brandenburg, "Teil C Informatik Wahlpflichtfach Jahrgangsstufe 7 – 10 Anhörungsfassung vom 28.11.2014", 2014.

[8] P. Blikstein, "Gears of Our Childhood: Constructionist Toolkits, Robotics, and Physical Computing, Past and Future," in *Proc. 12th Int. Conf. Interaction Design Children*, 2013, pp. 173-182.

[9] M. Przybylla and R. Romeike, "Key Competences with Physical Computing", in *Proc. Key Competencies Informatics ICT 2014*, Potsdam, 2014.

[10] G. Stager, "Outside the Skinner Box. Can Education Technology Make a Course Correction?", *Indep. Sch. Mag.*, vol. 74, no. 2, 2014.

[11] E. Ackermann, "Piaget's Constructivism, Papert's Constructionism: What's the difference?," *Future Learning Group Publication*, vol. 5, pp. 1-11, 2001.

[12] G. B. Demo, "T4T: A Peer Training Model for In-service Teachers," in *Proc. 9th Workshop Primary Secondary Computing Education*, 2014, pp. 120-121.

[13] S. Sentance and S. Humphreys, "Online vs Face-To-Face Engagement of Computing Teachers for their Professional Development Needs," in *Informatics in Schools. Curricula, Competences, and Competitions*, vol. 9378, A. Brodnik and J. Varenhold, Eds. Springer International Publishing, 2015, pp. 69-81.

[14] J. R. Byrneet al., "Computer Science Teacher reactions towards Raspberry Pi Continuing Professional Develop-ment (CPD) workshops using the Bridge21 Model," in *10th Int. Conf. Comput. Sci. Educ.*, 2015, pp. 267-272.

[15] K. Brennan, "Constructionism in the Classroom: Three experiments in Disrupting Technocentrism", in *Constructionism and Creativity. Proc. 3rd Int. Constructionism Conf.*, 2014, pp. 1–8.

[16] E. A. Lee and S. A. Seshia, *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*, 1.5 ed. 2011.

[17] D. O'Sullivan and T. Igoe, *Physical Computing: Sensing and Controlling the Physical World with Computers*. Boston: Thomson Course Technology PTR, 2004.

[18] M. Banzi et al. (2012) *Arduino - Introduction*. [Online]. Available: http://arduino.cc/en/Guide/Introduction.

[19] M. Resnick, "Sowing the Seeds for a More Creative Society," *Learn. Lead. with Technol.*, pp. 18-22, 2007.

[20] M. Resnick and B. Silverman, "Some reflections on designing construction kits for kids," in *Proc. 2005 Conf. Interaction Design Children*, 2005, pp. 117–122.

[21] Google. *Google Computer Science for High School*. [Online]. Available: www.cs4hs.com.

[22] Citilab. (2013). *S4A*. [Online]. Available: http://s4a.cat.

[23] Citilab. (2014). *Snap4Arduino*. [Online]. Available: http://s4a.cat/snap.

The Imaginatorium

Eleonora Badilla-Saxe, *eleonora.badilla@ucr.ac.cr*
Full Professor, University of Costa Rica
Academic Provost, University De La Salle, Costa Rica

Florencia Morado, *morado.florencia@gmail.com*
President, The Patagonia Lab, Chubut, Argentina
Coordinator of Virtual Environments, University De La Salle, Costa Rica

Abstract

The Imaginatorium is a low-tech Maker Space being installed at De La Salle University in Costa Rica. It has its roots in the Fab Lab from the MIT Media Lab, the LuTec at the Institute of Technology of Costa Rica and the Patagonia Lab (Pata Lab) in Chubut, Argentina. With Constructionism at its heart, The Imaginatorium aims to stimulate innovative and creative people, mostly generous with their knowledge looking forward a more caring, equitable and peaceful world.

Keywords

Imaginatorium, MakerSpace, Constructionism, Costa Rica

Innovators Wanted

The Imaginatorium is an innovation laboratory, a space for learning by doing and experimenting with art, science, ground-breaking technologies and new media. It is a place to learn with others, to try without fear of making mistakes, to remix ideas and integrate ideas to the ideas of others. It has been conceptualized to be settled into the University of Costa Rica and De La Salle University in San José, Costa Rica.

Intended to surpass the traditional teacher / student or expert / apprentice roles, it focuses on a collaborative process of creation and innovation, promoting teamwork and public / private cooperation, encouraging self-management, self-organization, entrepreneurship, motivation and leadership.

The proposal is to create a space that brings together students, teachers and community members in an innovative place, with digital technology that allows creation and advanced recreation with appropriate infrastructure. This means having equipped environments that will support the development and growth of innovative projects, mainly targeting to the existence of meeting places where exchange, discussion and confrontation will take place, in an inspirational and creative environment.

Technological upgrading offers the possibility to expand the availability of devices and platforms capable of incorporating multimedia content, which will account for the process started with the expansion of Internet, incorporating devices such as mobile phones, tablets and interactive television. Thus, content generation extends beyond the technological support to work at the conceptual and the transmedia level.

Innovation

The Imaginatorium seeks to promote innovation. Innovation means more than being creative: creativity is only a part of the innovation process.

Whether it is innovation for social or commercial purposes, the construction and implementation of an innovation process involves people from various backgrounds, disciplines and interests, whose performance is fundamental to plan, design, produce, distribute, market and evaluate innovation.

Innovation development requires detecting opportunity gaps. This stage should be focused on the search for suitable opportunities of new products, processes and services, which will improve existing ones or creating the necessary ones. Innovation is not just a great idea; on the contrary, good ideas are just the beginning of innovation and they demand an innovator with leadership profile, persevering and entrepreneur skills. In short: Creativity is to generate new ideas and new ways of looking at existing problems; or of finding new opportunities, taking advantage of emerging technologies or changes in the market.
Innovation is the successful development of new ideas. The process that brings along new products, services, new ways of conducting business and even new ways of doing business. Design is the bridge that connects innovation and creativity and it can be described as creativity displayed for a specific purpose.

Innovators

The Imaginatorium aims to encourage creative and innovative people in Costa Rica so that they can address the complex reality of today's world. The goal is to provide children, youth and adults with access to a place where they can develop key skills to overcome the ambiguity and uncertainty of the XXI Century. In other words, to facilitate a change in their ways of thinking in order to break the paradigms that keeps them in the "comfort zone". Among those skills there are some as challenging as feeling comfortable with the lack of information, developing a vision based on instinct, on prior knowledge and experience, research and define design criteria based on opportunity, not adhering to the method as a panacea - it will only pay off to the extent that the user of the method (the designer) is a deep thinker with flexibility and the skill to adapt to different scenarios.

It is intended that the people who actively participate in The Imaginatorium may gradually:

- Develop a challenging attitude towards the established.
- Lose the fear of taking risks.
- Rely on their instinct as vision in decision-making.
- Feel comfortable with complexity and uncertainty.
- Identify areas of strategic importance.
- Investigate social behavior.
- Recognize opportunity.
- Assume leadership and internal motivation.
- Develop intelligent and autonomous thinking

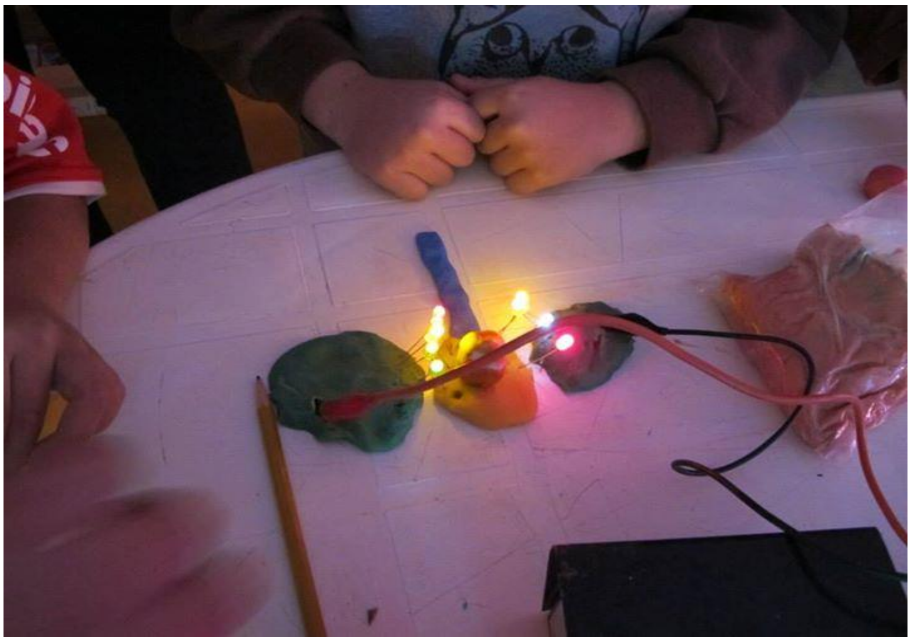
From Cambridge to Costa Rica

The background of The Imaginatorium can be found in Cambridge, Massachusetts when in 2001 a Laboratory of Personal Digital Fabrication (Fab Lab) opened in the MIT Media Lab. The MIT Media Lab is one of the main centers for technological innovation worldwide, dedicated to inter- and transdisciplinary research and experimentation. The Fab Lab is equipped with a number of flexible, digital controller tools and an extensive variety of materials. Its Fab Lab objective is to provide anyone with the possibility to make almost anything. It is not about mass production, but about the possibility for people to build prototypes. Like many other projects that have helped to improve education, science, art and technology, this program emerged from within a laboratory. The laboratory model works on the idea of a future which is simultaneously imaginable and achievable.
In 2002, the Grassroots Invention Group of the Media Lab, led by Bakhtiar Mihkak and the participation of Costa Rican Eleonora Badilla-Saxe, and with the sponsorship of the Central American Institute of Business Administration INCAE, brought a Fab Lab to Costa Rica that was installed at the Instituto Tecnológico (Technology Institute) TEC located in Cartago. It was the first Fab Lab installed outside Cambridge. It was named LuTec (the Technology Luthiers).

Gradually, Digital Manufacturing Labs, Fab Labs, have been set up around the world at the time that the ***Maker Movement*** emerged.

In the UK, the Futurelab experience was an incubator of ideas for educational purposes supported by the BECTA Department (British Educational Communications and Technology Agency), which promoted important research on practical solutions to improve education and learning using digital technologies. We find other examples of art science labs around the world. These emerged over the past five years with David Edwards of Harvard University and other techno-cultural entrepreneurs, who sat bases in Paris and Boston, with partners in Africa, Asia and North America. Its operation is fed by ideas from the fields of art and design, shifting from the educational level to that of cultural and social change. Some centers gather significant examples of artistic and professional groups, offering the community with training and production workshops in programming tools, electronics and digital art among others. For example, the Medialab at El Prado in Madrid as wells as the Espacio Telefónica Foundation and the Spain Cultural Center in Buenos Aires.

In Chubut, Argentina, Florencia Morado founded the Patagonia Lab (Pata Lab) to give the opportunity to children, youngsters and adults, to explore and experience the power of making, using low-tech digital technologies.



Picture 1
Making Light at the Pata Lab

Costa Rica is part of the **Maker Movement.** Along with the personal manufacturing laboratory Fab Lab, established in 2002 at the TEC, there is a research center for Innovation in the Véritas University, a Fab Lab at the Universidad Estatal a Distancia UNED (State Distance University) A High Tech Inventory, recently has been settled at the University of Costa Rica (De Camino, 2015). The Paniamor Foundation, a nonprofit organization that protects the rights of children and adolescents in Costa Rica, has just opened its Innovatorium.

Three main challenges are being addressed so adapt the original idea of the Fab Lab to Costa Rica:

1. From high to low tech. Due to costs and availability, the makerspaces in Costa Rica (except The Inventory at University of Costa Rica) are low tech. A broad research on low tech

technology has been carried out to define the resources that are suitable for the workshops and spaces. Re-cycled material has been included to interact with digital technology.
2. Collaboration: formal education has trained people to work isolated and not share learning or productions. Mentors and facilitators at The Imaginatorium must be very aware in stimulating collaboration. It´s a mindset shift, thus, a slow process.
3. Gender gap: in general male are more inclined to making with digital technology. In order to attract females, a wide range of materials and activities are proposed: storytelling, playing with lights, and others.

The low tech Imaginatorium being installed at the University of Costa Rica and De La Salle University, in collaboration with the Pata Lab from Argentina aims to be part of this Maker movement of manufacture, creation and innovation, increasingly spreading around the world and of a network of personal manufacturing laboratories starting in Costa Rica.

The Imaginatorium in ULASALLE will encourage collaboration between the people at the laboratory and among other manufacturing spaces settled in the country and beyond, to solve the problems and challenges that arise when developing projects.

Thus, developing a collaborative spirit it will seek to contact and coordinate the network of laboratories and makerspaces present in the country and internationally. It is important to remember that the Maker movement involves not only the ability to make and share products, but also to share the construction process based on the multimedia documentation and spaces provided by social networks and virtual communities.

Videos, blogs and photos become the documentation and the process of belonging to a global community. Sharing the process involves understanding the fundamentals and the development of communication and interpersonal skills.

Constructionism at The Imaginatorium

The processes of making, learning, creating, recreation and construction in The Imaginatorium is be addressed from the Constructionist perspective proposed by Seymour Papert.

Specifically, peer learning and collaborative learning are promoted as one of the most powerful forms of interpersonal growth and team knowledge construction. As noted by Papert, the best way to learn is through experience, from the experiences of others and the multiple points of view on a problem.

The conceptual bases of constructionism start with Jean Piaget who argues that "to understand is to invent" (Piaget, 1976) and asserts that people learn in their relationship with the environment. Later, Lev Vigotsky (1979) elaborates indicating that people learn in their relationship with the environment and with other people, thereby emerging the social constructivism. Piaget’s Constructivism suggests that knowledge is not given to people who learn, but it is constructed inside the mind from the analysis of problematic situations.

Based on the later, Seymour Papert (1980), called "the father of the maker movement" developed constructionism. It involves the assertion that knowledge develops from the construction of objects, meaning by objects things like music composition, toys, software, movies, mathematical models, etc. (Martinez & Stager, 2013).

> *From constructivist theories of psychology we take a view of learning as a reconstruction rather than as a transmission of knowledge. Then we expand the idea of manipulative materials to the idea that learning is most effective when part of an activity the learner experiences as constructing a meaningful product.* (Papert, 1986, as quoted by Martínez & Stager)

To Papert, knowledge is produced out of the mind, when the student engages personally and significantly, allowing the construction of real and shareable knowledge. He then deepens this conceptualization by including technology, especially computers which enhanced, almost infinite, the possibility of building based on their communication, research, modeling, object control, software development and project sharing skills.

All of his developments convey a clear message of "learning from the active construction of knowledge through the act of doing something that can be shared," referring to programming, robotics and the creation of multimedia products.

The Imaginatorium seeks to learn through playing with digital technology, exploring, making mistakes, trying to improve. It values other people's ideas in order to improve its proposals. It holds a high level of motivation and a huge spirit of collaboration among peers.

Spaces at the Makerspace

The Imaginatorium offers spaces to MAKE:

- Inter-art: creative and artistic uses of electronics and programming.
- Digital inclusion: training and integration in the digital culture.
- Viewing and immersion: Visualization and interaction strategies and tools embedded in digital media.
- Sound: sound and audiovisual creation.
- Automation: research and experimentation in prototyping development and programming
- Inter-media narratives.
- Creating and using games.

Picture 2
Spaces to Make: Inter-art: creative and artistic uses of electronics and programming

Working areas are visualized interdependent and inter articulated, each of them aiming for the development of knowledge and different cognitive and digital skills which are less emphasized in formal learning.

Playing with Codes and Bits: emphasis is given to coding the digital and analog interfaces through programming**.** Recreational electronics and creative robotics will be implemented**.**

Sensing our Stories: The use of all senses will be explored through transmedia narratives, immersive fiction and innovative artistic experiences**.**

Making on the Screen: Designing of interfaces will be vital to experience with topics such as micro formats **(**short films with mobile phones**)**, game simulations, video games, and interactive digital books**.**

All Welcome!

Although children and adolescents will probably feel more attracted to make and build in The Imaginatorium, it will be open to anyone interested in learning with new technology in a playful and constructionist atmosphere.

Imaginatorium will be open to**:**

- Students from different ages and interests
- Faculty members
- Community at large
- Multimedia producers
- Content compilers
- Computer systems professionals and students
- Software developers
- Designers
- Technicians
- Artists

The workshops: Make to Think

The Imaginatorium proposal offers a journey through crosscutting themes, organizing the digital storytelling, programming, electronics, robotics and music in workshops of different levels of complexity. They all seek to develop meaningful and lasting knowledge that can be shared by the core areas through playful, creative and innovative production.

But we need to remember that, as we established before the workshops are not a means in themselves: there is a higher level purpose in making. This is thinking, learning, collaborating, sharing, loosing fear of taking risks and making mistakes.

A workshop developed at the University of Costa Rica in 2011 with Faculty of different Departments and Disciplines: Make to Think: Ideas, Spaces and Tools, sets the pedagogical tone for de design of worshops at The Imaginatorium. (Urrea, Lara, Badilla, Miranda y Barrantes. 2012).

The digital narrative workshop is based on comics design experiences that will join personal characters with environments and characters shared by others on the Internet. Following, the story is animated using the Stop Motion technique, heightening the construction of the story and then interacting with the materials and the animation technology. Finally, there is three-dimensional design, which allows the possibility of 3D printing of personal creations.

The programming, electronics and robotics areas work almost simultaneously with introductory workshops to code with Blockly and Scratch, basic notions of electronic circuits through simulators and with electricity conductive mass (squishy circuits), experiences with Makey Makey plates, creating interactive textiles with LillyPad, and finally in a more complex level programming with Arduino and Raspberry Pi.

The programing area also connects with 3D printing through the Minecraft world building workshops, printing these creations on 3D after that.

Music and sound come hand in hand with all of the above axes. Digital stories are voiced; electronic music is created combining electronic circuits with software, mixers, contact microphones and everyday items. Programs created in Scratch allow music remix.

Thus, it is difficult, if not impossible, to analyze these workshops and refer to some of them as responding to one line of knowledge or to the dissected learning of any discipline or skill. Imaginatorium Workshops are a whole assemble of art, technology and science complex where participants are free to make their own journeys.

Our Utopia

Our Utopia is that the Imaginatorium at the University of Costa Rica and De La Salle University, through a process of self-organized learning from available resources and proposals, allows:

- Democratization of digital culture, creating new opportunities for innovation in the intersection between technology, design and social sciences.
- Cross way collaboration: through the exchange of disciplines, knowledge and people to improve the quality of life by promoting cooperation and the dynamics of interconnected networks.
- Co-creation for collective learning: rethinking the role of the receptor-cultural consumers: recognizing their ability and role as creators-producers of knowledge.
- Co-management of change and complexity: design and facilitate transformation processes in people, companies, institutions, organizations, populations and territories, encouraging motivation, self-management and leadership.
- Co-producing: co-produce new products, processes, services; create new collaborative strategies, agreements and forms of organization.
- Learn in order to value and preserve: to improve the construction of meaning and strategic capabilities of groups and sites involved, enhancing their identity and cultural, material and natural heritage.

In short we want to be part of the Maker movement to collaborate in stimulating innovative and creative people, but mostly people generous with their knowledge, looking forward a more caring, equitable and peaceful world.

References

De Camino, Tomás, Badilla, Eleonora. 2015. *Inventar para aprender.* podcast en Lenguajeos, Radio Universidad, http://radios.ucr.ac.cr/radio-universidad/programas/41-apoyo-a-la-academia/1215-lenguajeos-rucr.html 07 de setiembre

De Camino, Tomás. 2015. *Inventoría*, https://www.facebook.com/inventoria?ref=profile

Mikhak Bakhtiar, Badilla Eleonora, et al, 2002, *Redesigning Teacher Education with the use of Digital Technology:* Project LUMEN UCR-MIT Media Lab

Mikhak, Villegas-Lemus, Badilla, 2002 **LuTec: The Luthiers of Technology**

MIT Media Lab, **Fab Lab** https://en.wikipedia.org/wiki/Fab_lab

Paniamor Fundación www.paniamor.org

Papert, Seymour. 1990. A Critique of Technocentrism in Thinking About the School of the Future, en http://www.papert.org/articles/ACritiqueofTechnocentrism.html

Papert, Seymour. 1987. Desafío de la mente. Buenos Aires, Argentina: Ediciones Galápagos.

Papert, Seymour. 2000. What's the Big Idea? Steps toward a pedagogy of idea power. IBM Systems Journal, 39(3y4), 720-729.

Papert, Seymour, y Resnick, Mitchel. 1995. Technological fluency and the representation of knowledge. Proposal to the National Science Foundation. Cambridge, MA.

Promotora de la invención. 2002. http://wvw.nacion.com/viva/2002/septiembre/11/soc1.html

Stager & Martinez. 2013. *Invent to Learn: Making, Tinkering and Engineering in the Classroom.* Constructing Modern Knowlege Press. Torrance. CAç

Urrea, Lara, Badilla, Miranda y Barrantes. 2012. *Hacer para pensar: ideas, espacios y herramientas,* https://www.academia.edu/9409457/HACER_PARA_PENSAR_IDEAS_ESPACIOS_Y_HERRAMIENTAS Revista Actualidades Investigativas en Educación, vol 12, n 1, INIE, Universidad de Costa Rica

Turtles All the Way Down: Presenting LevelSpace, a NetLogo Extension for Reasoning about Complex Connectedness

Arthur Hjorth, *arthur.hjorth@u.northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

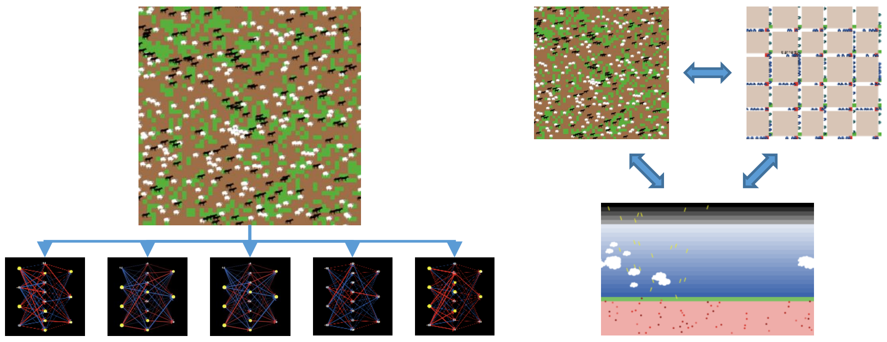
Corey Brady, *cbrady@northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Bryan Head, *bryan.head@u.northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Uri Wilensky, *uri@northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Abstract

In this paper we present LevelSpace, a fundamental extension to the NetLogo agent-based modeling language and software. NetLogo provides a low-threshold, high-ceiling environment for creating models of complex systems; LevelSpace enables *multiple* NetLogo models to interact with one another in “model ecologies.” The LevelSpace extension does this by allowing curriculum designers, researchers, and learners to programmatically open, run, and close models from inside the NetLogo language, thus allowing entire models to function as *agents* or *turtles* within Multi-Level Linked systems.



The paper first presents existing research on using agent-based models (ABMs) in education as a supportive environment for fostering complex systems thinking. It then argues that existing design tools are limited in the ways in which learners can engage with and question the assumptions built into ABMs or the conceptual boundaries that ABMs implicitly or explicitly draw around the phenomena they model. As a remedy for these limitations, we present LevelSpace and demonstrate how we have used it in two early studies to support and study thinking about Multi-Level Linked systems. We argue that connecting models in this way is a powerful constructionist design tool, both for eliciting thinking about individual and linked systems and for building and refining such systems in educational contexts.

Keywords

Agent based modeling; NetLogo; complex systems thinking; Multi-Level Linked systems.

Complex Systems Thinking and Objects to Think With

The ability to reason about complexity is increasingly recognized as a core competence for successful participation in the STEM disciplines, in the social sciences, and in modern society in general. This has raised the importance of research on how people reason about complex systems—both the productive intuitions and resources that people can tap into to understand complexity and the pitfalls that they can fall into, when these intuitions are misapplied. On the side of pitfalls, research has identified a number of problematic patterns in thinking about complexity and emergence. First, when learners interpret systems that involve individual, agent-level behaviour and aggregate, system-level behaviour, they often “slip” between levels (Wilensky & Resnick, 1999) ascribing causality to interactions at the wrong level; Second, learners are often convinced that the presence of system-level order implies that there is a mastermind that controls the system, and they reject the idea that such order can emerge from randomness (Resnick & Wilensky, 1993); Finally, learners often apply inappropriate causal intuitions when attempting to predict system outcomes and behaviours (Wilensky & Abrahamson, 2006).

This research has taken a constructivist view of cognition and learning, framing these patterns not as misconceptions, but as expressions of productive cognitive structure that contains the seeds of correct and effective reasoning. To support learners in developing more effective ways of thinking about complexity, then, researchers from this perspective do not seek to “root out” naïve intuitions. Instead, they aim to first identify, and then recruit these knowledge resources as building blocks for constructing more viable models that do a better job of explaining emergent phenomena. Much of this work focuses on using computational models to aid in thinking about complex systems. Because computational models are “runnable,” they provide learners with the means to explore the implications of the conceptual systems they represent. In particular, agent-based modelling (ABM) has been used to study and teach complex systems thinking for two decades now, possibly due to ABMs’ close ontological alignment with complex systems (Jacobson & Wilensky, 2006; Wilensky & Jacobson, 2014): like complex systems, ABMs consist of decentralized computational elements (called ‘agents’) that interact with each other. By programming the behaviours of agents and distributing those behaviours to many – often hundreds or thousands – of these agents, modellers can computationally “grow” (Epstein, 1999, 2012; Epstein & Axtell, 1996; Wilensky, 2001) complex systems phenomena that have been observed in the real world.

A large body of literature on using agent-based modelling to support learners in complex systems thinking has deployed activities designed in NetLogo (Wilensky 1999), an extension of Seymour Papert’s Logo language (e.g. Papert, 1980) that enables *many* turtles to interact. NetLogo is one of the most widely used (Thiele, Kurth, & Grimm, 2012) ABM environments and is specifically designed to support modelling of complexity and emergence. Its low-threshold, high ceiling design makes it both a powerful, expressive environment and an effective tool for eliciting complex systems thinking in novice-modeller learners. NetLogo has been used for learning in domains as diverse as evolution (Centola, McKenzie, & Wilensky, 2000; Wagh & Wilensky, 2012, 2013, 2014; Wilensky & Reisman, 2006), chemistry (Brady et al., 2015; Levy & Wilensky, 2005; Stieff & Wilensky, 2003), material sciences (Blikstein & Wilensky, 2008, 2010), and physics (Sengupta & Wilensky, 2009). In all these domains, many of the central focuses of study can be conceptualized as complex systems: For instance, in evolution, individual organisms struggle to eat and reproduce, while at the aggregate level species evolve new traits or split off entirely into new species. In chemistry, individual molecules move, collide, and interact, while at the aggregate level properties such as temperature and pressure emerge. In material sciences, individual molecules interact to produce properties of materials in the aggregate. And in physics, interactions between electrons and atomic cores in conductive metals lead to the emergent phenomena of current and resistance.

By offering an external, executable representation of a phenomenon, an ABM can be adopted by a learner as an object-to-think-with (Papert, 1980) about that phenomenon at various levels: at the most “micro” level, learners can reason about how individuals of the system interact with each other and reflect on the alignment of their mental model with the model's representation of this micro-level. In this way, the agents of the ABM can invite *embodied modeling* (Wilensky & Reisman, 2006), in which learners project themselves into the 'world' of agents and adopt the agent perspective. At the same time, ABMs can also be adopted by learners as supports for reasoning at the aggregate level. This is because they represent emergence "authentically"—in ABMs, emergent phenomena are *not* directly programmed: they are the computational effects of running the system of agent-level rules and behaviors. Finally, ABMs provide learners with so-called *mid-level* objects-to-think-with (Levy & Wilensky, 2008). Often arising through reflection on the "processes-to-think-with" mentioned above, mid-level phenomena are ad hoc collections of agents that can help the learner reason through how the individuals work together to produce the aggregate outcome.

Limitations of Current Modeling Tools and a new Line of Inquiry

The research on using ABMs in learning about complex systems has come a long way. However, current design tools have some limitations: even with languages like NetLogo, in which learners always have access to the underlying code, there are certain design and modeling decisions that are difficult for learners to address or question.

Whenever we build a model of a phenomenon, we draw conceptual boundaries around it, and we code into the model a wide array of assumptions about how and why things happen, what is important, and what is irrelevant. In general, we express our particular epistemological interpretation of the phenomenon we are modeling. These features are actually what make models such a powerful "window" into modelers' ways of thinking (Noss & Hoyles, 1996): models are simplified, bounded abstraction of the real world, and in the real world almost everything is connected. Therefore, it is powerful to be able to question and adjust the conceptual boundaries of a model—both in one's own inquiry process and in exploring and critiquing the constructions of others. For instance, the NetLogo classic model of Wolf Sheep Predation (Wilensky, 1997) shows an ecosystem in which grass, sheep, and wolves coexist. Many other systems that affect the real-world version of this ecosystem are excluded: systems like the climate, pollution, soil chemistry, etc. But what if a learner wished to critique the model's assumptions about what is important enough to be included, or if they were simply curious about how "neighboring" systems like the ones listed above interacted with the wolf-sheep ecosystem, and wanted to explore this? Currently, agent-based modeling thinking tools have not allowed us to link a model of one system to models of other systems that we think may be related.

We believe that such "thinking by linking" is a productive line of inquiry. In any discipline, a significant part of the meaning of an idea comes from how it is connected with other ideas. The same is true of complex systems. Reflecting on the connections between systems is not only productive in creating a "big picture" – appreciating and exploring these connections also triggers new thinking about the nature of the component systems. Research on students' reasoning about various phenomena suggest that the knowledge activated during reasoning is highly context sensitive (Russ, Scherr, Hammer, & Mikeska, 2008), and that learners' explanations depend on what particular components of the system they are attending to (Kapon, 2010; Kapon & diSessa, 2012). In viewing an ABM as part of a linked system of models, learners shift the context and frame of their reflection on each of the models involved, providing new and potentially illuminating perspectives on them.

Presenting LevelSpace

We hypothesized that reasoning about how systems are connected could be a productive way to elicit, study, and ultimately strengthen complex systems thinking using agent-based modeling. To study this, we designed and built the LevelSpace NetLogo extension (Hjorth, Head, & Wilensky, 2015). LevelSpace is an open source extension[38] written for NetLogo, using NetLogo's Extension API and its Controlling API. NetLogo has proven to be a low-threshold, high ceiling environment,

[38] Full LevelSpace documentation and source code are available on GitHub: www.github.com/NetLogo/LevelSpace

as evidenced by its wide use as a Constructionist learning tool in classrooms and its equally wide use as a "serious" modelling tool in peer-reviewed research; for this reason we chose to design the semantics and syntax of LevelSpace to emulate NetLogo.

LevelSpace allows modellers, learners, and curriculum developers to program what we call Multi-Level Linked (MLL) systems of models; entire ecologies of models that each are unitary and fully functional models in their own right, but that affect each other in different ways. We have constructed ecologies consisting of thousands of concurrent models: the only limitation is the host computer's resources. LevelSpace's primitives allow modellers to code their models to interact in different topologies or structures: some models may run side-by-side, allowing us to think about how they affect each other. Other models may interact in a hierarchical way so that some models are "contained inside" other models.

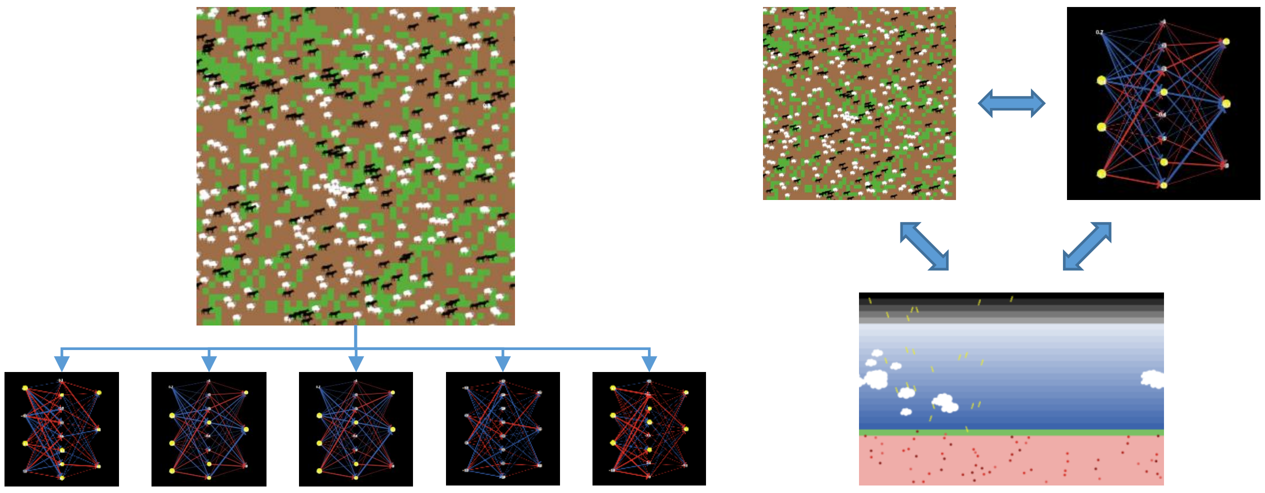


Figure 9: Multi-Level Linked Systems. Each wolf and sheep run their own neural net model which acts as their 'brain'. [Left] An ecosystem, a polluting traffic system, and the environment interact [Right].

Pilot Studies and Preliminary Data

As part of our development and design efforts, we created two pilot activities that we implemented in two very different settings: in the first, we interviewed two teachers who were experienced NetLogo users in public and parochial schools outside of Chicago. In the second, we designed a small unit that we implemented in a low-SES, selective-enrollment school on the south side of Chicago. In the following, we will discuss our two cases and our findings there, and then discuss what we see as potential areas of future research in complex systems thinking and design using multi-level linked systems.

Pilot 1: Connecting Existing ABMs as a Window on Systems Thinking in Levels

In our first pilot, we interviewed two teachers, Hank and Christy, who both had previously used NetLogo in their teaching. As this was the first time we had ever tested think-aloud modelling activities involving LevelSpace with people who are not affiliated with our research lab, we took a very open-ended approach, being simply curious what would happen if people who were familiar with two separate models were then asked to connect them, cognitively and computationally. We sat down with both teachers together, and recorded both their interpersonal and on-screen interactions with Camtasia. We first presented Hank and Christy with the Wolf Sheep Predation model and encouraged them to explore it by changing variables, observing how that changed the runtime behaviour of the model. We then discussed the model with them, probing their understanding of the model at the level of individual sheep and wolves, and we asked them to make predictions about how the model would respond to different changes in its parameters that we suggested. We continued this until we were sure they felt that they understood the model. We then did the same with the Climate Change model. Finally, we asked them to propose ways that the two models might be connected, and how the connections would lead the two systems to affect each other. Hank responded,

> "Vegetation I think is a key thing if we have climate change. Now we have different environmental conditions for the producers, which may affect how many resources are there at the bottom of the primary producers to support the rest of the population. *[...]* And if there is less vegetation it's gonna affect the sheep, which is ultimately gonna affect the wolves as well."

It should be clear from this quote that Hank is connecting his knowledge to aspects of the parameter space of the two systems and reasoning productively about how they might interact. In this particular example, he reasons that the Climate Change model could change how much grass there is, which would affect the Wolf-Sheep world by changing the dynamics of the sheep population. Christy then responds,

> I would think you'd have the biggest thing in the ratio of the carbon dioxide by photosynthesis vs. cellular respiration. *[...]* If you have more of the grass then you can do more photosynthesis, and you'll have less greenhouse gases then the climate change is not going to be as significant as if you have more wolves and sheep.

Likewise, Christy has no problem connecting what she knows about one system to what she knows about the other and reasoning about how they interact dynamically. What is interesting, however, is that Christy seems to attend to different aspects of the connected system: whereas Hank focused on how climate change would affect the population dynamics among wolves and sheep, Christy focuses on how the ecosystem model might affect the climate change model. To us, this illustrates one of the powerful aspects of the LevelSpace: by simply asking people to connect two systems that they know well with LevelSpace, we can gain a "window into their thinking" about the salience of particular components in each system, and in the linked system. To Christy, the most salient component of the linked system was how the eco system would produce and absorb CO2; whereas for Hank, the ways in which climate change might affect the growth of grass (which in turn would affect the sheep population, and then the wolf population) was more salient.

Another interesting feature of the think-aloud activity was that while both Christy and Hank were able to generate interesting ideas about ways to link the two systems at the aggregate levels, as well as to reason about first order inter-model effects of such connections, it seemed less easy or salient for them to reason about these links or to mentally simulate second-order and feedback effects between the models at the micro-level. We know from the literature that reasoning from micro- to macro-level helps learners avoid many of the cognitive biases that we have when reasoning about complex systems. To us, this suggests that future design work should focus on helping learners make these connections. Our current work therefore includes adding visualization and graphing tools to LevelSpace, to support learners' reasoning between micro- and systems-level when thinking about the dynamic properties of multi-level linked systems.

Pilot 2: "What's Missing?" as an Invitation to Extend ABMs through Linking

Our second implementation was in an informal learning setting: a computer club for girls in a low-SES selective-enrolment magnet school in Chicago. There were 13 girls in the club, none of whom had any prior experience with agent-based modelling. Our unit ran over the duration of four 90-minute sessions, introducing computer modelling in general, ABM in particular, and investigating why it is useful as a general scientific approach. As part of this unit, we discussed how models are always simplifications, and thus that there are always aspects of the world that we choose to exclude from our model of that world. The first hands-on activities in the unit supported students in exploring the modelling components of ABMs (especially turtles and patches) and building simple models from scratch with these components. After this, we provided them with a template for a "Model Brief," which structured their descriptions of ABMs designed and built by others. In small groups, the students conducted a survey of the NetLogo Models Library, each group using this template to capture their observations of two chosen models. Sharing these provided the group with a sense of the contents of the Library and of the range of phenomena that could be represented with ABMs.

We then introduced Wolf Sheep Predation and structured students' exploration of the model by asking them to do two things (1) attempt to create a "stable" system, and (2) identify the rules and behaviours of the model (both at an aggregate level and at the level of individual wolves and sheep). We deliberately introduced the "stability" challenge without defining what a stable system was: in the debriefing session, students' explanations of systems that were "almost" stable illuminated both their working definitions of this term and their emerging understandings of the systems relations between grass, wolves, and sheep.

We used the students' fine-grained descriptions of the wolf-sheep system and its population dynamics as a starting point for a brainstorm about what real-world actors, factors, and events were missing from the NetLogo model. For instance, the girls all identified "grass grows" as a part of the model, and several girls suggested that *seasons* were missing from the model (as well as the factors of sunshine, weather patterns, and water flow). From on-the-spot interviews with some of the girls, we believe that the general salience of grass-centred dynamics in the model came from the very simplicity of the wolf-sheep control of grass. In the model, there is a simple switch that controls whether grass is a finite or infinite resource (that is, whether the world is always green or whether grass is depleted and regrows). Because that simple switch made such a huge difference in the functioning (and stability) of the system, girls were interested in making it more nuanced.

Other groups were interested in introducing more intermittent and catastrophic events, such as disease, fire, or flooding. Again based on informal questioning of the girls as they worked, these interests seem to stem from two sources: (1) a "narrative" view of the world of the model, in which "everyday life" can be disrupted by catastrophes, and (2) the repeated patterns of population fluctuation that appear in the model's runtime graphs, and an interest in discovering what could disrupt these patterns.

These and other student ideas for expansions of the Wolf-Sheep model led to a variety of proposals to "affect the model from the outside." Two ways of doing this emerged: (1) link the ABM with *other* models from the NetLogo models library, and (2) introducing features or ideas inspired by other models through commands in the command centre. For instance, the "catastrophes" brainstorm led to an idea to connect the Wolf-Sheep model with the Fire model (cite), with the idea that with more vegetation, the likelihood of a fire might increase. And the "seasons" brainstorm led to an idea that making patches white could simulate snow, and making them blue could simulate bodies of water or flooding in a rainy season. The code of the Wolf-Sheep model actually enables these ideas to *work* if students modify the patches from the command centre, so that the girls could imagine simple "season controlling" just as a *timer* that periodically triggered colour changes in some of the Wolf-Sheep model's patches.

Before the next class session, we implemented several of the girls' ideas in LevelSpace, preparing linked systems for them to explore. In spite of the *technical* complexity of our prototype user interface and the *conceptual* complexity inherent in linking systems in this way, the girls were able to explore and modify the behaviour of coupled systems that realized their own ideas about expansions to Wolf-Sheep predation. Their interactions have fuelled our ongoing development of LevelSpace and an interface designed specifically to *explore* the behaviour of multi-level linked systems while they are running.

Conclusion

In this paper, we have presented LevelSpace, a NetLogo-based platform for creating multi-level linked systems of ABMs, and for thinking about the connectedness of complex systems phenomena. The work that we have described in this paper is still of a very preliminary nature, both on the side of design and development, and on the side of implementation and research. Ongoing efforts include (a) building flexible and intuitive user interfaces for authoring multi-level linked systems and exploring the behaviour of such systems as they are running, (b) supporting multi-computer scenarios, in which linked systems can consist of models running on a network of different machines, as in a classroom group of students and computers, and (c) developing curriculum modules that use LevelSpace to extend model-based inquiry with ABMs to enable learners to follow their own interest-driven lines of inquiry about multi-level linked systems.

In our first study, we saw how idiosyncrasies in the perspectives and ways of thinking of each participant both contributed to shared inquiry and revealed features of their underlying cognitive structures and distinctive patterns in what participants attended to in the two linked ABMs. Under a constructivist perspective it becomes a high priority to study such components of understanding and the patterns in different learners' attention to particular features of a linked system and the particular knowledge that these activate. This study suggests that LevelSpace activities can play a central role in eliciting and externalizing these aspects of thinking. In our second study, we saw that the process of analysing a model's behaviour and attempting to achieve a goal outcome (stability) gave learners an entry point to imagine ways in which it could fit into a larger system. This in turn raised specific questions about the assumptions of the initial model, and acted as a starting point for inquiry and model-construction. Questioning assumptions is a key component of computer modelling literacy, and a core scientific skill.

In spite of their preliminary nature, the studies presented here are suggestive of an extremely rich space for research on how learners reason about complexity that can be explored with LevelSpace. As we have shown, working to construct and explore multi-level linked systems not only contributes to an understanding of the "model ecology" represented by the MLL system as a whole: conceiving of, reasoning with, and constructing MLL systems also provides new perspectives on the individual ABMs that are incorporated into model ecologies. This is consistent with other research on thinking with ABMs. With LevelSpace, we have made it possible to think of entire ABMs as agents: and our related hypothesis is that there is a *reflexive* process at work in understanding of these new agents and the systems they can participate in.

References

Blikstein, P., & Wilensky, U. (2008). Implementing Multi-Agent Modeling in the Classroom: Lessons from Empirical Studies in Undergraduate Engineering Education. In *International Conference of the Learning Sciences*. Utrecht, The Netherlands.

Blikstein, P., & Wilensky, U. (2010). MaterialSim: A constructionist agent-based modeling approach to engineering education. In *Designs for learning environments of the future* (pp. 17–60).

Centola, D., McKenzie, E., & Wilensky, U. (2000). Survival of the Groupiest: Facilitating Students' Understanding of Multi-level Evolution through Multi-Agent Modeling - The EACH Project. In *Proceedings of The Fourth International Conference on Complex Systems*. NH: New England Complex Systems Institute.

Epstein, J. M. (1999). Agent-based computational models and generative social science. *Generative Social Science: Studies in Agent-Based Computational Modeling*, 4–46.

Epstein, J. M. (2012). *Generative Social Science: Studies in Agent-Based Computational Modeling*. Princeton University Press.

Epstein, J. M., & Axtell, R. L. (1996). *Growing Artificial Societies: Social Science From the Bottom Up* (1St Edition edition). Washington, D.C: Brookings Institution Press MIT Press.

Hjorth, A., Head, B., & Wilensky, U. (2015). *LevelSpace NetLogo Extension. http://ccl.northwestern.edu/levelspace. Center for Connected Learning and Computer-Based Learning. Evanston, IL.* Retrieved from www.github.com/NetLogo/LevelSpace

Jacobson, M., & Wilensky, U. (2006). Complex systems in education: Scientific and educational importance and implications for the learning sciences. *Journal of the Learning Sciences*, *15*(1), 11.

Kapon, S. (2010). Instructional explanations as an interface—the role of explanatory primitives. In *2010 PHYSICS EDUCATION RESEARCH CONFERENCE* (Vol. 1289, pp. 189–192). AIP Publishing.

Kapon, S., & diSessa*, A. A. (2012). Reasoning through instructional analogies. *Cognition and Instruction*, *30*(3), 261–310.

Levy, S., & Wilensky, U. (2005). An analysis of students' patterns of exploration with NetLogo models embedded in the Connected Chemistry environment (pp. 11–15).

Noss, R., & Hoyles, C. (1996). *Windows on Mathematical Meanings: Learning Cultures and Computers*. Springer Science & Business Media.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, New York: Basic Books, Inc.

Resnick, M., & Wilensky, U. (1993). Beyond the deterministic, centralized mindsets: New thinking for new sciences. In *annual meeting of the American Educational Research Association, Atlanta, GA*.

Russ, R. S., Scherr, R. E., Hammer, D., & Mikeska, J. (2008). Recognizing mechanistic reasoning in student scientific inquiry: A framework for discourse analysis developed from philosophy of science. *Science Education*, *92*(3), 499–525. http://doi.org/10.1002/sce.20264

Sengupta, P., & Wilensky, U. (2009). Learning Electricity with NIELS: Thinking with Electrons and Thinking in Levels. *International Journal of Computers for Mathematical Learning*, *14*(1), 21–50.

Stieff, M., & Wilensky, U. (2003). Connected Chemistry—Incorporating Interactive Simulations into the Chemistry Classroom. *Journal of Science Education and Technology*, *12*(3), 285–302.

Thiele, J. C., Kurth, W., & Grimm, V. (2012). Agent-Based Modelling: Tools for Linking NetLogo and R. *Journal of Artificial Societies and Social Simulation*, *15*(3), 8.

Wagh, A., & Wilensky, U. (2012). Evolution in Blocks: Building Models of Evolution using Blocks. Presented at the Constructionism 2012, Athens, Greece.

Wagh, A., & Wilensky, U. (2013). Leveling the playing field: Making multi-level evolutionary processes accessible through participatory simulations. In *Proceedings of the Biannual Conference of Computer-Supported Collaborative Learning (CSCL), Madison, Wisconsin*.

Wagh, A., & Wilensky, U. (2014). Seeing patterns of change: Supporting student noticing in building models of natural selection.

Wilensky, U. (1997). *NetLogo Wolf Sheep Predation model. http://ccl.northwestern.edu/netlogo/models/WolfSheepPredation.* Center for Connected Learning, Northwestern University. Evanston, IL.

Wilensky, U. (2001). Modeling nature's emergent patterns with multi-agent languages. In *Procedings of EuroLogo*.

Wilensky, U., & Abrahamson, D. (2006). Is a disease like a lottery?: Classroom networked technology that enables student reasoning about complexity. In *Proceedings of the American Educational Research Association*. San Francisco, CA.

Wilensky, U., & Jacobson, R. (2014). Complex Systems in the Learning Sciences. In *The Cambridge handbook of the learning sciences* (Vols. 1–2). Cambridge, UK: Cambridge University Press.

Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories—an embodied modeling approach. *Cognition and Instruction*, *24*(2), 171–209.

Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and Technology*, *8*(1), 3–19.

Working Toward Equity in a Constructionist Scratch Camp: Lessons Learned in Applying a Studio Design Model

Deborah Fields, *deborah.fields@usu.edu*
Instructional Technology & Learning Sciences, Utah State University

Lisa Quirke, *lisa.quirke@mail.utoronto.ca*
Faculty of Information, University of Toronto

Tori Horton, Jason Maughan, Xavier Velasquez, Janell Amely, Katarina Pantic,
Instructional Technology & Learning Sciences, Utah State University

Abstract

This project examined learning in a series of three, week-long Scratch summer camps intended for novice kids aged 10-13. In this paper we share our first steps toward to building a more rigorous and equitable constructionist learning environment by applying an art studio design model of pedagogy to computer science. Our goals are: 1) to articulate our application of the studio model of design to learning programming in the Scratch Camps, 2) to investigate how widespread and deep campers' learning was, and 3) through this investigation to consider lessons learned that can be applied to future constructionist learning environment designs. Our study applied quantitative and qualitative analysis to a combination of automated backend saves of campers' projects, observational data, and frontend saves of campers' projects.

Keywords

computer science education, informal learning, studio design, computational thinking

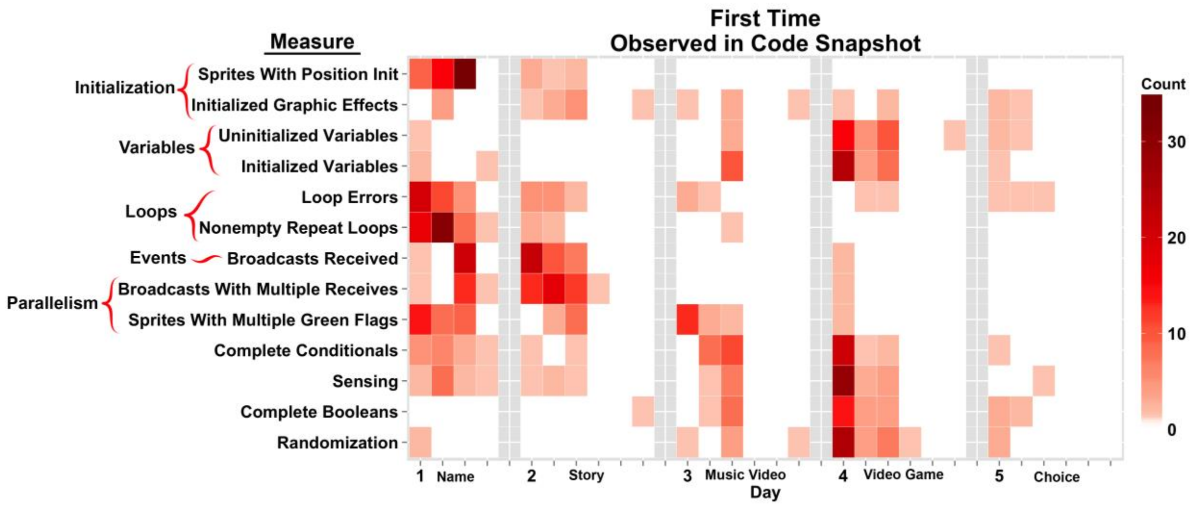


Figure 1. This figure shows the distribution of the first time campers used each programming concept. Each coloured block represents one programming session; there were about four sessions each day of the Scratch Camps. The darker the block, the more campers used a concept for the first time within that session. For instance, "Broadcasts Received", a measure of event-driven programming, shows which campers used a broadcast command under an event hat with a matching receive (i.e., a working broadcast). First time use of this measure was centered at the end of Day 1 (seen in the left-most column) and the beginning of Day 2. This visualization allows us to see the ways in which learning was spread out across the week-long camps. It also highlights when learning was focused on particular days, which tend to correlate with creative constraints used in projects on those days.

Introduction

Many efforts to broaden participation in computer science (e.g., Resnick et al, 2009) have drawn inspiration from Seymour Papert's ideas of constructionism. One of the keystone attributes of constructionist learning is that it is interest-driven, allowing users to instill personal interests in what is created, developing their own ill-formed problems as they pursue their desires, and providing for different epistemological approaches to problem solving (e.g., Papert, 1991; Turkle & Papert, 1990). An important issue to this interest-driven approach is equity in learning

When interest is the only driver, it seems that too often the rich get richer while the poor get poorer. In studies of online programming environments like MOOSE Crossing and Scratch, most participants tend to learn only basic programming skills while a small group of highly-motivated learners make substantial gains and use a greater variety of programming concepts in their projects (Bruckman, Edwards, Elliott, & Jensen, 2013; Fields, Giang & Kafai, 2014). In many local informal educational settings, educators have succeeded in engaging kids' interests in computer science but have struggled to support all kids in learning more challenging programming concepts. For instance, in two years at a Computer Clubhouse, very few youth ever began to use concepts such as variables, Booleans, and conditionals in their programs (Maloney et al, 2008). As Grover et al (2015) point out, these trends are not new. Over thirty years ago, researchers identified that in pure discovery conditions, children using Logo for more than 50 hours gained some basic knowledge of programming, but had difficulty in learning other important and fundamental programming concepts (Kurland & Pea, 1985). How can we promote rigorous learning while maintaining the student-centered, interest-driven approach that defines constructionism?

In this paper we share our first steps toward building a more rigorous and equitable constructionist learning environment by applying an art studio design model of pedagogy to computer science. Recently scholars have been empirically documenting the range of practices found in art and design education, articulating how such environments foster greater creativity and learning (see Sawyer, 2012; Hetland, Winner, Veenema & Sheridan, 2013). We designed a week-long Scratch camp for kids aged 10-13 using these studio design theories, with the goal that *all* kids would go deep into learning key programming concepts while still supporting their personal interests. Hosting three identically structured camps allowed us to repeatedly test this method with students of varying ages and interests. To evaluate kids' learning, we used an innovative approach applying quantitative and qualitative analysis to a combination of automated backend saves of campers' projects, observational data, and manual frontend saves of campers' projects.

In this paper our goals are: 1) to articulate our application of the studio model of design to learning programming in the Scratch Camps, 2) to investigate how widespread and deep campers' learning was, and 3) through this investigation to consider lessons learned that can be applied to future constructionist learning environment designs.

Scratch Camp Design

This project examined learning in a series of three, week-long Scratch summer camps for novice kids aged 10-13. Scratch is an online, visual computer programming environment for children developed by researchers at MIT and launched in 2007 (Resnick et al, 2009). Drawing on rich research of art teachers' practices, Hetland et al (2013) identify four main components of studio design: student work, demonstration lectures, design criticism, and exhibition. In their view, most class time should be devoted to students' project work, an emphasis shared in constructionist literature. In the Scratch camps we followed this model by devoting the majority of each of the four sessions per day (60-75 min each) to camper work time with snack breaks or lunch in between sessions. Below we introduce the theoretically motivated educational features of the camps as they relate to the studio design model, linking to theories and research while providing examples of how things worked in practice. In connecting theory to practice, readers will be able to evaluate the utility of each element as it relates to their own local learning environments.

The Anchor: Creatively Constrained Projects

Camps were structured around creatively constrained projects: open-ended tasks designed to encourage specific skill attainment (Sawyer, 2012; Hetland et al, 2013). In this model, students control the project's content and outcome, while navigating design constraints built to challenge skills within their discipline. In computer science education, single projects with creative constraints have been used to encourage the use of specific computing concepts or collaborative skills (e.g., Fields, Vasudevan & Kafai, 2015). Building on these prior models, we created a *series* of creatively constrained projects that progressively built up skills to support novice kids' learning. Our objective was to support *all* campers in learning challenging programming concepts within personally meaningful, interest-driven creations.

Project selection was thus critical to designing an effective studio model experience that would promote equitable learning. We planned four creatively constrained projects Monday through Thursday (see Table 1). Friday was open to finish whichever project campers wanted to feature for the culminating family invitational Gallery Walk. Drawing on work from Brennan and Resnick (2012) and our own prior research, (Fields et al, 2015) each project included design requirements that encouraged utilizing specific computing skills. For example, on the second day, campers faced the challenge of creating a Story project with the constraint that they include at least three different scenes, a restriction intended to elicit learning of concepts such as events and initialization.

Constructionism provides a powerful way to engage students by allowing them to follow their own interests as they create (e.g., Papert, 1991). Situating this interest-driven creating within constraints to promote learning, our goal was for campers to have a personal connection to their projects while developing challenging skills in order to accomplish their goals.

Project	Requirements	Computational Learning Goals
Name (Day 1)	Animate the letters of your username. Each letter should act differently.	Events: Green Flag Basic sequences Initialization Loops: Forever loops
Story (Day 2)	Create a story with at least three scenes and a magic effect.	Events: Broadcasts Sequence: Waits Parallelism with broadcasts or multiple green flags Initialization: Position, show/hide Loops: Repeat loops
Music Video (Day 3)	Create a 20-30 second music video with visual effects timed with the music.	Conditionals: Wait until Operators: Greater than Variables: Built in Scratch timer Parallelism: Multiple green flags
Video Game (Day 4)	Make a video game with a user interface, instruction screen, two or more levels, and an end game screen. Game can be a remix.	Conditionals Sensing User Interaction Variables: User-created variable Initialization: Variables

Table 1. Projects of Scratch Camp

Demonstration Lecture

The demonstration lecture is a brief, often visually rich lecture by a teacher to the class or small group (Hetland et al, 2013, p. 21). An opportunity to efficiently convey immediately relevant material, it is usually followed by student work, providing for "just in time" teaching. In Scratch Camp we applied this technique in a few different ways. First, each day began with a 15-20 minute demonstration lecture with sample projects kids would make (i.e. counsellors' own Name, Story, Music Video, and Video Game projects), and introductions to relevant programming ideas for the day's project. Second, each afternoon campers solved a series of small debug projects (Brennan, 2011), exposing them to reading code and introducing them to new skills. Third, we included spontaneous mini-demonstrations when campers were struggling with concepts. Counselors often found several students were struggling with the same idea and enlisted the attention and help of the entire camp to provide “just in time” learning opportunities. As our analysis shows, campers learned and applied skills at their own pace. Many iterations of presenting and teaching were needed before many campers implemented some concepts.

Design Criticism

In design criticism sessions, one or more students’ work-in-progress is shown to the entire class, and both students and teachers give external feedback and reflection (Sawyer, 2012). This peer review supports transparency by showing the teacher’s (or expert’s) viewpoint early and often, before projects are finished. This supports an emphasis on the process, not just the final product of creative work, and helps students learn to give constructive, positive critiques on others’ projects. Design criticism sessions also contribute to the community side of creating, showing the social side of individual projects as kids receive feedback and build a sense of pride over each others’ accomplishments.

Gallery Walks: Providing an Authentic Audience

The final aspect of studio design is audience and exhibition (Hetland et al, 2013). Artwork should be viewed, heard, and experienced by others. Engaging students in exhibiting their work and viewing others’ work is an authentic practice that has strong effects on both motivation and cognition (Magnifico, 2010). Beyond art, Kafai and Burke (2014) identify audience as an important component of computing, and propose a shift from an individualistic focus on computational thinking to a community emphasis on computational participation. This participation is especially relevant to constructionist projects that express their creators’ interests and personality (Bruckman, 1998). To reflect the importance of audience in the Scratch Camps, we included design criticism sessions (see above), supported kids’ roaming during work time, and held daily Gallery Walks. Gallery Walks took place during the last 10 minutes of each day, when all programming stopped and campers displayed their projects of the day. Campers and counselors wandered the room, leaving comments and compliments as they roamed. Knowing that they had an audience brought both accountability and purpose to campers’ creating. Gallery Walks also provided a high degree of transparency as campers saw and drew ideas from the many different ways their peers met the project constraints. At the end of the week, campers chose to perfect one or two projects to feature at the family invitational Gallery Walk.

Methods: Participants, Data, and Analysis

We ran three Scratch Camps throughout Summer 2014 at a land grant university in the intermountain west, partnering with the local 4H organization. Camp 1 included kids aged 10-11 (3 girls and 20 boys). Camp 2 welcomed kids aged 12-13 (7 girls and 12 boys), and in an effort to draw more girls Camp 3 was girls-only (25 girls) aged 10-13. A total of sixty-four campers (33 girls and 31 boys), ages 10-13, consented to participate in research. Campers were invited from the community and did not know one another before participating. Over 70% of campers were new to Scratch and programming in general; 15% had a few hours or a couple of weeks of experience, and 14% had more than two weeks of prior experience with Scratch. Camps were led by one director and four trained undergraduate and graduate counselors.

We collected three types of data: backend project data from the Scratch website, observational data from the camps, and the working (frontend) projects that campers created. With the help of the Scratch Team at MIT, we were able to collect the JSON (text-based) backend files of the campers project code, with file saves every two minutes, or every time campers shifted from editing backgrounds, sound, and costumes to coding. This resulted in 32,465 code snapshots in this text-based format (for more see Fields, Quirke, Amely, & Maughan, 2016). Observational data for this study came from two sources: field notes taken by a researcher attending all three camps, and video logs recorded at the end of each session as counselors reflected on and recounted events of the day. Field notes focused on the activities of five campers during each camp, and were helpful in providing context on kids' goals for their projects, their interactions with others, their struggles, and learning moments. Finally, we regularly saved campers' frontend projects in full Scratch format (vs. in text-based form) at the end of every session for the focal kids of each camp (5 each for Camps 1 and 2) and for all of the kids in Camp 3: 32 campers.

For this paper we conducted three primary types of analyses. Although we describe each individually here, the process was iterative and highly collaborative as we constantly compared findings from each analysis to inform the others. First, we developed and applied several quantitative "measures" of programming that identified campers' use of various programming concepts in their code. These measures were based on hand-analysis of campers' projects, developed to interpret their programming process through the backend data. More details on this process are reported in Fields et al (in press). Second, we coded the field notes and video observations to understand the social and emotional processes of campers' learning, looking for moments that defined excitement, frustration, and focus while considering the social situation at those moments. After initial analyses we focused on peer pedagogy and sharing as two important social practices where campers demonstrated changes in the ways that they took up those practices. Finally, using measures from backend data, observations, and projects, we created 32 case studies of campers' trajectories of programming, identifying the different pathways that kids took in navigating their project creation in the Scratch Camp. While the first two analyses show overarching trends in campers' learning of programming and social practices, the case studies provided information on how individual campers developed over the entire span of a Camp, setting learning and creating within a socially situated context.

Findings

Learning Programming

From this first look at kids' learning, what did campers learn and how widespread (or equal) was that learning? Further, did the creatively constrained projects allow enough room for personal expression? Using the measures we created to study the automated saves of campers' projects, we looked at the first time each camper used particular programming concepts (see Figure 1). From this we can see that certain programming concepts were used by all campers, especially in the areas of initialization, loops, events, parallelism, and conditionals[39]. The use of these concepts for the first time appears to match the timing of the projects intended to support that learning. For instance, every camper engaged in initializing the position of their sprites, a concept predominantly learned in the first two days of the camp, linked with the Name and Story projects. Similarly, all campers used broadcasts with matching receives in the Story project, and began using parallel forms of programming in both the Story and Music Video projects. Conditionals were largely introduced during the Music Video and Video Game projects.

One of our concerns was whether different pathways of learning were valued within the structure of the Scratch Camps. Interestingly, we found that while "first use" synced with particular projects, campers incorporated these concepts at different rates. Some kids began to use certain concepts very early, as soon as they were formally taught. Others did not begin to use them until the end of the day. This matched with findings from our case studies of campers' learning trajectories; despite being introduced to a concept multiple times, some campers simply did not appear ready to use it in their programs until well into their projects.

We also studied whether campers continued to use various programming concepts in later projects. Focusing on the last two days of the camps, we found that all campers continued to use initialization, loops, and conditionals consistently in their projects, often applying the concepts in different ways (e.g., using multiple forms of initialization). Further, campers continued to use events (87%) and parallelism (83% with multiple green flags in a sprite, 79% with multiple receives per broadcast). These findings suggest that while certain projects were useful for introducing kids

[39] While Figure 1 does not show the numbers, we counted to see how many kids used each kind of programming concept in their projects.

to programming concepts, once learned, most campers continued to implement them in other types of projects, demonstrating some robustness in their learning**.**

Evidence for campers' creative freedom is visible in the wide variety of topics and aesthetics in their projects, including stories such as visual narration of a Robert Frost poem or a scuba-diver-eating shark; music videos that ranged from a feminist re-enactment of *Star Wars* (with Princess Leia wielding a light saber) to wild animals singing in a jungle; and video games that required catching bloodied, zombie kitty heads falling from the sky or dodging frosted pink cupcakes. Not everyone was initially happy to be in a programming camp, and some kids expressed frustration at first, saying that they were artists or writers, not programmers. However, within a few days each of these youth demonstrated changed attitudes, finding ways to express their writing and artwork in their projects, demonstrating that despite some constraints, the projects allowed much personal expression. Many parents also reported (and our automated data support) that after six hours of programming at camp each day, kids went home and continued to program.

Lessons Learned

Remixing as Intentional Pedagogy: Room for Improvement

Learning was not always equitable in the camps. In particular we noticed a large difference in the complexity and quality of programming in the Video Game project. Remixing was an essential pedagogical strategy in the Video Game project. To introduce campers to concepts such as variables we provided four sample video games for them to cheat or hack at the code level and then, if they chose, to remix: shooter, dodge, catch, and maze genres. These starter projects helped focus campers on the kinds of games they could make in a limited number of hours in Scratch. Cheating these games led to playful manipulation of variables, "discovering" them by changing points given for health or score. Most of the games were simple and somewhat clunky in their designs with much room for improvement. Campers added new levels, increased challenges, edited artwork and themes, and built in instruction and game-over screens. About 60% of campers chose to remix the starter games; the remainder created original games. However, kids who chose to remix the "Maze" game tended not to use sensing or variables in sophisticated ways, if at all. Indeed, of the 25% of kids who did not create or program with user-created variables, most of them chose to remix the Maze Game. Why the inequity of learning with this particular starter game?

Analysing the changes campers made in their remixes, we discovered that the maze game did not require many changes to these features. It already had two levels, good user controls, and a built-in timer that created an appropriate level of challenge for players. In other words, there was no *need* to improve the game in ways that would support deeper learning of programming. A few kids who wanted to add enemies or rewards in the maze (akin to the Pacman genre of games) did delve into concepts of sensing and variables, but most kids stuck with aesthetic changes and additions of introduction and game-over screens. This suggests that the less well-developed starter games may have provided better fodder for learning by remixing, while the best-designed starter game instigated more shallow learning.

Indeed, our findings suggest that the more kids focused on improving the quality of gameplay, the deeper they went into challenging programming concepts such as conditionals, user interaction, variables and sensing. Some went beyond the material presented formally in the class to learn randomization and Booleans. These concepts allowed campers to provide more pleasure and surprise in the challenges their games presented to players (i.e., randomization), and in the smoothness of character interaction in the game (i.e., Booleans). The quality of the games therefore may have tied in closely with the kind of programming campers used. When that quality was already present, as in the Maze Game, campers did not need to learn how to program it themselves.

Learning Social Norms of Constructionism: Peer Pedagogy and Sharing Projects

While most of this paper has focused on campers' learning of specific programming concepts, learning social norms particular to a constructionist environment should not be taken for granted. In analysing observational notes we found two social practices that campers struggled to learn: peer pedagogy and sharing in-process projects. Both of these norms ran counter to traditional classroom culture where students are not supposed to give help because it is considered cheating, and where only the final product is relevant to achievement. However, in the Camps, peer pedagogy and sharing unfinished work were both expected of the kids. One of the few rules was that they should "ask two friends first" before they could ask a counsellor for help. During the first few days, many campers resisted this rule, saying that their neighbours were busy or finding another excuse. How did they get from this initial resistance to embracing the opportunity to ask for and give help to others? Counsellors used several strategies to take an active role in *brokering* peer pedagogy: asking for camper volunteers who knew how to solve a particular problem, referring knowledgeable campers to others, and applying a cascading model of learning where counselors would teach one camper how to solve a problem with the explicit expectation that this camper would then teach others.

Another area of challenge was sharing in-progress work, both in design criticism sessions and in Gallery Walks. Some campers felt vulnerable when sharing work with their peers and were resistant to sharing. When campers chose not to display their projects, they weakened the community-learning network by their non-participation. However, beginning each design criticism session with two compliments to the creator and making this a regular (hourly) habit established these new practices as legitimate. After a couple of days, many campers begged to have their projects shared. After four days, even the most resistant campers began to embrace the opportunity to share their projects with others when it was framed as contributing to the group's learning. This made the projects more than just individual achievements: everyone contributed to each others' learning and took pride in the progress their peers made.

Discussion

In this paper we have described the theoretical foundations of a studio model of pedagogy applied to computer science learning and investigated how well this provided an equitable approach to novice learners. Overall, the structure provided with creatively constrained projects did successfully promote all kids' learning of specific programming concepts while still allowing much room for personal interests. This approach provides more structure than in some constructionist environments and less than is often desired in a computing classroom. One surprise to us was the challenges campers faced in learning not only the programming but key social practices important to this environment: peer pedagogy and sharing in-progress work. The staff played a key role in brokering teaching roles to the campers, facilitating kids teaching kids and learning to provide and accept constructive criticism. In future research we recommend looking more closely at learning not only academic content but also social practices that undergird the core values of constructionist environments.

Using automated analyses to evaluate learning in open-ended projects can be a challenge (e.g., Blikstein et al, 2015). We approached this by developing measures of programming and looking at when and how often they were used in kids' programs. To understand what this meant we developed and compared the automated measures with qualitative analyses, acknowledging the highly contextual nature of our particular pedagogical goals and intervention. Visualizing learning patterns across a group of novices as in Figure 1 can help educators evaluate their learning environments and quickly identify which individuals might be missing out on key opportunities. In our camps, certain learning was well concentrated on particular days, tying in with some project goals (e.g., events, initialization). Other learning was spread throughout the camps (e.g., parallelism through multiple green flag use) and some was less equitably taken up than others (e.g., variables, sensing, Booleans, and randomization). This analysis allowed us to identify problems in our own pedagogy, specifically in how to better support learning by remixing less sophisticated projects. In other work we are exploring how to trace and compare individual trajectories (Fields et al, in press), which can provide a deeper contextual understanding of different learning pathways in programming.

References

Blikstein, P., Worsley, M., Piech, C., Sahami, M., Cooper, S., and Koller, D. (2014). Programming pluralism: Using learning analytics to detect patterns in the learning of computer programming. *Journal of the Learning Sciences.* 23(4), 561-599.

Brennan, K. (2011). *Debug It!* Retrieved on May 1, 2014 from http://scratched.gse.harvard.edu/resources/debug-it

Bruckman, A. (1998). Community support for constructionist learning. *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, 7, 47-86.

Bruckman, A., Edwards, E., Elliott, J., and Jensen, C. (2013, April). Uneven achievement in a constructionist learning environment. In *International Conference of the Learning Sciences: Facing the Challenges of Complex Real-world Settings* (Vol. 7, No. 17, p. 157). Psychology Press.

Fields, D. A., Giang, M., and Kafai, Y. (2014). *Programming in the wild: Trends in youth computational participation in the online Scratch community*. In Proceedings of the 9th Workshop in Primary and Secondary Computing Education, ACM, New York, NY, 2–11 .

Fields, D. A., Quirke, L., Amely, J. and Maughan, J. (2016). Combining "big data" and "thick data" analyses for understanding youth learning trajectories in a summer coding camp. In *Proceedings of the 47th ACM technical symposium on Computer science education* (SIGCSE '16). ACM, New York, NY, USA.

Fields, D., Vasudevan, V., and Kafai, Y. B. (2015). *The programmers' collective: fostering participatory culture by making music videos in a high school Scratch coding workshop*. Interactive Learning Environments, 1-21.

Grover, S., Pea, R. and Cooper, S. (2015). *Designing for deeper learning in a blended computer science course for middle school students*. Computer Science Education, 25(2), 199-237.

Hetland, L., Winner, E., Veenema, S., and Sheridan, K. (2013). *Studio thinking 2: The real benefits of visual arts education*. Teachers College Press, New York, NY.

Kafai, Y. B. and Burke, W. Q. (2014). *Connected Code: Children as the programmers, designers and makers for the 21st century*. MIT Press, Cambridge, MA.

Kurland, D. M. and Pea, R. D. (1985). *Children's mental models of recursive logo programs*. Journal of Educational Computing Research, 1, 235–243.

Magnifico, A. M. (2010). *Writing for whom? Cognition, motivation, and a writer's audience*. Educational Psychologist, 45(3), 167-184.

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M. and Rusk, N. (2008). *Programming by choice: Urban youth learning programming with Scratch*. ACM SIGCSE Bulletin, 40(1), 367–371.

Papert, S. (1991). *Situating constructionism*. In Constructionism, I. Harel and S. Papert, (eds). Ablex Publishing Corporation, Norwood, NJ, 1–11.

Peppler, K. A., and Kafai, Y. B. (2007). *From SuperGoo to Scratch: Exploring creative digital media production in informal learning*. Learning, Media and Technology, 32(2), 149-166.

Resnick, M., Maloney, J., Hernández, A. M., Rusk, N., Eastmond, E., Brennan, K., Millner, A. D., Rosenbaum, E., Silver, J., Silverman, B., and Kafai, Y. B. (2009). *Scratch: Programming for everyone*. Communications of the ACM, 52(11), 60–67.

Sawyer, R. K. *(*2012*). Learning how to create: Toward a learning sciences of art and design.* In The Future of Learning*:* Proceedings of the 10th International Conference of the Learning Sciences *(*ICLS 2012*)*, Volume 1, Full Papers, edited by J*.* van Aalst, K*.* Thompson, M.J*.* Jacobson, and P*.* Reimann*.* International Society of the Learning Sciences*:* Sydney, NSW, Australia, pp*.* 33-39*.*

Turkle, S. and Papert, S. (1990). *Epistemological pluralism: Styles and voices within the computer culture*. Signs, 128–57.

Communities of Learning Designers in Japan – From constructing products to constructing communities –

Yoshiro Miyata, *miyata@sist.chukyo-u.ac.jp*
School of Engineering, Chukyo University, Toyota, Japan
Nanako Ishido, *nanako@canvas.ws*
Graduate School of Media Design, Keio University, Tokyo, Japan
Kazuhiro Abe, *abee@si.aoyama.ac.jp*
School of Social Informatics, Aoyama Gakuin University, Tokyo, Japan
Mihoko Kamei, *kamei@sugiyama-u.ac.jp*
School of Culture-Information Studies, Sugiyama Jogakuen University, Nagoya, Japan

Abstract

In this panel, we will showcase some activities in which we have constructed communities of workshop designers in Japan in recent years. Through hundreds of thousand years of human history, construction of products has been supported by human relations in communities of people who designed, produced and used the products. It is only very recently that our products are designed to hide the production process and we have lost these relations. The panelists have been active in reconstructing communities in which designers of workshops can exchange ideas and find possibilities to collaborate so that production is supported by human relations. The contents of workshops range from hand crafts, creative arts, computer programming, to community building, and multi-cultural collaboration. In these workshops people with different experiences and skills collaborate to construct creative expressions that are meaningful to everyone involved. We have found out that in these communities in which production is supported by human relations, the constructed meanings gain larger perspectives that are beyond individual workshops and designers. The four panelists will share their experiences of constructing four of these communities of workshop designers that are closely related with each other: Workshop Collection, Programming Education Gathering, Aichi Workshop Gathering, and World Museum. They will then describe how constructionism in Japan has expanded from construction of products to construction of human relations, meanings, and to communities, and discuss about some principles underlying this expansion.

Figure 1.CANVAS Workshop Collection and Aichi Workshop Gathering

Keywords;

community of learners, creative expression, global collaboration

CANVAS Workshop Collection (Ishido)

Nanako Ishido established CANVAS in 2002, a non-profit organization to promote cooperation among schools, government and companies to construct creative learning environments for the children of the digital age. Since then, CANVAS has provided workshops for 350,000 children based on its belief that it is important for all children in the 21st century to learn to express themselves creatively and communicate effectively. Its annual event "CANVAS Workshop Collection" has grown now to gather 150 workshops attracting 100,000 participants. CANVAS has held many programming workshops since it started, and it initiated a new division "PEG" (Programming Education Gathering) in 2013 to expand its programming workshops to 25,000 children in its first year of activities.

Programming Education (Abe)

Kazuhiro Abe will talk about programming education based on Constructionism in Japan which he has pioneered. In Japanese schools, computer education tends to emphasize either using standard applications or becoming IT specialists. Abe has been influential in educational movements that emphasize creative expressions based on Constructionism advocated by Papert and learning through new style of expression and sharing promoted by Resnick and the Scratch Team at MIT Media Lab. He and PEG (described above) have helped many schools in Japan to use Scratch on Raspberry Pi in many different subjects, including Math, Science, Social Studies, Language, Music and Citizenship.

Aichi Workshop Gathering (Kamei)

Aichi workshop gathering is a community of people who organize activities or workshops in arts, crafts, and programming, in and around the Aichi area. To provide children and adults with a variety of activities, the community started to gather in May 2014 with the help from PEG (above). Since then, it held workshop-days in August in 2014 and 2015, each providing about 20 programs for 300 to 500 participants at Sugiyama Jogakuen University in Nagoya. Members of the community are educators, university students, museum curators and IT engineers. After the gathering in 2014, the members with such diverse backgrounds realized the value of supporting each other in the process of constructing the event together. So, they met several times before the event in 2015 to share concepts and ideas. It was quite useful resulting in some collaboration among programs as well as useful learning opportunities for the students involved. The students who have learned how to support or create workshops in the gathering will play important roles in developing the community especially as local citizens after they graduate.

World Museum Project (Miyata)

World Museum Project started in 2010 as a collaboration between a few schools in Japan and the US, and has grown into a network of educators in more than 30 countries around the world, who help children to collaboratively create artworks and learn from each other. Yoshiro Miyata has coordinated many projects in World Museum based on the Create/Connect/Open model he has developed for expanding the Constructive Mindset through global collaboration. It was found that through these collaboration projects, the participants expanded their constructive mindsets from constructing products, to constructing relations, and to constructing meaning and communities (Miyata 2012).

References

Miyata, Y., Ueshiba, T., Harada, Y., L. (2012) *Cultivating Constructive Mindset in World Museum, collaboration across cultures and generations*. In Proceedings of Constructionism 2012. August.

Computational Thinking in School: reviving programming in the context of digital culture

José Armando Valente, *jvalente@unicamp.br*
Dept of Midia Studies, State University of Campinas (UNICAMP)

Abstract

In the beginning of the 1980s, computer programming was one of the main activities related to the application of informatics in education**.** This was thanks to the use of the Logo language, created by Seymour Papert **(**1980**).**

With the arrival of tools for the workplace **(**text and drawing editors, spreadsheets, and presentation software**)**, programming was practically forgotten in elementary and secondary education**.** The use of these tools did not stimulate the development of the students**'** logical thinking, they did not contribute to the understanding of the specificities of how these technologies work, nor the concepts involved in programming**.** This knowledge has become fundamental within the context of digital culture and critical for life in the knowledge society**.**

The digital technologies can offer much more**.** In order to take advantage of the true benefits of digital culture, we will need to deepen our conception of these technologies and understand how they work, and how they can be adapted to the various contexts and situations in our daily lives**.** These concerns have brought educational policy makers to emphasize the widespread importance of programming and the ideas originating in Computer Science**.** Examples include the site, **"***Computer Science is for Everyone***!"** launched by the White House in 2013 (White House, 2013), and researchers like Rushkoff (2011) who advocate in his book that we should learn to program or we will be programmed.

This emphasis on the concepts used in Computer Science has been justified based on the argument that the activities in the realm of this science develop critical and computational thinking in the students. These activities reveal how to create digital technologies and not simply how to use them as instruments for the office. This knowledge is considered essential to prepare students for the 21st century, independent of the field of study or career path (Code, 2015).

This line of reasoning has promoted changes in the basic education curricula in various countries, where programming or Computer Science is being introduced even in the first years of primary education. For example, in Europe, the European Commission published a report, *European Schoolnet* (2014) based on a study of the current situation in 20 European countries. The report reveals that in 13 of these countries, including Estonia and Greece, programming has already been included as one of the required subjects in kindergarten through ninth grade (K-9, equivalent to our "Ensino Fundamental"). England changed the required subject of "Informatics" (known as ICT) which explored office tools, substituting it for "Computing", organized according to the triad of Computer Science, Information Technology, and Digital Literacy **(**Department for Education, 2013).

In the United States, this concern for the development of computational thinking has not promoted changes in the basic education curriculum (K-12). Initiatives of this type have come from companies and nonprofit organizations including Code.org, the College Board, the National Math and Science Initiative, Teach for America, and the Project Lead the Way, which are involved in activities around the production of curriculum materials or the promotion of professional development courses for teachers or the general community interested in the field of Computer Science (White House, 2013). For example, Code.org (Code, 2015) was founded in 2013 and has made a significant contribution to the awareness of the lack of Computer Science as a subject in basic education in the USA.

The emphasis on Computer Science, specifically on programming, has been criticized by various researchers, including those from this same field. Hemmendinger (2010) views the proposal for computational thinking linked to Computer Science as an expression of arrogance, suggesting that the computer scientists are trying to tell everyone else how they should think. He concludes, "Computer scientists can contribute, but we should be careful not to speak as if we are the only ones to lead people to a promised land. In the end, though, perhaps we should talk less about computational *thinking* and focus more on computation *doing*" (Hemmendinger, 2010, p.6).

Hemmendinger's proposal is extremely pertinent, considering that programming is not restricted to generating code. Many of the ideas originating from traditional programming can be explored in other activities using digital technologies, such as digital narratives, the creation of games, development of animations, etc. In addition, these ideas can be explored through activities that do not use the computer (Bell et al, 2009). Finally, a study of teachers from Basic Education showed that many of the traditional classroom activities, with or without digital technologies, are related to activities similar to programming, such as, selection of relevant information, identification of patterns, organization of data in graphs and charts (Manilla et al, 2014).

It is true that these ideas about computational thinking are impacting educational policy decisions in many countries, placing us as scientists involved in this area in a position to discuss and understand:

a) What is computation thinking?
b) Can computational thinking only be developed through programming?
c) What is the relation between what is being proposed as computational thinking and the powerful ideas of Papert? Are we just switching terminology?

References

Bell, T. et al (2009). *Computer Science Unplugged: School students doing real computing without computers.* The New Zealand Journal of Applied Computing and Information Technology, 13(1), pp. 20-29.

Code. (2015). Site of the Code.org organization.

Department for Education. (2013). *The national curriculum in England: framework document.* London: DfE.

European Schoolnet. (2014). *Computing our future: Computer programming and coding. Priorities, school curricula and initiatives across Europe*, October 2014.

Hemmendinger, D. (2010). *A plea for modesty.* ACM Inroads, 1(2), p.4-7, June 2010.

Mannila, L. et al (2014). *Computational Thinking in K-9 Education***.** *ITiCSE-WGR'14*, Proceedings of the Working Group Reports of the 2014 on Innovation & Technology in Computer Science Education Conference. June 21-25, 2014, Uppsala, Sweden, p. 1-29.

Papert, S. (1980) *Mindstorms. Children, computer and powerful ideas*. New York: Basic Books.

Royal Society. (2012). *Shut down or restart? The way forward for computing in UK Schools*. 2012.

Rushkoff, D. *Program or be programmed*: *ten commands for the digital age.* Berkeley: Soft Skull Press, 2011.

White House (2013) Computer Science is for Everyone! 2013.

Constructionism and the Internet of Things

Paulo Blikstein, *paulob@stanford.edu*
Stanford University

Kipp Bradford, *kb@kippworks.com*
Massachusetts Institute of Technology

Taylor Martin, *helen.taylor.martin@gmail.com*
Utah State University

Arnan Sipitakiat, *arnans@eng.com.ac.th*
Chiang Mai University

Nalin Tutiyaphuengprasert, *nalin.tt19@gmail.com*
Darunsikkhalai School for Innovative Learning

Marcelo Worsley, *worsley@usc.edu*
University of Southern California

Abstract

The intent of this panel is to foster discussion around the role that the Internet of Things has on current and future work in Constructionism, and the role that Constructionism should have in shaping the Internet of Things. We will invite researchers who are conducting scholarship at the intersection of these two areas, in order to highlight some of the forthcoming opportunities and challenges that are likely to emerge.

Figure 1. Gogo board meets raspberry pi meets data mining meets sensor networks

Keywords

Internet of things, networked devices, sensors, connected learning, multimodal analysis.

Panel Description

Over the past few years, the buzz over the Internet of Things has been steadily increasing. The world has begun to see what it means to tap into the power of "smart" and connected machines. These machines span the gambit from at home thermostat systems, to automobiles. A critical question, however, is what the possibilities of the Internet of Things means for present and future work in Constructionism, as well as what Constructionism means for the Internet of Things. In this panel researchers and practitioners will share examples and ideas for how the Internet of Things introduces new opportunities for constructionist learning. In particular, the panel will bring together scholars that will examine four themes.

Theme 1: Students and teachers developing connected/networked devices.

Developing devices using sensors and actuators has been a central theme of constructionist learning experiences. There is a long history of microcontrollers that include notable devices like the Lego Mindstorm, Makey Makey, the Gogo Board and the PICO Cricket. More recent work, however, has highlighted the ways that these microcontrollers can easily be coupled with Internet enabled devices (the Raspberry Pi, for example) to create complex sensor networks that control any number of devices. These devices are further enhanced by the ongoing development of platforms like the Gogo Board, which allow a microcontroller to control a large host of externally powered machines. Quite simply, then, as we make it easier for students and teachers to develop increasingly complex devices, we introduce new, and increasingly complicated "objects to think with." Moreover, we raise the ceiling on what students and teachers can accomplish, and continue to foster the development of increasingly "Big Ideas." Arnan "Roger" Sipitakiat and Kipp Bradford will share their expertise on creating such devices and consider how these tools can transform and broaden learning.

Theme 2: Incorporating data science into constructionist learning

As noted above, one opportunity is for students and teachers to create sensor networks that are controlled through cause and effect types of interactions. Another, opportunity, is for learners to engage in Data Science through the collection of custom data. Whereas, there is a host of readily available data on the Internet, learners should have the opportunity to gather data that they "own" and that answers local, or global problems that are significant to them. With devices that contain wireless capabilities, learners have the ability to collect data, even in locale that they may otherwise not have continuous access to. For example, the CubeSat project, which originated at DSIL, highlights ways for collecting data and images from near outer space conditions, something that would not have previously been possible for anyone that didn't have access to an industrial satellite. At the panel, Paulo Blikstein and Nalin Tutiyaphuengprasert will share their experiences and goals from two Data Driven workshops with teachers from DSIL.

Theme 3: Developing multimodal connected learning experiences

While we all recognize that learning takes place across a large variety of contexts, learning is typically only documented and validated as it takes place within formal learning settings. While elements of this are changing, as different organizations develop badges and other systems to acknowledge student development, there are still very few connections among the vast array of locations in which individuals experience learning. One of the opportunities that arises with connected devices is the capability to develop a more holistic chronicling of learners' experiences. This documentation need not be wholly based on sensory data, but can also be combined with learner self-reflection. Furthermore, as the different learning contexts begin to "speak" to one another, there is an opportunity to bring forth a new level of learning continuity. As we develop better ways to trace students' learning experiences, across context, we also move towards the development of enabling better distributed learning experiences. Marcelo Worsley, who is currently working with Matthew Berland, Mike Tissenbaum, and Erica Halverson will discuss their work towards expanding the capabilities and practices of the Connected Spaces Framework. This framework aims to foster better collaboration across distributed learning sites. The work also aims to construct a representation of each student based on their experiences across a variety of spaces.

Theme 4: Leveraging multimodal sensory networks to study learning in constructionist learning environments.

Constructionist learning involves a range of activities, tools and experiences. One opportunity, then, is to use multimodal sensor technology to better understand the affordances of the different practices, tools and experiences, such that we can better support learning. Moreover, with the various critiques raised about constructionist learning environments, using high precision multimodal sensory data may get us closer to quantitatively demonstrating why these learning environments are of significant value. Being able to reliably make this claim is becoming increasingly important as organizations around the world adopt constructionist learning approaches and need to effectively train teachers and practitioners how it can be used to promote meaningful learning. Taylor Martin and Marcelo Worsley will discuss their prior work on analyzing data that includes everything from facial expressions to student generated computer programs, as a means to better hone in on what works well, why and for whom. They will also discuss forthcoming work that is aimed at advancing our ability to do this work.

The four topics listed above are meant to serve as an entry point into discussion with participants around potential opportunities for incorporating the Internet of Things into constructionist learning experiences. Elements of the discussions may also touch on the ways that the Internet of Things may be a hindrance to Constructionism, forthcoming challenges, and additional themes not listed in this proposal application.

Darunsikkhalai School in Bangkok and its 15 years improvisation on Samba School

Nalin Tutiyaphuengprasert, *nalin.tt19@gmail.com*
Darunsikkhalai School for Innovative Learning

Yuphin Trangkatarn
Darunsikkhalai School for Innovative Learning

Darunsikkhalai School for Innovative Learning (DSIL) was established in Bangkok, Thailand in 2001 by King Mongkut's University of Technology Thonburi and Suksaphattana Foundation with the support of Prof.Seymour Papert and his colleagues from the Future of Learning group at the MIT Media Lab. From the beginning, all the young generation of teachers were enrolled and LOGO workshops were provided for them. The workshops provided an inspiring learning experience and powerful ideas to many first generation teachers and motivated them to explore and develop new possible learning environments in order to bring better learning experiences to their students.

Starting with the inspiring ideas of the "Logo Environment", from day one, the DSIL community questioned themselves about the changes needed to create a school for "learning" which is not just for "schooling". Many generations of leaders as well as teachers, students and parents learned together through real life practices. DSIL encountered many problems and challenges during the transformation of the new school. The school has been growing and a new mindset of "learning" has been constructed by the people involved.

Yupin Trangkatarn and Nalin Tutiyaphuengpraset will share their perspectives as the first generation of school administrators and facilitators who have gone through the transforming processes in the past 14 years. Yupin will share her experiences as a school director at a traditional school in Lampang province on how a conventional school leader would drive the change starting from scratch as compared to her existing experience as the leader in a "Constructionist" school like DSIL. Leadership played an important role in building and orchestrating the new learning context to allow it to emerge, stumble, grow and flourish.

Three different groups of teachers will present the existing classroom management and strategies that they are using in combination with the Constructionism learning process to help students achieve learning outcomes in each level. Teachers acted as "anthropologists" and observed and learned how students at each age level interact with each other in the learning environment. They then redesigned classroom strategy to support effective Constructionist learning in their classroom.

Pintippa Sangwan and Rossarin Sookpraderm will present a learning model for early learners (first graders and second graders) at DSIL where they integrated the Waldorf learning philosophy with Constructionism. This model was started in 2013 and it helped support young learners to be responsible for themselves, and to respect and collaborate effectively with others in harmony.

Titima Kitcharoen will share her observation and classroom strategies with third and fourth graders. The Constructionist learning environment empowered students to learn very well. However, her challenge, interestingly, is to manage all the diverse and powerful ideas of her many young students.

Ittichai Rattanatavorn will present a model for early advanced learners. (7th – 9th graders). In order to deal with the diversity of project interests, using the Constructionist learning environment as the backbone, he has applied problem-based learning and research processes as a platform to help students identify and organize their own project goals and execution plans.

The panelists will include:

1. Yuphin Trangkatarn (Provost)
2. Nalin Tutiyaphuengprasert (Vice Provost)
3. Pintippa Sangwan (Facilitator)
4. Rossarin Sookpraderm (Facilitator)
5. Titima Kitcharoen (Facilitator)
6. Ittichai Rattanatavorn (Facilitator)

Making Constructionism Read in Real Schools Every Day

Gary Stager
CEO: Constructing Modern Knowledge

Amy Dugré
Director of Technology: The Willows Community School

Brian C. Smith
Technology Integrator: Hong Kong International School

Tracy Rudzitis
Teacher and Stanford FabLab Fellow: NYC Public Schools (The Computer School)

Abstract

The panellists work in real schools in New York, Los Angeles, and Hong Kong to make constructionism come alive on a daily basis. This panel will explore their triumphs, failures, and lessons learned for moving an entire school in a constructionist direction and sustaining such progress. Each of the participants teaches programming, robotics, mentors colleagues, develops curricula, and is active in the maker movement. This work extends beyond programming and robotics to making the rest of the school day reflect approaches to constructionism. In addition to their day-to-day work of inspiring colleagues, creating productive contexts for constructionist learning, and leadership, each of the participants has spent many years involved in leading Constructing Modern Knowledge, the summer institute for educators built-upon the principles of constructionism.

What would the ideal constructionist programming language (or languages) be like?

Ken Kahn, *toontalk@gmail.com*
Academic IT Services, University of Oxford

Abstract

In a discussion about future programming languages with Seymour Papert, he asked the question whether their design should be driven by mathematical aesthetics or engineering. He thought that most current efforts come from engineering criteria and was uncertain if that was good or bad.

When designing a constructionist programming language one must make choices about the underlying computational model, the syntax, the programming environment, and the extent to which it is compatible with existing languages and systems.

What are the ultimate goals of a new programming language?

Should we strive for a single ideal language or would multiple languages be best?

(Kahn, 2007) brings up many of these questions.

Logo, and more recently Scratch, has played a foundational role in constructionist learning. Could something other than incremental progress lead to much better tools for using computers in a constructionist manner? What lessons can be learned from existing languages such as Logo, Scratch, Snap!, NetLogo, ToonTalk, etc.?

A panel consisting of people involved in programming language design and research would be ideal. I suggest Cynthia, Brian, Ivan, Uri, and myself.

Keywords

constructionist programming languages, computational models, programming language design

References

Kahn, K. (2007) *Should LOGO Keep Going FORWARD 1?* Proceedings of the 2007 EuroLogo Conference. Bratislava. August.

A New Robot in a Classroom

Pavel Petrovič, *ppetrovic@acm.org*
Dept of Applied Informatics, Comenius University, Bratislava

We engaged in developing a set of 20 different activities with LEGO MINDSTORMS EV3 aimed at how to use the robots in a classroom on various subjects. As contrasted to most of the activities, work-sheets and courses that are based on this technology, we are not interested in teaching introduction to robotics and control, or teaching programming with robots. In this project, we would like to hand over a set of project ideas with solutions to teachers in Physics, Mathematics, Art, Chemistry and even Informatics. Each idea can be either 1) adopted to enhance the teaching, for instance when demonstrating a phenomenon in a classroom, or 2) given as a project assignment to a group of motivated students who will then demonstrate their results to a whole classroom, or 3) used in hands-on classroom activities, provided the teacher has access to multiple robotic sets. We have written a summarizing workbook in Czech language, but the tasks are available on-line in English as well as part of the Centrobot portal with robtivities.

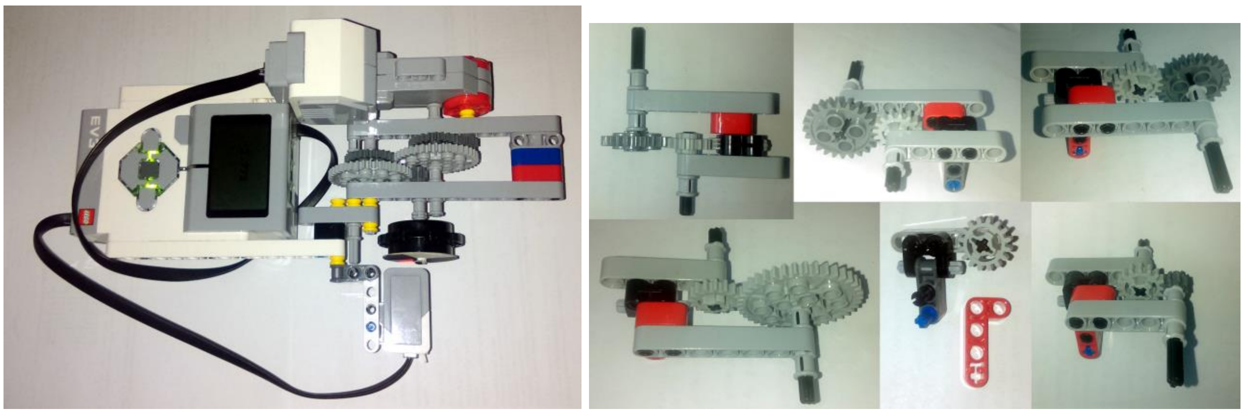


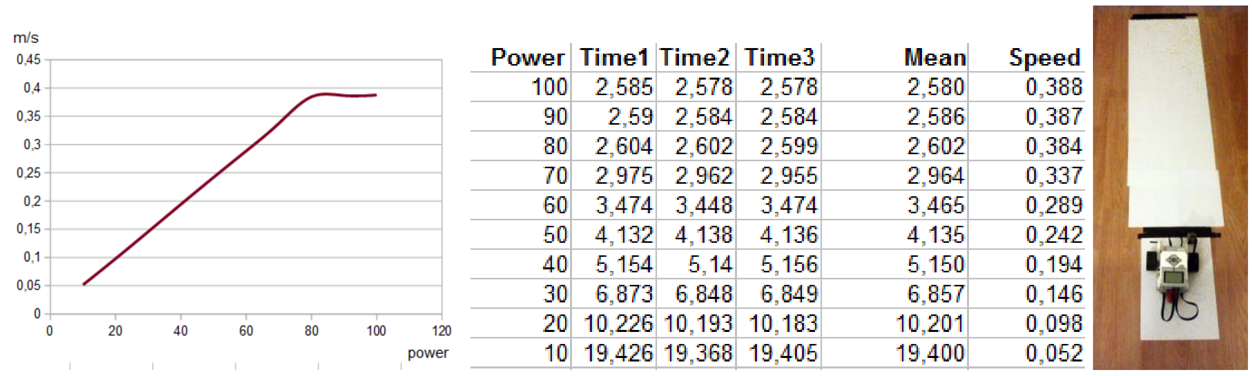
Figure 1*:* Teaching fractions on mathematics class with various LEGO gear ratios, counting the rotations with the light sensor *(*left*).* Using the 16-teeth gear requires connecting the structure in half-unit distance, which can be achieved in several different ways *(*right*).*

Keywords

EV3, centrobot, robtivity, teaching with robots

Centrobot portal

During a project within the framework of Slovak-Austrian cross-border cooperation programme in the years 2007-2013 'Creating the Future' financed by the EU European Regional Development Fund, several wonderful ideas have come to flourish. Viennese Happy Lab – a public, open hardware workshop that has later been transformed into FABLab has grown rapidly, RobotChallenge – today definitely the largest and most renowned robotics competition in a region has been supported to grow to its contemporary scale of hundreds of robot registrations every year, and Centrobot (Balogh et.al, 2010), a portal for robtivies, i.e. robotic activities for classroom, club, and home has been designed, and a prototype developed, later superseded twice with a new version. It builds on an already established base of robotics activities with NXT robots (Petrovič, 2007-2015), and other platforms (such as Acrob and Robotnacka), and is currently being updated with EV3 projects. This portal serves as home platform where activities in this work will also be accommodated.



Power	Time1	Time2	Time3	Mean	Speed
100	2,585	2,578	2,578	2,580	0,388
90	2,59	2,584	2,584	2,586	0,387
80	2,604	2,602	2,599	2,602	0,384
70	2,975	2,962	2,955	2,964	0,337
60	3,474	3,448	3,474	3,465	0,289
50	4,132	4,138	4,136	4,135	0,242
40	5,154	5,14	5,156	5,150	0,194
30	6,873	6,848	6,849	6,857	0,146
20	10,226	10,193	10,183	10,201	0,098
10	19,426	19,368	19,405	19,400	0,052

Figure 2: Math: investigating the relation between the power output and the resulting speed.

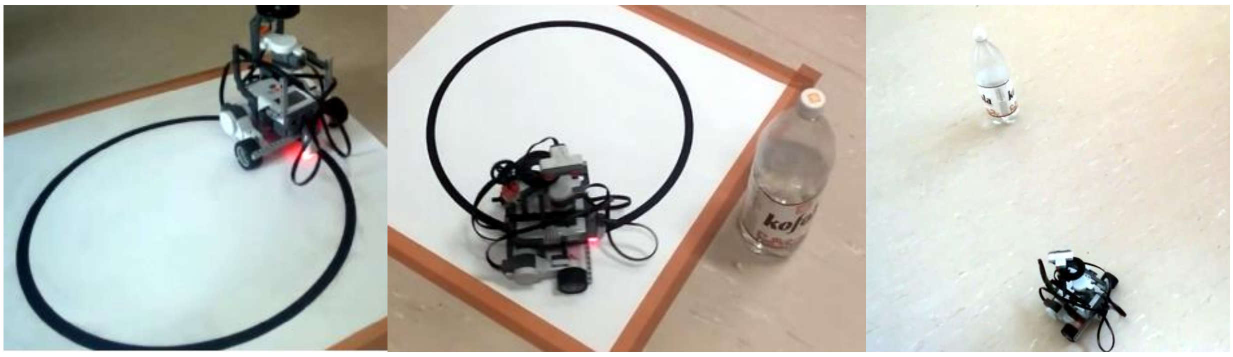


Figure 3: Computing circle diameter: using Tales circle (left), using circumference (middle), measuring width as the altitude in a general triangle with all sides known (right). NXT-based robots were used here.

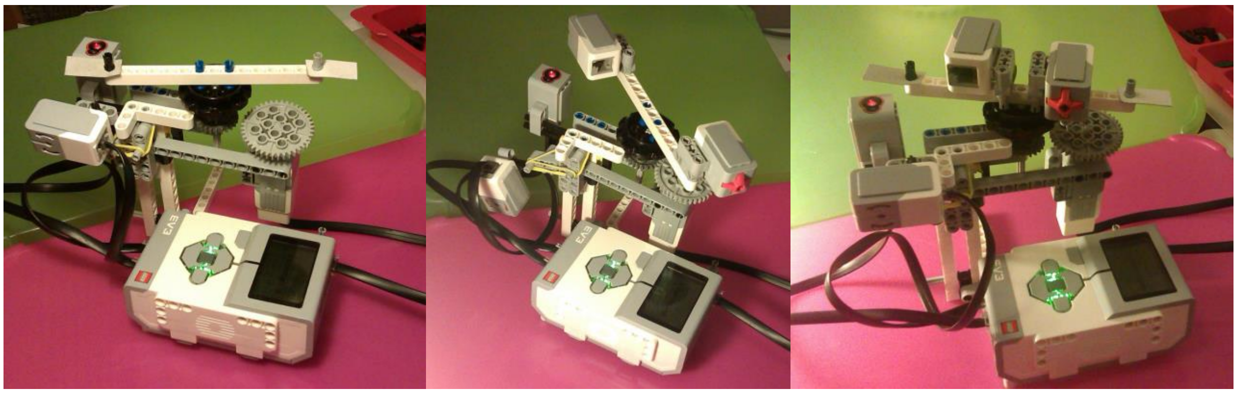


Figure 4: A physics project on rotational inertia momentum: weight distribution influences kinetic energy.

Topics

- The set of exercises currently covers various topics from mathematics and physics. Mathematics themes include usual Pythagorean Theorem and triangles, fractions using gears (Figure 1), plotting power-speed function in a graph (Figure 2), determining circle diameter and measuring distances in various ways (Figure 3), and others. Physics projects include understanding the concept of average speed, acceleration, rotational momentum of inertia (Figure 4), and other. We are constantly working on improving the projects and inspiring teachers to use the robots in teaching.

References

Balogh, R. et al. (2010) *Centrobot Portal for Robotics Educational Course Material*, in Proceedings to Robotics in Education.

Petrovič, P. (2007-2015) *Using LEGO Mindstorms in Classroom*, available on-line: robotika.sk/nxt

Petrovič, P. (2015) *Centrobot Portal*, available on-line: portal.centrobot.eu.

A practical report on a course of learning by “making” at the university in Japan

Aoi Yoshida, *aoi@si.aoyama.ac.jp*

Kazunari Ito, *kaz@si.aoyama.ac.jp*

Kazuhiro Abe, *abee@si.aoyama.ac.jp*
School of Social Informatics, Aoyama Gakuin University

Abstract

At Aoyama Gakuin University in Japan, we offered a course titled “Hands-on Practice in Social Informatics.” We employed a method of learning by “making,” prepared two slogans, and used a wide range of gadgets. This poster describes the course content and students' final products.

Keywords

Physical computing, Maker, Scratch, Practical report

Introduction

“Maker’s movement” has come, and an increasing number of universities in Japan have been introducing fabrication tools in class. While previous studies have found that physical computing can be effectively incorporated into K-12 programming education, we believe that it can also contribute to university education. At Aoyama Gakuin University in Japan, we offered a course titled “Hands-on Practice in Social Informatics.” A major objective of this course is to teach basic study skills at the university level, such as programming skills and ability to learn proactively. For this purpose, we employed a method of learning by “making.” We prepared two slogans in employing this method: "We (lecturers) do not give you (students) any instruction." and “Let’s become a maker. Let’s make new things from your ideas.”

Practice outline

Participants

This course was offered in four sections in the first (April-August) semester in 2015. There were about 60 students in each section, and thus, about 240 students are enrolled in this course in total. Most of them had never done programming before since they were freshman in our multidisciplinary department.

Software and hardware

We used Scratch, a programming language developed at MIT Media lab (https://scratch.mit.edu/). In addition to Scratch, many attractive gadgets were prepared as listed below. All of them are compatible with Scratch and Scratch-based programming tools. Those gadgets were intended to stir students’ imagination for creation and capture their hearts and mind.

- Nanoboard AG, a sensor board for Scratch based on Aruduino (http://tiisai.dip.jp/?page_id=935)
- Studuino, a micro computer board based on Aruduino (http://www.artec-kk.co.jp/studuino/)
- Rolling spider, a mini drone (http://www.parrot.com/jp/products/rolling-spider/)
- Leap Motion, a gesture sensor (http://www.leapmotion.com/)
- Pocket Miku, a speech synthesis software based on NSX-39 (http://otonanokagaku.net/nsx39/)
- Raspberry Pi, a single-board computer (https://www.raspberrypi.org/)
- Motors, LEDs, LEGO and Artec blocks

Course content

Week	Class*	Content	Activities
1	1	Basic of Scratch programming	- Self-study on a text and websites
	2	Basic of Scratch programming for using sensor or web camera	- Self-study on websites - Write a manual about how to program
2	3	Basic of Scratch programming for using sensor or web camera	- Read and review the manual written by other students
	4	Making (Group work)**	- Think of creative ideas based on a given theme*** - Select gadgets to implement ideas
3	5	Making (Group work)	- Develop a product
	6	Interactive presentation	- Show-and-tell each other - Invest virtual money for favorite products in crowd-funding style

* There were 2 consecutive classes in a week, and each class was 90 minutes.
** There were 4 students in a group.
*** Themes, such as “an idea that makes the Olympics Games 2020 in Tokyo successful” and “a useful product for our lives.”

Table 1. The course content and activities

Students’ final products

All teams could develop and present their products, and most of them could write a program using some sensors and actuators. The videos of this course and students’ final products are available at https://youtu.be/qqf5V3VJYMQ and https://goo.gl/XYE7Mx. (Only in Japanese. English version is to be prepared.)

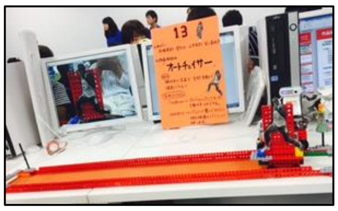
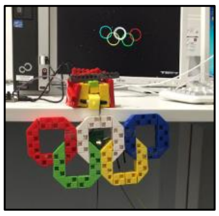



Figure 1. Examples of students’ final products
(Left) Auto-chaser of runners using an infrared photo reflector (Right) Interactive flag hoisting device

After all classes, we conducted course evaluations using a list of questionnaires. About 96% of students were satisfied with this course. More than 80% of students became interested in “making” and programming. This course involved 2 lecturers and 8 assistants. All of us tried not to give participants any instruction. When participants asked for instruction, we tried to give them some clue only. As a result, many students wrote, in a free writing section in the questionnaires, that they thought deeper and participated in the class more proactively than usual, because lecturers didn’t give them a direct answer to their questions. Moreover, some students wrote that they could acquire programming skill by participating in the course. On the other hand, some of unsatisfied students wrote that they wanted to receive some instruction about the basic of programming. For future work, we plan to improve the effectiveness of the course content. We also aim to explore how instructors should evaluate students’ programming skills.

References

Sylvia Libow Martinez Gary Stager, Ph.D. (2013) *Invent to Learn: Making, Tinkering, and Engineering in the classroom*. Constructing Modern Knowledge Press.

After Scratch: Logo(Writer)?

Mícheál Ó Dúill (Mike Doyle), *logios.org@googlemail.com*
British School of Sofia

The MIT Constructionists have made two forays into the education arena. Both were aimed at upper primary and lower secondary school. No consideration was given to the youngest children (five upwards) or to the imperatives of the curriculum (cf. the later Furber report). Their curriculum, crystallised in the late nineteenth century, has a core of literacy and numeracy.

The first foray was ostensibly to introduce programming through the educationally oriented language Logo. With a mathophilic tunnel vision peculiar to computer scientists, their first mindstorm was to introduce a novel mathematics, Turtle geometry, as an “easy” way to introduce programming to young children. The Turtle commands: fd, bk, lt (degrees), rt (degrees) had all the simplicity of binary arithmetic. The shapes drawn: squares, triangles, circles and variants were, on the surface, easy for children – after all they used them in their earliest drawings. This face-validity soon came under classroom scrutiny, where it was challenged by experienced teachers. The Constructionist response was to castigate teachers as luddites. Moreover, few schools actually used Logo because it was easier and less expensive to buy a turtle graphics package written in, e.g., BASIC. The net result, confirmed by Trading Standards in England was that Logo and Turtle geometry were conflated. Logo was little used in schools, and certainly not with small children.

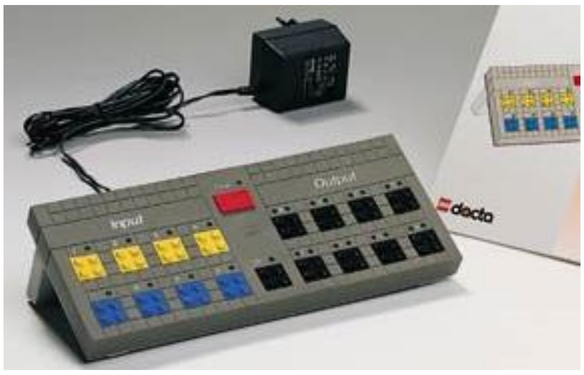
Into this febrile environment entered LogoWriter and LEGO Dacta Control Lab Logo. The former, which addressed written language and had low resolution graphics, was not well received by the comuterists. As a primary school teacher I was delighted by it, as were well-informed colleagues; indeed I briefly sold it in the UK. Control Lab, sans Turtle, was aimed at secondary school and equipped with the gadgets like buttons, sliders and assorted boxes that later became commonplace. Ilieva, in Bulgaria, boldly used it in primary school to animate LEGO models with computer-controllable elements. I use this approach. It is educationally far more powerful than current LEGO Mindstorms and WeDo based ‘success’ products that work in the same way that LOGO sets do: follow the instructions and it will work. A purely perceptual approach, it does not engage cognition. This is the educational legacy of the other MIT mindstorm: the programmable brick.

Having got wrong about everything there is to get wrong in school, and as the Turtle killed Logo, MIT moved into computer clubs. At least there a splinter activity was unproblematic. Also, enthusiasm having waned in the developed world, attention was displaced into the third world and the realm of disadvantaged children. The result was Scratch (again targeted at pre-teens). A cat replaced the Turtle with clothes rather than shapes. A “jigsaw block” metaphor replaced traditional text. Social networking and teach-yourself was the culture. This was fine. Then, following the Furber report, it came to school to be taught either by untrained teachers or IT professionals. As with the Turtle and the LEGO products, there is a disjunction from the curriculum and a perceptual (concrete) rather than cognitive (formal) level of thought. For instance, motor commands are “motor this way” and “motor that way” when it is the direction of the current that is being changed. Scratch, like the LabView based LEGO products, denies children access to some very basic science.

Swaddled in a protective philosophy, Constructionists are immune from classroom reality. Had the nature of LogoWriter’s complement to the curriculum and its potential for catalysing a transition in education been comprehended, children might now be beginning the exercise their budding literacy and numeracy skills in an active and supportive environment.

Logo(Writer) some illustrations from Control Lab Logo

My favourite illustration of the failings of the MIT (and Tufts) approach is nicely illustrated by the LEGO lamps for the 9v system. Lamps are particularly suitable because they are found in many realistic settings – and light is fundamental to life. The computer interface and lamp are shown below.



The original lamp is filament but with LEDs appearing in homes, I converted some to LED (the 3mm white version with 470 ohm resistor is a direct replacement).

Note the small LEDs above the output contacts. They light up green to indicate the positive terminal. The commands 'setleft' and 'setright' change the position of the positive terminal = like turning a battery round. With a filament lamp this has no effect but with the diode it does. In the documentation these commands are not associated with polarity, rather motor direction. In Logo I can dispel any uncertainty by defining procedures 'setplusleft' and 'setplusright' which execute the primitive commands. I can treat the 'rd' (reverse direction) command in terms of polarity, which makes it applicable to a sound element. Thus, in Control Lab Logo I can move from the perceptual to the cognitive level in a manner accessible to the youngest. In other words, *I can teach with this software.*

This Christmas the whole school created a winter scene. The third grade made alpine houses with a filament lamp inside. We wanted to simulate a flickering candle or fire. The following procedure did the job:

```
Talkto “lampa on loop [setpower random 3 wait 1]
```

Every part of this was comprehensible to the children, including the punctuation (we talked of the double quote as 'rabbit ears'). It relates to the English they are learning and introduces the notions of randomness and time intervals. Eight outputs offered eight different variants. However we used only one because the effect was OK (we used the lamp with a single top light that has a built in flasher unit) and cabling eight would have been visually intrusive.

As a primary-school teacher, I don't want teach-yourself software or step-by-step on-screen instructions. My kids are not permanently chained to the internet – working in teams, they have local peers. I want a flexible classroom resource. **Logo**(Writer) is it.

Bots for Tots: Leveraging 'Ways of Knowing' to Increase Diversity in Makerspaces

Nathan Holbert, *holbert@tc.columbia.edu*
Department of Mathematics, Science, and Technology, Teachers College, Columbia University

Abstract

Projects designed to give children experiences playing and building with high tech equipment such as 3D printers, laser cutters, and microcontrollers have gained momentum in recent years among researchers, educators, and parents. Despite an explicit commitment to epistemic diversity, *makerspaces* have struggled to serve a diverse population of creators and have become heavily dominated by men and the highly educated and wealthy (Moilanen, 2012). The Bots for Tots project is an effort to move beyond surface level participant characteristics (such as girls like fashion) and to instead explore the affordances of activity framings and structures that tap into alternate mental dispositions and ways of knowing to broaden participation and interest in maker activities.

The Bots for Tots project engages elementary children to design and build a "dream toy" for younger children in their community. Workshop sessions are designed to engage participants in interviewing stakeholders, brainstorming and critiquing, prototyping, and construction. In a pilot study involving 8 girls and 2 boys, dream toys were constructed using a variety of methods, such as sewing, laser cutting, and 3D printing as well as materials such as fabric, cloth, wood, acrylic, and extruded plastic. While data collection is ongoing, early findings suggest this activity framing may be fruitful as participants drawn to the project were overwhelmingly female, were highly interested in technology and making, and had some experience engaging in craft activities. Further analysis will evaluate the materials and techniques used by participants, how mixed and same gendered teams interacted, STEM content encountered by participants, and the degree to which framing the activity as being about making for others impacted day to day activities.

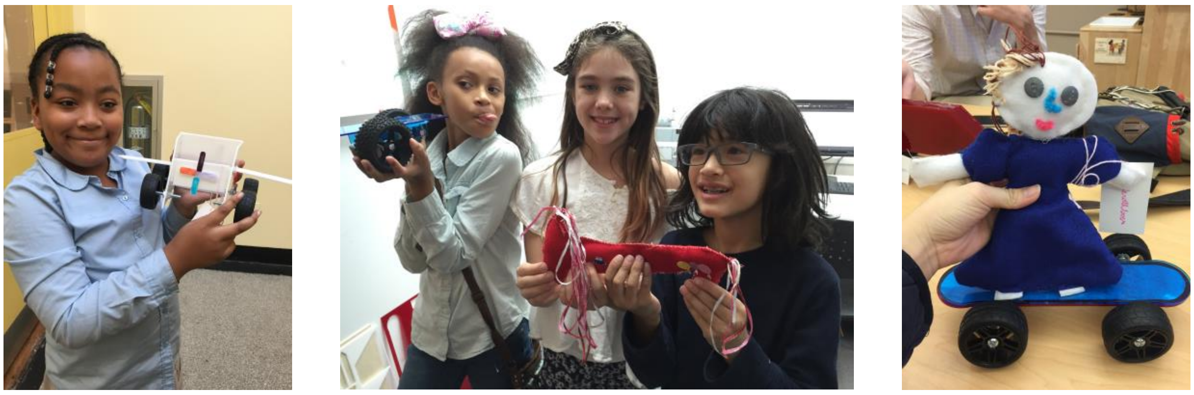


Figure 1. Dream toys were constructed using a variety of methods, such as sewing, lasercutting, and 3D printing as well as materials such as fabric, cloth, wood, acrylic, and extruded plastic.

Keywords

Construction; STEM; making; technology; craft; increasing diversity

Introduction

Projects designed to give children experiences playing and building with high tech equipment such as 3D printers, laser cutters, and microcontrollers have gained momentum in recent years among researchers, educators, and parents. Despite an explicit commitment to epistemic diversity, *makerspaces* have struggled to serve a diverse population of creators and have become heavily dominated by men and the highly educated and wealthy (Moilanen, 2012). Maker communities, companies, and flagship projects tend to revolve around typically male-centric projects such as cars, robots, and rockets and fabrication equipment development kits continues to require a prohibitive investment (Buechley, 2013). As the “maker movement” continues to expand and enter formal education settings this mainstream method of engaging kids with STEM ideas is in danger of marginalizing girls and underrepresented communities, leaving them out of this exciting opportunity.

In an attempt to counteract this imbalance and begin to explicitly target women and underrepresented communities some designers and scholars have explored the potential of adding technology and computation to existing craft cultures and communities such as sewing and woodworking (Buechley & Perner-Wilson, 2012). While the development of sewable microcontrollers and conductive threads has led to a burst of activity within existing craft communities, these tools and fashion-centric activities have also become the de facto method of engaging young girls (who may or may not have any experience sewing) in making (e.g. Kafai, Peppler, Burke, Moore, & Glosson, 2010). Though we are optimistic about these efforts, there is a danger that relying solely on interest and assumed gender and cultural “norms”, such as “girls like fashion,” may inadvertently perpetuate gender and cultural stereotypes and exacerbate existing community divides.

Drawing on literature from the feminist tradition (e.g. Belenky, 1986), research on service learning (e.g. Bielefeldt, Paterson, & Swan, 2009), and data indicating female makers are driven by a desire to help and give back to their communities (e.g. Intel Corporation, 2014), the Bots for Tots project is an effort to move beyond surface level participant characteristics and to instead explore the affordances of activity framings and structures that tap into alternate mental dispositions and ways of knowing to broaden participation and interest in maker activities. In this poster we offer a brief description of the project and describe some early findings from a pilot implementation.

Activity Design

The Bots for Tots project engages elementary children to design and build a “dream toy” for younger children in their community. Workshop sessions are designed to engage participants in interviewing stakeholders, brainstorming and critiquing, prototyping, and construction. In our first pilot of the Bots for Tots project, students in two forth grade class from an elementary school located in a highly urban Northeastern US city were invited to attend a free, five-day “making workshop” with the explicit goal of designing and building toys for their school’s pre-kindergarten (preK) class. Participants were broken into five design teams consisting of two children each. Design teams interviewed 3-5 preK children asking them to describe a dream toy. Design teams were prompted to probe explicitly for information about what the dream toy might look like, what it would be made of, and what it could do.

In the following workshop session, design teams described the dream toys requested and began brainstorming, individually, as a team, and as a full group, ways to combine and construct these toys. Following this brainstorming session, design teams began prototyping possible designs with an eye towards exploring the mechanics and functionality of the toy. Prototype construction and iteration occurred over 2 workshop sessions before design teams began constructing dream toys. Throughout all prototyping and final construction sessions, participants frequently came together to discuss the design process and to offer suggestions and criticism. Dream toys were constructed using a variety of methods, such as sewing, lasercutting, and 3D printing as well as materials such as fabric, cloth, wood, acrylic, and extruded plastic.

All design sessions were video recorded. Participants were interviewed before the workshop began and after the final session to determine the extent to which their interest in making and craft changed or stayed the same and to determine which activities they found interesting, motivating, and challenging. A STEM and technology interest survey was also conducted during the first session (composed of selected items from the Young Children's Computer Inventory v5.27 (Miyashita & Knezek, 1992) and the STEM Semantics Survey (Tyler-Wood et al., 2010)).

Preliminary Findings

Data collection and analysis is ongoing, however, a few interesting observations suggest this activity framing may be fruitful. In our first pilot of the project, ten children, eight girls and two boys, ages 8-10 responded within 36 hours of receiving a flier describing the workshop. One of the boy participants even expressed dismay when arriving for the first workshop session stating, "I figured there would be more boys than girls here!' Three girls explicitly indicated an interest making toys for the school's preK children suggesting that the workshop goals may have been motivating for some of the girls.

During pre-workshop interviews, all participants expressed enthusiasm for technology and making and these findings were verified by the interest survey. Participants also claimed to frequently use craft materials during their free time. For two participants, crafting and construction seemed to be particularly central to their interests as they described discovering on their own that they could mix flour and water to create play objects such as cups and bowls for use with dolls and other toys. Further analysis will evaluate the materials and techniques used by participants, how mixed and same gendered teams interacted, STEM content encountered by participants, and the degree to which framing the activity as being about making for others impacted day to day activities.

References

Belenky, M. F. (1986). *Women's Ways of Knowing: The Development of Self, Voice, and Mind*. Basic Books.

Bielefeldt, A., Paterson, K., & Swan, C. (2009). Measuring the impacts of project-based service learning. In *ASEE Annual Conference Proceedings*.

Buechley, L. (2013). *Thinking about making*. Presented at the Fablearn 2013. Retrieved from http://edstream.stanford.edu/Video/Play/883b61dd951d4d3f90abeec65eead2911d

Buechley, L., & Perner-Wilson, H. (2012). Crafting Technology: Reimagining the Processes, Materials, and Cultures of Electronics. *ACM Transactions on CHI*, *19*(3), 21:1–21:21.

Intel Corporation. (2014). *MakeHers Report: Engaging Girls and Women in Technology through Making, Creating, and Inventing*. Retrieved from http://www.intel.com/content/www/us/en/technology-in-education/making-her-future-report.html

Kafai, Y. B., Peppler, K. A., Burke, Q., Moore, M., & Glosson, D. (2010). Fröbel's Forgotten Gift: Textile Construction Kits as Pathways into Play, Design and Computation. In *Proceedings of the 9th International IDC Conference* (pp. 214–217). New York, NY, USA: ACM.

Miyashita, K., & Knezek, G. (1992). The Young Children's Computer Inventory: A Likert Scale for Assessing Attitudes Related to Computers in Instruction. *Journal of Computing in Childhood Education*, *3*(1), 63–72.

Moilanen, J. (2012). Emerging Hackerspaces – Peer-Production Generation. In I. Hammouda, B. Lundell, T. Mikkonen, & W. Scacchi (Eds.), *Open Source Systems: Long-Term Sustainability* (pp. 94–111). Springer Berlin Heidelberg.

Tyler-Wood, T., Knezek, G., Christensen, R., Tyler-Wood, T., Knezek, G., & Christensen, R. (2010). Instruments for Assessing Interest in STEM Content and Careers. *Journal of Technology and Teacher Education*, *18*(2), 345–368.

Bridge of popsicle sticks: A project and a contest

Esteban Pablo Díaz, estebanpablo@yahoo.com.mx
Jean Piaget University

Avenilde Romo Vázquez, avenildita@gmail.com
CICATA-Legaria, National Polytechnic Institute

Alejandro Rosas Mendoza, alerosas@ipn.mx
CICATA-Legaria, National Polytechnic Institute

Abstract

Building popsicle sticks bridges is consider as a relevant activity for engineering students, many professional associations (Professional Engineers and Geoscientist of BC, 2015) and academic corpuses have established contests for different academic levels. As part of his degree thesis, one of the authors opened a contest for engineering students in the University where he teaches and we report some of the results we have found in this contest. By now, students have shown a great interest and participation in the project.

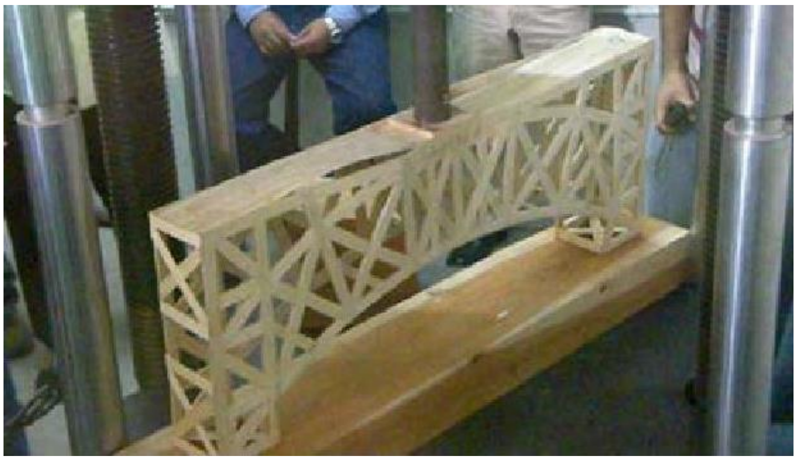


Figure 1. Popsicle sticks bridge

Keywords

Popsicle stick bridge, contest

The project and the contest

As part of his thesis work one of the authors suggested, to develop a research project which poses as a final product to build a model bridge-like structure built with wooden sticks. To develop this project we decided to present it to students as a contest. Students to organize into teams and conducted in parallel with their courses, as this activity is not part of the agenda of any current course. The time allowed for the development of the bridge since its introduction to the constitution of timber structures is 10 weeks. In order that students can take after this time, some tracks, consisting of the suggestion of some documents, videos, interactive websites, and simplified calculation programs armour, among others proposed.

This activity seeks as its main focus to build a model bridge-like structure with wooden sticks (as used by doctors) and white glue, which must meet the spatial and geometric requirements established in the contest. Technologies that will support them to achieve maximum efficiency of this model is that students also seeks to optimize the use of materials, using their skills, recognizing the different tasks involved, the techniques available and they must learn, as well.

The features that the building structure must meet are among others:

1) The maximum total weight of the bridge will be 1,150 gr. including the weight of the adhesive but not the supports and / or bases.

2) The bridge must be supported only at the ends, i.e. at the edges of the base (no supports on the sides).

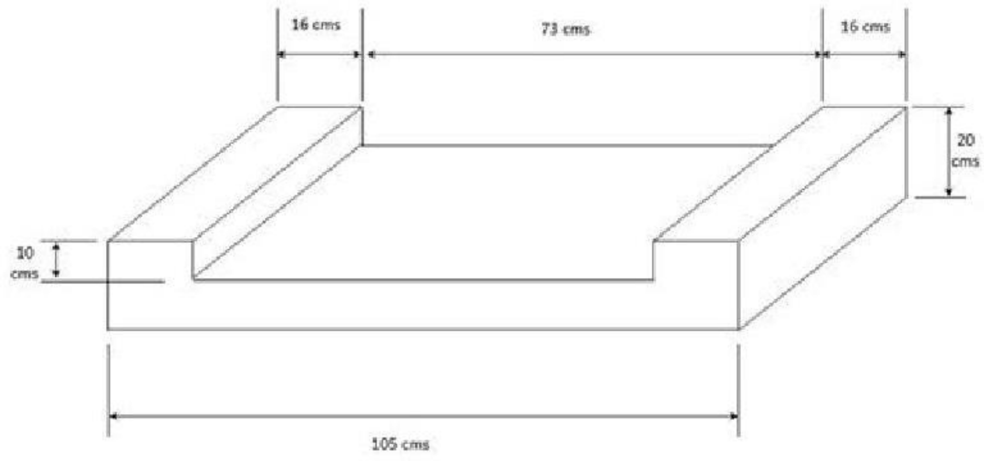


Figure 2. Measures of the popsicle sticks bridge

Results

By the time we write this report we only have preliminary results because the project is in its 6th week out of 10 weeks. In December the project will be complete and we will show the full set of results in February. But for now we can say that students have been participating actively, they have searched for information in internet and in libraries. Their first two reports let us see a big interest in this contest.

References

Professional Engineers and Geoscientist of BC. (2015). 2015 popsicle stick bridge building competition. Rule Book. Recovered from https://www.apeg.bc.ca/getmedia/37223d5c-ada5-4626-83cf-a2aa4a0c5207/northernbranch-rulebook2015.pdf.aspx

Compassion and Empathy through Inventions: GoGo Board Toolkit for 7-10 years old

Sawaros Thanapornsangsuth, *st2839@tc.columbia.edu*
Teachers College, Columbia University

Abstract

GoGo board toolkit is a hardware-embedded curriculum designed to teach children aged 7-10 years old (grade 2-5) the concept of compassion and empathy through inventions. The toolkit is inspired by stories that adults read out loud to their loved ones. Many storybooks are interactive but not so many engender proactive behaviors from the young readers. Storybooks in GoGo Board toolkit aims to help young children observe and understand the problems of others. With support from adults, children can construct original inventions that help others solve their problems using GoGo Board, scrap or prototype materials, and their imagination. The main objective is to inspire children to be active social inventors who can see the problems of others or themselves and be eager to solve them

Keywords

GoGo Board; Tinkering; Constructionism; Elementary Education; Toolkit; Inventions; Empathy

Problems

Learning should be a fun and engaging experience and it should take place anywhere and anytime. School time should not be the only place and time that children can learn. Teachers also should not be children's only source of knowledge. The problem that I perceive is that young children, once they enter the school system, often gain new knowledge by receiving rather than constructing. Children are "active builder of their own intellectual structure" (Papert, 1980, p. 19). Papert (1980) argues that knowledge does not build from nothing but is influenced by the environment and surroundings. However, many schools that teach in didactic manners views teaching students as a linear progression: from basic skills to more complicated skills. Teachers have obligation to manage the classroom as well as making sure that they cover the material that their students need to know (Wenglinsky, 2005). Thus cross subject area integration often lacks and feeding knowledge seems to be a timely efficient way to catch up with curriculum. However, it is not effective since not every student can receive the same amount of knowledge and aren't able to learn at their own pace. Although learning takes place informally and in a less stressful environment in home settings, constructionist learning has not been much promoted. For example, when parents read a story to their child, the child tends to be the receiver of knowledge and passively listens to the story rather than contributing or making observations about the story.

Literature Review

Constructionism incorporates two types of construction: "construction of knowledge in the context of building personally meaningful artifacts" (Kafai & Resnick, 1996). GoGo Board is a low-cost open source hardware device kit that encourages young users to plan and set a goal while creating meaningful automated projects (Learning Inventions Laboratory, 2015) that they care about through simple programming blocks. Within the process, the software provides a tinkering space for young users to learn and experiment. They can try using multiple sensors and actuators to see how they react right away. The design of the hardware supports "low floor" and "wide wall" that is easy for young users to get started while still providing opportunities for the creation of a wide range of different explorations (Resnick & Silverman, 2005). Unlike early programming software, where syntax is too difficult for young users to master, Tinker: GoGo Board offers a more kid-friendly interface with bright design, where the programmable actions can be inviting and meaningful to young users.

GoGo Board Toolkit

The GoGo Board toolkit is comprised of four main objects:

1. Adult Guidebook (printed and/or digital) are for parents, teachers, or caregivers as an introductory guide to GoGo board, as well as some suggestions of how GoGo board can be taught to their children or students. The guidebook introduces the following topics: (1) Introduction to GoGo Board, (2) Coding with GoGo Board, (3) Guideline and ideas to facilitate children learning and making process in constructionist manner. The guidebook comes with *Program Control Mat*s and a deck *of function cards, sensor cards, and actuator cards* that would teach children about each input and output in a fun and experimental manner.
2. "Tommy and the Inventor" storybook (printed and/or digital) for adults to read out loud to children. The book helps children to understand different kind of sensors by inviting them to help the main character by choosing the right sensors. At the end of the book, children are encouraged to write their own stories
3. "I am an Inventor: Family Edition," or a creator storybook (printed) guides the children to invent for people in their families. It's a combination of: (1) story book, (2) drawing and coloring book, and (3) basic guidelines for using GoGo Board. It allows the children to personalize their inventions and ask them to write problem statements to solve the given problems. The book encourages children to start programming with GoGo Board, create prototypes of automatic devices, and test their inventions out. Adult guidance is highly encouraged.
4. Blank book (printed and/or digital) for the children to create their own narratives. Children can write, draw, or animate original stories from their own experience, imagination, or even stories from others. The stories must include the problem that they want to solve. Then they will build automatic devices using GoGo Board to solve the particular problems.

Learning Goals

1. The ability to understand basic functions of the GoGo Board and start programming and tinkering with it.
2. The ability to initiate creative and meaningful physical computing projects of their own using resources that are available to them and the GoGo Board. My hope is to see the inventions that children can continually surprise themselves and people around them. Within the process of designing and coming up with tangible products to solve the problems that interest them, the children need to tinker, encounter obstacles, make mistakes, and experience disappointing results. Being able to persevere, manage frustration, think creatively and work with others are valuable skills that can be generated through making with GoGo Board.
3. The ability to see the problem and figure out how to solve it in a structural and logical manner. Inspired by Papert's Mindstorms (1993) the design believes that kids learn most effectively when they try to solve problems where they are genuinely interested in the outcome of the problem.
4. Engaging in constructive social and emotional interactions especially in compassion and empathy in the narratives. Moreover, the process of making can be a social experience that children learn to interact and work with others with respect.

The learning goal for the adults:

1. Be able to gain confidence in technology to the level that they can facilitate the children to use GoGo Board.

References

Learning Inventions Laboratory. (2015). GoGo Board. Retrieved from https://learninginventions.org/?page_id=1252

Kafai, Y. B., & Resnick, M. (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world*. Routledge.

Konrath, S. H., O'Brien, E. H., & Hsing, C. (2010). Changes in dispositional empathy in American college students over time: A meta-analysis. *Personality and Social Psychology Review*.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc..

Resnick, M., & Silverman, B. (2005, June). Some reflections on designing construction kits for kids. In *Proceedings of the 2005 conference on Interaction design and children* (pp. 117-122). ACM.

Wenglinsky, H. (2005). Using technology wisely: The keys to success in schools. Teachers College Press.

Increasing Learning Gain in Linear Algebra through Play

Suttipong Thajchayapong, *suttipong.thajchayapong@nectec.or.th*
National Electronic and Computer Technology Center (NECTEC), NSTDA, Thailand

Tanut Choksajawatee, *tanut.tanata@mail.kmutt.ac.th*
Institute of Field Robotics, King Mongkut's University of Technology Thonburi, Thailand

Paron Israsena, *paroni@cscoms.com*
Darunsikkhalai School for Innovative Learning, King Mongkut's University of Technology Thonburi, Thailand

Chanikarn Wongviriyawong, *chanikarn.won@kmutt.ac.th*
Institute of Field Robotics, King Mongkut's University of Technology Thonburi, Thailand

Abstract

This paper proposes a tool for learning linear algebra we invented called electronic Math (eM), which was aimed to assist learners in constructing mathematical understandings on solving multiple variable equations through play. We conducted an experiment on Grade 7 and 8 students at Darunsikkhalai School for Innovative Learning, Bangkok, Thailand. We found that using this tool, learner's scores on exams increased on average by 32.5% ($p<0.05$, paired *t*-test). This tool will be further developed to collect real-time data of learner's motion while being engaged on a task to help us deepen our understanding towards reasoning behind for such dramatic improvement in learning gains.

Keywords

Embodied Cognition; Learning Environment; Mathematical Education

Electronic Math (eM)

Facilitating learners to learn complex mathematical concepts such as gauss elimination in linear algebra to simultaneously solve a system of equations can be a challenging task. We are interested in designing toys that can assist this process of learning how to solve systems of equation through interacting with “objects to think with”. Such object was designed to mimic the thinking process similar to that occurring while one was solving systems of equations. We focused on a lesson on solving a system of linear equations.

eM tool has 3 main components: 1) a tablet with our mobile application for monitoring learning outcome (Fig. 1), 2) multiple blocks (squares, rectangles, and triangles) that send signals to distinctly identify themselves (Fig. 1), 3) base of the eM tool (wooden-like color), comprising main controller that gather all data from blocks before sending via bluetooth to the tablet. Each set consists of a tablet, a base and 16 blocks (4 shapes). The shape of each block is distinctly identifiable by its color. Square block is pink; short rectangle is light green, long rectangle is dark green (not shown in Fig. 1) and triangle is blue.

One of our designed lessons on linear algebra was tested. Learners were given a puzzle where the task was to fill the shadow with real blocks on the base to form the shape they saw (Fig. 1). Although there were many ways to form the same shadow, there was only one correct combination. Blocks of certain shape were represented a distinct positive integer value. A combination of blocks symbolized a sum of those numbers. For example, 2 squares (with a value of 2) and 1 short rectangle (with a value of 7) when placed in combination would yield 11, that is $2*2 + 1*7$. These numbers were presented to learners as each shadow was correctly filled. This lesson comprises 10 distinctly different puzzles.

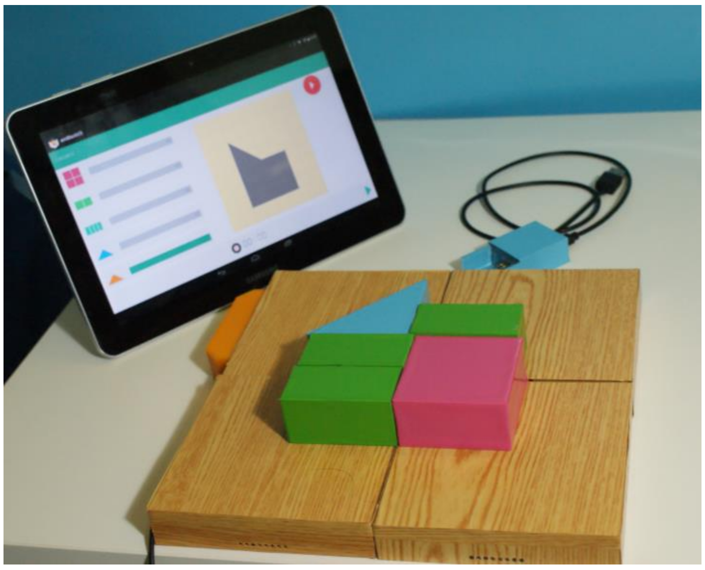


Figure 1. eM tool Designed as a Tangible Environment for Learning Linear Algebra, where blocks in each color have distinct numerical values, and their juxtaposition corresponds to the sum of those numbers. This tool comprises 3 parts: blocks, base, and the mobile application.

Experiments

We conducted two experiments: one using eM tool and the other a standard lecture method in a math classroom at Darunsikkhalai School for Innovative Learning. The first experiment was conducted with 11 students from Grade 7 (age: 12-13 years old, M:F = 8:3). Each student took a pretest consisting of 15 questions about solving one variable equations, such as 2*X + 7 = 11, where X had to be an integer (similar to our lesson design). Students were given 50 minutes to complete the test. The week after, we presented how to use the eM tool, but did not tell them that it was related to math. Each group (5 groups of two, and one group of one) got 50 minutes to play with the eM tool at their own pace. Each group must correctly complete the lesson before the next question. Once all the questions were completed or 50 minutes lapsed, whichever one came sooner, the game stopped. After all groups completed the game, we administered a test under 50 minutes. Second experiment was conducted in a pre-post intervention design with 6 students from Grade 8 (age: 13-14 years old, M:F = 5:1). However, this time students did not interact with our eM tool, but went through a standard teacher-teaching style class. We calculated %Correct (Pre) and %Correct (Post) as an accuracy of their answers (average scores on questions with any answers) for each student. Learning behaviors of students were manually recorded by a team of 4 researchers. Videos were also recorded. Data were collected in the same way in both experiments.

Results and Discussion

We found that in Grade 7 class where our eM tool was used, the %Correct before and after learners were engaged with the tool were significantly different. %Correct of learners increased after using our eM tool ($p<0.05$, paired *t*-test), with an average increase of 32.5%. However, in Grade 8 class where a standard teacher-teaching lecture was used, the pre-lecture and post-lecture %Correct were not statistically different. Figure 2 shows two scatter plots comparing the first experiment with our eM tool (left) and second experiment with a standard lecture approach (right). Each circle represent %Correct of a single student pre and post engaging with our eM tool or being in a standard lecture. A total of 8 out of 11 students had higher % accuracy on the test after engaging with the eM tool (Fig. 2 Left), while only one student had better accuracy on their test after a lecture (Fig. 2 right).

Moreover, we found that the number of questions students answered was significantly different before and after they used our eM tool ($p<0.05$, paired *t*-test). The number of questions students answered after engaging with our eM tool reduced on average by 15.2%. Interestingly, even though the number of questions answered were lowered, their accuracy increased. This could hint at the potential validity of using our proposed eM tool to facilitate learning how to solve equations without involving in a teacher-teaching approach and still receive better results on exams. This finding did not come as a surprise, as we would expect that learners who are engaged in a thinking process similar to that occurring when one was solving a system of equations with multiple variables would construct their thoughts while manipulating these objects. It is possible that while immersing oneself to this new task. Students may be constructing their own mathematical constructs, having their sensorimotor capacities stimulated and engaged with the tool.

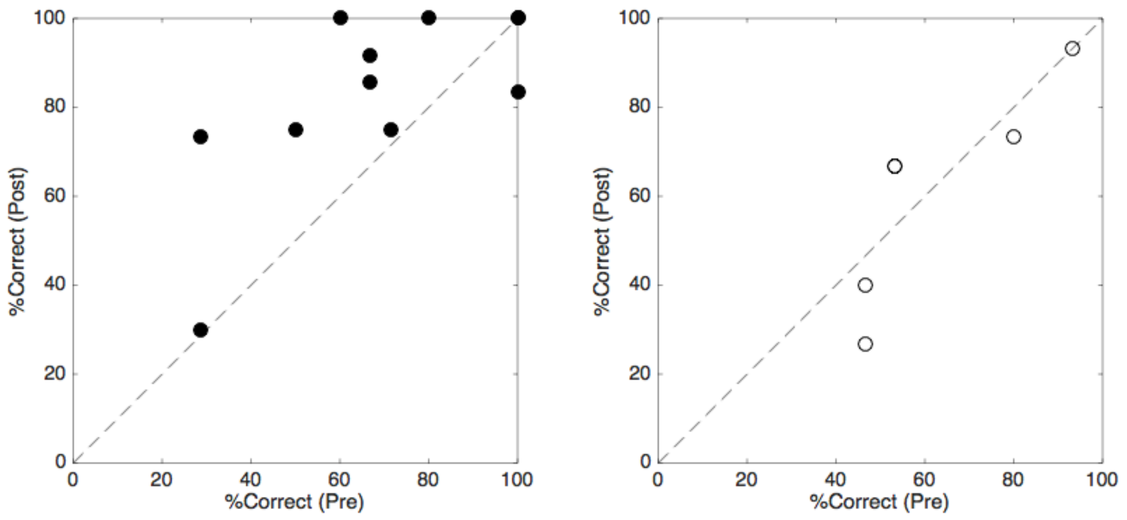


Figure 2: Results from having Grade 7 students use eM Block prototype between the pretest and posttest. Student's posttest scores improved by approximately 32%.

We observed that despite differences in the characteristics of learning behaviors, the time students spent and the number of attempts before reaching correct answers substantially reduced, not in a linear manner. This could have indicated that learning occurred in between engaging in solving these puzzles presented by this tool. Although these results are based on a preliminary analysis of clusters of students, they could pinpoint that these students were learning while manipulating these objects. Our preliminary results prompt for further investigation. Our future work would, therefore, include conducting same experiments on a much larger number of students. We are currently investigating how our data can be used in several cognitive models such as learning factor analysis (Cen, 2006), ACT-R theory of skill and knowledge (Anderson, 1993; Lebiere, 1999), ACT-RN (Anderson, 2003), ICARUS (Langley, 2009), and time-scale dynamical model (Thelen, 1995).

References

Anderson, J. (1993). Rules of the mind Lawrence Erlbaum Associates.Hillsdale, NJ.

Anderson, J. R., & Lebiere, C. (2003). The Newell test for a theory of cognition. Behavioral and brain Sciences, 26(05), 587-601.

Cen, H., Koedinger, K., & Junker, B. (2006, January). Learning factors analysis–a general method for cognitive model evaluation and improvement. In Intelligent tutoring systems (pp. 164-175). Springer Berlin Heidelberg.

Langley, P., Laird, J. E., & Rogers, S. (2009). Cognitive architectures: Research issues and challenges. Cognitive Systems Research, 10(2), 141-160.

Lebiere, C. (1999). The dynamics of cognition: An ACT-R model of cognitive arithmetic. Kognitionswissenschaft, 8(1), 5-19.

Thelen, E. (1995). Time-scale dynamics and the development of an embodied cognition. Mind as motion: Explorations in the dynamics of cognition, 69-100.

Little Builders: Empowering At-risk Children by Building and Design

Sawaros Thanapornsangsuth, *st2839@tc.columbia.edu*
Teachers College, Columbia University

Yongyuth Laitavorn, *yongyuthlai@gmail.com*
Chevon Thailand Exploration and Production, Ltd.

Abstract

Little Builders is a social enterprise that aims to disrupt Thai education by employing constructionist design paradigms and human-centered design processes in at-risk formal and informal schools to foster a grit and growth mindset. We believe that constructing is a valuable learning process for the students; “children don’t get ideas; they make ideas” (Kafai & Resnick, 1996). *Little Builders* was first established in early 2014 by a group of recent graduates from various backgrounds but a common goal of shifting the Thai classroom from instructionist to constructionist systems, where students can work on their own meaningful project, learn at their own pace, and at the same time create a community of learners. Through our two years of commitment, *Little Builders* has received a grant from Thai Social Enterprise Office and currently has 7 core team members, over 100 active volunteers, and 533 supporters.

Keywords

Constructionist design paradigm; Human-centered design; Thailand; At-risk students

Literature Review

Instructionism views teaching students as a linear progression: from fact to analysis and from basic skills to more complicated skills (Wenglinsky, 2005). The teacher is the center of knowledge in the instructionist classroom and she or he would spend the majority of class time on conveying new knowledge to the students. Piaget asserts that “knowledge is not simply transmitted from teacher to students, but actively constructed by the mind of learner” (Kafai & Resnick, 1996, p. 1). Constructionism is in sharp contrast to the didactic approach. For instructionists, transmitting knowledge is a quick process by employing a top-down approach from expert to novice. It is taught by showing and moving toward abstractions. The knowledge also has to be in an accurate form of answer. Constructionist design paradigm is a slow process that encourages emergent knowledge sharing and embraces a community of learners. It is facilitated by building and moves towards concrete objects. Thus, the result of learning can be messy because learners are always experimenting and tinkering (Holbert, 2015).

Human-centered design (HCD) comprises of 5 distinct iterative steps: empathy, define, ideate, prototype, and test (IDEO, 2015). It starts from understanding the needs and motivation of targeted users. The human-centered designer shares core elements of the constructionist in being optimistic, experimental, and collaborative (Brown, 2008). Moreover, design thinkers are optimists believing that their designs can create change no matter how big the obstacles are; for them, solving challenges is an enjoyable process. Since the HCD is an iterative process, it gives the designers permission to fail fast and learn from their previous mistakes. They can always come up with new ideas, add on to them, then receive feedback to fix and iterate. HCD is all about learning by doing. (IDEO, 2015)

Previous Projects

Little Builder’s first project was piloted at *Janusz Korczak School*, an informal school for HIV+, immigrant, low-income, and street children in Slum Klong Teoi with 22 students and 8 volunteers.

We began by teaching simple circuitry and using a motor to develop low-cost racing cars from recycled milk cartons. We captured students' full attention. Students connected the circuits by themselves and assembled the car within the end of the period just to be able to race with their friends. They said that it felt more like a playground than a classroom. The same lesson was organized in several foundations and informal schools in Bangkok.

The next phase of development was at *Baantawanmai Foundation*, a foster foundation for Thai narcotic at-risk children. There were seventy 9-14 years old participants and almost thirty volunteers. We combined HCD process into students' creations. We started by letting students find their own problems or needs in their own community. They came up with a host of problems. The library was too loud because it is located next to a music room; the decomposing plant and fish caused foul odor from the pond; there were pests in the bathroom; the local farm was too big which took too long to water. They ideated solutions, and developed quick prototypes from scrap materials then presented in front of the whole community. Two ideas were selected to be built by the students and volunteers during *Builder Day.* Together, we designed and created an irrigation system covering an acre, using 300 meters of piping system and almost a thousand sprinklers. The water was supplied by automatic reciprocating pumps, creatively made by modifying salvaged bicycle parts. The pump can either supply the irrigation system directly or fill up a water storage tank for later use. The younger participants also learned the concept of art and chemistry by creating pest repellent candles to solve the problem of pest control in the community's residential area.

Current Project

Seeing the problem with sustainability in running constructionist projects, we seek ways to integrate *Little Builders* in to Thai formal schools and provide professional development for teachers at the same time. We will be initiating our project at Wat Pak Bor School (an urban school under Department of Education Bangkok Metropolitan administration) from November 16, 2015 - February 13, 2016. We received strong support from the school's principal, authorizing 160 8th grade students and 13 teachers to participate in our program. The project will be assessed by pre and post GRIT (Duckworth et al., 2007) and Growth Mindset (Dweck, 2006) assessment.

The participants will be using HCD processes to design and build constructionist projects. The program will start off with ice breaking and team building activities then a full day of overview in HCD processes through activities of building small projects. After giving the participants an overview of constructionist projects and HCD, the participants will come up with their own meaningful project on *Empathy & Define Day*. Coaches will train the participants on interview methods then let them go out to interview the community and be immersed with their target users to figure out the users' needs (define). We will continue to work on the rest of the HCD processes (ideate, prototype, and test) the following week. Once an idea is formulated, the participants will start building on *Builder Day*; the building process will take up to 2 weeks. Since the participants will be presenting and exhibiting their *Little Builders* projects to the public audience, we will also coach them on public speaking skills. The last day of the project is the *Showcase Day,* where we hope to see the participant taking ownership of their projects.

The future of education is no longer a straight line of desks and a chalkboard with standardized tests to measure students' competency, nor it is in front of an iPad screen. *Little Builders* sees the future of education in training the next generation to think and act creatively. Design the tools that allow children to learn, and allow children to design what they want to learn.

Reference

Brown, T. (2008). Design thinking. *Harvard business review*, *86*(6), 84.

Duckworth, A. L., Peterson, C., Matthews, M. D., & Kelly, D. R. (2007). Grit: perseverance and passion for long-term goals. *Journal of personality and social psychology*, *92*(6), 1087.

Dweck, C. (2006). *Mindset: The new psychology of success*. Random House.

Holbert, N. (2015). *Tool and Toys for Knowledge Construction.* [PowerPoint slides].

IDEO. (2015). Our Approach: Design Thinking. Retrieved from www.ideo.com/about/

Kafai, Y. B., & Resnick, M. (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world*. Routledge.

Wenglinsky, H. (2005). Using technology wisely: The keys to success in schools. Teachers College Press.

On Controlling LEGO Education Platforms from Imagine Logo

Pavel Petrovič, *ppetrovic@acm.org*
Dept of Applied Informatics, Comenius University, Bratislava

Imagine Logo (Kalaš and Hrušecká, 2004) is a successful learning environment. It is providing a rich set of features, yet it is very accessible to novice learners. On one hand, it is based on a very old textual programming language Logo, which suffers from a couple of disputable design choices. For instance a word has three different meanings: a variable, a function, and a symbol. A cumbersome syntax distinguishes between them: a colon, and an opening double-quote character. These may perhaps be difficult concepts to comprehend for a child, but many were able to overcome this for their benefit. On the other hand, the interactivity and universality of Logo programming language, and the default mode of immediate command-line control of full-screen graphical environment is not replaceable by puzzle-like environment with build-and-run paradigm of Scratch. It disturbs the learner by almost always open editor, which is not necessarily visually appealing. Language, not building puzzles is the most natural way for humans to communicate with each other, and so it is also the most natural way to communicate with the computer, and always will be. Logo allows the user to extend its language. For a similar reason command-line based operating systems are so popular even in 2015. In every case, Imagine is a popular learning environment. LEGO Education robotics platforms provide an accessible, understandable, and well-established constructionist educational tool that complements it. In this poster, we present the technical solution and example projects for bridging both of these worlds in four generations. The result is a true constructionist tool and environment with unlimited possibilities for creativity.

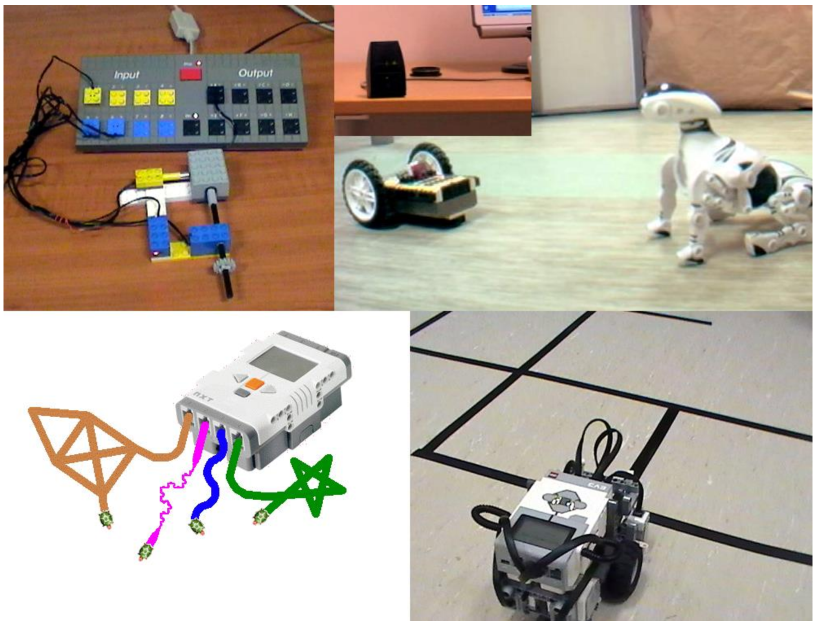


Figure 1: Four generations of robotic LEGO can be connected with Imagine Logo. From top-left: LEGO Dacta Control Lab – sensors and motors are connected through a serial port, and the control program runs on a PC; RCX – messages are delivered over infrared connection from an IR tower connected to a serial port, here RCX robot even controls a toy dog robot over IR; NXT – immediate commands or messages sent to programs written in NXT-G, or any other software that recognizes the standard message format; and EV3 –messages sent over Bluetooth connection to EV3 programs.

Keywords

Imagine Logo, LEGO MINDSRTORMS, Control Lab, RCX, NXT, EV3

Four generations of robotic LEGO

In its zeroth generation, LEGO Dacta Control Lab, the models must have been connected to the computer with cables and the control signals were generated by a program running on a personal computer and delivered to and from the model over specialized interface connected to a serial RS232 port. However, the educational programming language that was part of the setup, LEGO Logo, made it possible for children to develop interactive educational projects with data visualization, and system control. Unfortunately, this potential have never been replaced by any of the later models, which adopted the program-download-run paradigm, throwing away computer language interactivity we mentioned above for good. On one hand, this simplified the scenario, perhaps, making it more accessible and understandable for learners. On the other hand, it disabled the programming part during the robot performance. As a consequence, programs for the MINDSTORMS robots are difficult to debug, and do not fit very well with incremental interactive learning scenarios that have worked so well with Logo language. Interactive programming languages let the user to evaluate any expression at any time, manipulate the values of the variables, inspect and modify the variables and structures in memory, call individual functions in random combinations as suitable, modify the settings during the run, all that in a step-by-step analytical and playful manner. This makes the programming more fun, promotes learning and active exploration and facilitates constructionist and constructivist learning as well. On the contrary, programming MINDSTORMS robots of all three generations is often a time-consuming, try-again-and-again-and-again frustrating experience. When the hardware complexity is combined with the software complexity, the number of possibilities how things can go wrong multiplies. It is therefore a sad discovery that MINDSTORMS programming has no debugging features, not even a Bluetooth serial console window – the most natural element of the popular Arduino learning environment. The user cannot see what is written on a little passive LCD when the robot performs manipulation tasks and the display is hidden under the robot chassis. There is still so much that LEGO should learn from its great past, a little bit can be seen in WEDO. Despite the powerful technologies provided by the National Instruments, it should definitely reconsider whether the LabView kernel is a suitable one for children programming language. In spite of having a rich set of possibilities and unique position in terms of being prepared for the inevitable upcoming large-scale computing parallelization because of its data-flow design model, which, unfortunately, has not been adopted by LEGO programming languages NXT-G and EV3 anyway.

Solution

We started to provide a workaround with (Petrovič, 2007). We have interconnected the two successful platforms – Imagine Logo and MINDSTORMS robots of all four generations. Our set of utilities allow sending infrared – in case of RCX, and BlueTooth, in case of NXT and EV3 messages between Imagine and LEGO. In addition, in a school project work, our students have implemented an interface between Imagine and the old CtrLab interface module: sensors can be read and motors controlled. The software and examples is available (Petrovič, 2007-15).

The technical solution considered different ways of built-in features of Imagine Logo. First of all, there is built-in support for communication over serial port, but it is not fully configurable and shares thread with other system components of Imagine. A problem on a serial line may thus lead to further problems. Second, special pluggable ActiveX components can be mapped to Imagine objects. Their methods can be directly called from Imagine. We used it in some projects, however, not all versions of Imagine support ActiveX and this 1996 technology has finally been discontinued by Microsoft anyway. However, Imagine can communicate over TCP sockets with another instance of Imagine. We tweaked the protocol and plugged our communication component at the other end of line instead of Imagine and implemented details.

References

Kalaš I. and Hrušecká A. (2004) The Great Big Imagine Logo Project book, Logotron.

Petrovič, P. (2007) Program Your NXT Robot with Imagine, Eurologo 2007.

Petrovič, P. (2007-2015) Projects with four generations of robotics LEGO, on-line at http://dai.fmph.uniba.sk/projects/lego

Robot Programming Workshop for Middle and High School Girls

Miki Matsumasa (MS, Computer Science, Tsuda College), Chieko Harayama,Kazuhiro Abe, Nobuko Kishi(*Tsuda College*), Manabu Sugiura(*Yamanashi Eiwa College)*

Abstract

We held programming workshops for middle and high school female students using Scratch to program educational robots. We presented the educational robots, Romo, as social robots to the students and asked them to design programs to improve the interactions between human and robots. We believe we were able to have the students, novice programmers, imagine the future society with very low cost robots in our daily life. In this paper we would like to describe the workshop outline, the interaction programs that the students created in the workshop and the feedback that we received from them.

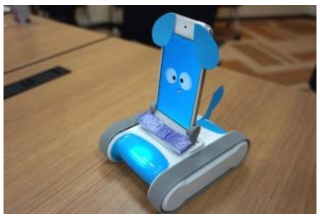
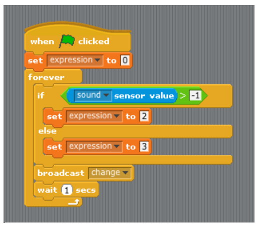


Figure 1. Romo From Romotive Inc. with iPod Touch



```
when [flag] clicked
set expression to 0
forever
    if sound sensor value > -1
        set expression to 2
    else
        set expression to 3
    broadcast change
    wait 1 secs
```

Figure 2. Scratch2Romo an enhanced version of Scratch

Keywords

Computer Science Education, Middle and High School Education, Social Robots.

Background

At Tsuda College, we have held programming workshops for middle and high school students over the past several years. At our workshops we teach Scratch (scratch.mit.edu) together with other new technologies. We chose Scratch because it is very easy for middle and high school students to program. We also believed they were capable of learning more than Scratch, so we decided to use Romo as a new technology in the summer of 2015. Romo is a low-cost educational robot. It has caterpillars with a docking station for iPhone or iPod touch (Figure1). Scratch2Romo is a bridge application to connect Scratch with Romo. It can control Romo from Scratch by using Wi-Fi network and Scratch Remote Sensor protocol (Figure2).

Workshop Outline

The students have engaged in the following activities to understand what robots can do and cannot do.

1. Understanding social robots

   Keep a question "What is a social robot?" in mind.

2. Preparing instruments

   Listen to the explanation of instruments, Romo and PC, and set them up.

3. Programming Romo by Scratch

Understand the blocks for Scratch2Romo like changing expression, moving, saying, and sensing.

4. Designing and creating a social robot.

   Form a team of two students and one TA. Propose an idea of a social robot on design sheet, and code the idea in Scratch. .

5. Sharing their robots with others

   Walk around the room and watch other robots. Give some comments.

6. Reflection

   Reflect on the activities in the workshop.

The Interaction the Students Designed

It turned out that the students enjoyed the design process. The interactions that the students designed are:

- Wake-up robot:
  This robot is a robot that wakes you in the morning. When it is the time you set beforehand, the robot repeats saying “Morning! Wake up”. If you wake up and touch the hand of the robot, the robot stops saying and change its expression. And then, the robot says “Good morning!” Finally, the robot dances and takes a bow.

- Companion robot:
  This robot is a robot that talks to you. While you and the robot are enjoying some of the conversation, the expression of the robot changes depending on the volume and content of the conversation. At the end of the conversation, the robot takes your selfie photo.

- Recipe robot
  This robot tells you how to cook a recipe. First, the robot asks you “what do you want to cook? ”. If you answer “Fried eggs”, the robot tells you how to cook the fried eggs. The process of teaching becomes interactive.

Figure 3. Programinng Romo with Scratch.

Feedbacks and Conclusions

In the evaluation sheets, 28 out of 30 students stated that the workshop was fun. And some student said that they want to program again. All the students were able to have robots make expressions and conversation by using Romo and Scratch. Furthermore, they were able to design interactions between a robot and human. We believe the students understand the possible roles of social robots in the society though this workshop.

References

MIT Media Lab, Lifelong Kindergarten Group (n.d.) Scratch – Imagine, Program, Share, retrieved Oct 15, 2015 from https://scratch.mit.edu/

Romotive, Inc. & Sales On Demand Corporation, (2014) from http://www.romotive.jp/

Tsukuru, Inc. LLC, Scratch2Romo, (2015) from http://www.scratch2romo.com/

Teaching programming constructively and playfully

Ildikó Tasnádi, *tasnadi@radnoti.elte.hu*
ELTE Radnóti Miklós Gyakorlóiskola, 1146 Budapest, Cházár András u. 10

László Csink, *csink.laszlo@nik.uni-obuda.hu*
Károly Farkas, *farkas.karoly@nik.uni-obuda.hu*
John von Neumann Faculty of Informatics, Óbuda University, 1034 Budapest, Bécsi út 96/b

Abstract

We demonstrate our best practice to develop IT vision and thinking in school education and how we link the various subject groups to IT. We wish to show that writing programs is the best game to play with the computer, because it is better to command the computer than being commanded by it. To teach programming should not simply aim at practical competence but also to develop cognitive skills similarly to teaching math, but in many respects more effectively. We have experimented in a secondary training school in two age groups: 6-10 and 10-14. For the young group we use our own educational package, for the older group we use a freely available package (Fling the teacher http://www.contentgenerator.net/fling/ (last visited November 6, 2015)).

Keywords

Syntonicity, playful informatics, programming

Teaching programming

Do not train a child to learn by force or harshness; but direct them to it by what amuses their minds, so that you may be better able to discover with accuracy the peculiar bent of the genius of each. Plato

While the other teachers in our school used the traditional method of teaching basic commands, procedures, use of parameters etc., in our group we set a problem: design a game program. For implementation we used *Scratch*. This makes it possible that the user could experiment with her ideas. The concepts of program sequence, selection and iteration will be learnt without defining these. All the 126 students (four 11-year-old groups and three 14-year-old groups) were happily working on the project of designing their own game programs.

Scratch is especially suitable for this purpose. The environment is easy to use and the results can be seen at once.

The students knew the criteria of evaluation beforehand, based on competency evaluation principles. Their game programs had to satisfy the following:

- ✓ Use a built-in background or design a new one.
- ✓ Motion has to be governed using arrows the arrow keys (or some other way).
- ✓ Sensors (of colour or another actor) should be included.
- ✓ Sensors should evoke events (e.g. make another actor move).

The more ambitious students were expected to set initial values and use variables.

We spent six 45-minute lessons for the experiment.

This was sufficient for them to get some insight into programming, to raise some motivation so that students might work with the projects in their free time as well. The following were produced by 11-year-old students:

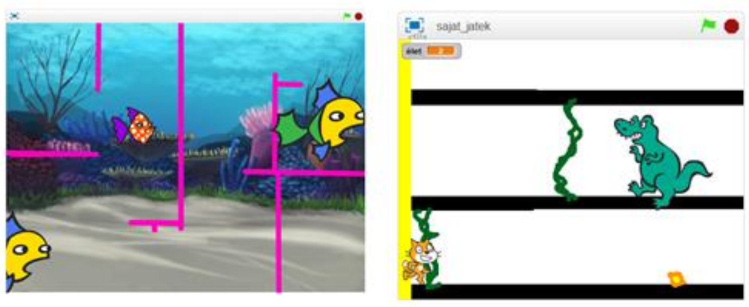


Figure 1. Labyrinths (designed by 11-year-olds)

Thematic unit plan example (Grade 5)

Lesson	Topic covered	Primary didactic task
1	Coding	Motivation
2	Introduction to the Scratch programming environment	**Setting the goal:** create your own game program
3	Control statements	
4	Event handling (simple conditions)	Presenting new material; Practice
5	Practice using variables (additional material)	
6	Finalizing the game program	Application of the learnt material; Control, Evaluation

The process of problem-based programming

When you work up the material, you can use various approaches to solve a problem. Everyone thinks differently, so he/she will solve the problem in his/her own way. Our aim was that the students should understand the gist of the problem. We provided hints but we never forced them to follow our guidelines. We let them experiment with their own ideas. Due to this they often met problems and asked for help or ideas. We had to be on standby for help in debugging or suggesting a new approach. This is the process of problem-based programming (Figure 2.).

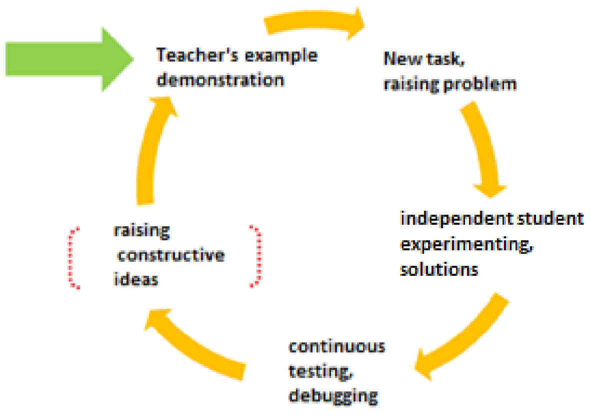

Teacher's example demonstration
New task, raising problem
independent student experimenting, solutions
continuous testing, debugging
raising constructive ideas

Figure 2. The process of problem-based programming

The results exceeded all our expectations. Even those students worked enthusiastically whom we couldn't make work earlier. As Polya said the task you are thinking on may be simple, but if it raises your interest, it will move your ideas and when you have succeeded solving it, you experience the excitement and victory of a discovery.

References

Csink, L., Farkas, K. (2012) *Greek Salad instead ...* In C. Kynigos, J. E. Clayson, N. Yiannoutsou Eds.: Constructionism: Theory, Practice and Impact, Athens, Greece. pp. 602 – 609.

Polya, Gy. (1954) *How to Solve It. Induction and Analogy in Mathematics*. Princeton.

Technicity

Micheál Ó Dúll (Mike Doyle), *logios,org@googlemail.com*

The British School of Sofia

Psychology has little to say about the 'how' of human creativity and the technologies that have, for more than 300 millennia, been at its foundation. Papert's intuition that "consciously constructing a public entity" is cognitively more powerful than a verbal formulation remains mired in philosophy. Yet the capacity for the 'good play' at the heart of creativity is uniquely human whilst language is held in common with others. How are humans able to create technology that gives the species power over nature? Perhaps answering the simple question "How can children draw?" would be a start?

Step 1: Assert that all information used to construct representations of the world and act upon it is genetically embedded in brains.

Proof: For over 500 million years vertebrates have perceived pink. No photon maps to pink. Therefore pink must be the product of genetically determined neural structures. The Hubel colour "blobs" in primary sensory cortex are such a structure. A corollary is that all perception, whatever the sense, is founded on such internal information. I.e. so-called feature detector neurones are information sources activated by sense data.

Step 2: Determine how this information might be made available to cognition.

Proposal: Given that increasing invasion by prefrontal neurones of all parts of the brain is a feature of evolutionary brain expansion, it is proposed that in humans connections are made to the "Hubel" information neurones in humans. They may thus be activated internally independently of sense data.

Support: The original Hubel and Wiesel 'feature detectors' specialised in lines:
1. Young children's first drawings are linear.

2. We conceive ideal geometric forms.

3. A square rotated changes its name to diamond, violating object constancy.

4. All technology and arts entail the combination of simple elements.

Step 3: Determine the adaptive advantage.

Discussion: Technology, going beyond the natural, gives humans, through science, power over nature. This presents a difficulty. Technology, from a classical viewpoint, is lower in entropy than any biological entity; yet it is the product of one. This appears to violate the second law of thermodynamics. The post-blackhole concept of entropy, with the four forces added, paints a different picture. No longer is there an equilibrium state of bouncing balls in a box (and its Maxwell demon); molecules aggregate under the influence of gravity and, as the other forces come into play, form imploding stars that produce high entropy black holes and low entropy molecules. Of the latter, DNA is unique: The ultimate Maxwell Demon; it encodes information, detects molecules, creates new ones, phenotypically reproduces, and creates new information: e.g. pink.

Given the character of the natural entropic processes that gave rise to DNA, the idea that DNA might gain access to its own information is not bizarre. Achieved through a suitable phenotype, that information may be applied at a macro rather than molecular level. The resulting technology will then provide the necessary entropy differential.

Step 4: Determine the developmental process.

Given: The capacity to construct, characterised by drawing (or making mud pies), emerges after language and progresses through childhood in concert with prefrontal neurological maturation. Piaget apart, the study of construction has been very limited, mainly because the bulk of educational research in the early and primary years has been focussed on literacy and numeracy. The potential for the Turing medium (computer) to transform primary education makes timely the opening of a new chapter in psychology: the genetic epistemology of technicity.

Step 5: Apply

In the book-bound schoolroom there is little opportunity to exercise emerging technicity. Drawing, writing and arithmetic do exploit it in a constrained two-dimensional sort of way. Young children are encouraged to play with constructional materials, bricks, and, post school, many adults live by construction (with bricks, beams, wheels and connectors. Were the core curriculum of literacy and numeracy to be taught more efficaciously using Turing media then time would be available to keep constructional continuity. As it is, children have only games in the playground through which to develop the type of technicity skills that use information from within the nervous system. As they grow they will, without educational intervention, use this same information when buying/making perfume, cooking/choosing food, buying/creating clothing, and so forth. In school the basis for these adult skills is considered vocational rather than academic yet the only difference is the medium. We have an education system that binds children's mind as cruelly as once were the feet of Oriental women.

It is possible to create a workshop environment in which the core of literacy, numeracy, and science are taught creatively and cognitively. Such an environment is the one I am creating at the British School of Sofia (see snap below). Within a LEGO system environment teaching starts from first principles and children learn to build stable and attractive models of their world ntegrated into scenes and situations, Choice of colour and the creation of patterns and shapes within the constraints of the system develops three dimensional thinking. Lights offer an inexpensive introduction to control and electricity. Motors and sensors later add a wider range of possibilities, BUT the system used is the old LEGO Dacta Control Lab and Logo (windows version) because it complements the language and maths curriculum in a way that more recent offerings cannot.

Step 6 Bury philosophy

All sciences have been through their natural philosophy phase. Education remains stuck in the philosophical mire. Piaget complained about this in the 1950s, Papert mentioned it in the 1990s. It is now time to move forward and the Technicity notion offers a staring point.

What's New in Snap! and BJC

Brain Harvey, bh@cs.berkeley.edu
University of California, Berkeley

Abstract

Snap! news includes keyboard-based script editing (the beginning of support for visually impaired users), the ability to include Javascript code in a Snap! block definition, and (thanks, Citilab!) the ability to create standalone applications from projects.

BJC news includes two major curriculum efforts: BJC is now live on the edX platform, and a substantial rewrite with supporting teacher materials is under way to make BJC easier to use and more successful in high schools. Also, we have taught BJC to the first 28 of a planned 100 New York City public high school teachers, and they are teaching students (mostly 17-year-olds) this year.

Keywords

BJC, Beauty and Joy of Computing, Snap!, edX, New York City

Workshop on Game Programming in Scratch

Tomohito Yashiro, *g3115003@fun.ac.jp*
Graduated School of System Information Science, Future University Hakodate

Kazushi Mukaiyama, *kazushi@fun.ac.jp*
Department of System Information Science, Future University Hakodate

Yasushi Harada, *haraday@fun.ac.jp*
Department of System Information Science, Future University Hakodate

Abstract

In this paper, we introduce the results of game programming from a Scratch workshop for children. Scratch is a programming language for children (Resnick et al., 2009).

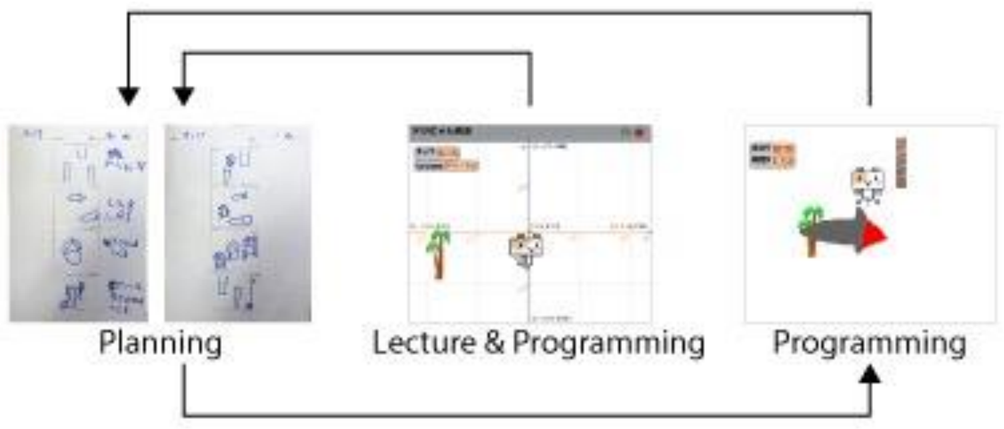


Figure 1. Schematic diagram of the workshop procedure

In the workshop, children first brainstormed ideas for computer games with a facilitator. Second, they planned how to program the games. Third, they programmed their own, original games following their plan. Finally, the children planned future work on their game. It is our hope that this workshop provided the children new experiences in planning and execution. The workshop results showed that game programming was good for the children, as it encouraged their interest and motivation. Additionally, the children were interested in each other's work. In the stage of the workshop where the children planned their work, we represented their ideas on paper. The paper enabled the workshop staff to visualize the children's ideas and the appropriate programming procedures. Additionally, the children used the paper to describe their ideas easily because the facilitator showed a model of planning in the programming lesson using the paper. We concluded that having a tool to visualize the children's thinking was effective in the programming workshop. In addition, we considered that it is important to prove the usefulness of the tool through its use by the facilitator.

Keywords

Workshop, Programming, Scratch

Workshop Results

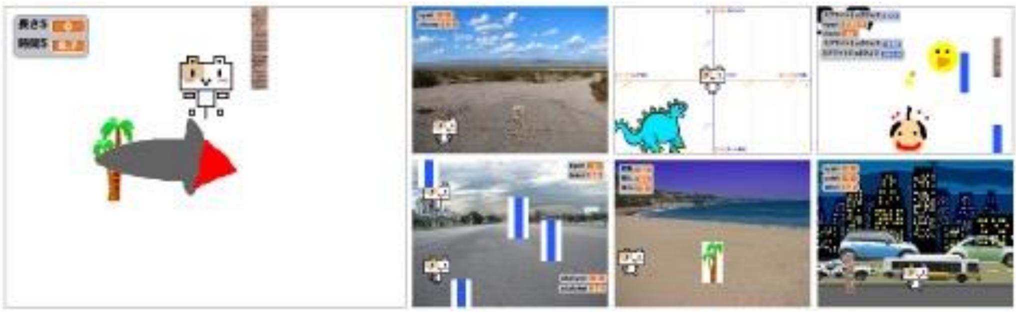


Figure 2. Screenshots of the children's games

The purpose of the workshop is for children to engage in programming and planning to develop computer games. The workshop was held in Hakodate City's Citizen Center. Seven children participated: 6 boys and 1 girl, ranging in age from 9 to 12. All of them were in primary school. During the workshop, the staff helped the children with their tasks. Figure 1 shows the workshop's procedure. The workshop was divided into four parts: the lecture and programming stage, first planning stage, programming stage, and second planning stage. In the lecture, we used an existing theme in game programming, a hurdle race game (Abe, 2013). We explained how to program the game to the children through a demonstration to facilitate their understanding and practice. Simultaneously, we had the children imitate the programming demonstration. Then, we used the same paper where the children had recorded their ideas to explain the game's function. In the first planning stage, the children wrote their game ideas on paper with a staff member. Next, they drew objects on a screen and explained the objects' or the game's function in a few sentences. In the programming stage, the paper enabled the staff to visualize the children's ideas and the appropriate programming procedures. Therefore, we were able to help each other quickly because we understood the children's intentions through the visualization. Figure 2 shows screenshots of the children's work. Each child presented his or her own completed game to the other children. In the second planning stage, the children clearly described their plans for future work on paper. They described some ideas that had not been realized in the workshop. They also described their games' functions, such as consumer games. We reached the following conclusion by analyzing the children's behavior during the workshop. We initially believed it would be difficult for the children to plan how to program a game. However, we found that the children were able to plan and execute programming. Therefore, we concluded that game programming is a good theme to facilitate children's ideas and learning activities. Additionally, we believe that the children's observation of the facilitator's model helped them describe their ideas on paper. Since the paper for describing ideas was very useful in presenting a visualization of the children's plans, we concluded that having a tool to visualize the children's thinking was effective in the programming workshop. Moreover, we considered that it is important to prove the usefulness of the tool through its use by the facilitator. Finally, the children were interested in each other's work, as they often played each other's games in the workshop. Therefore, we concluded it is important that children have others use their own program.

References

Abe, Kazuhiro. (2013) *Shogakusei kara hajimeru wakuwaku programming (Exciting programming to start in elementary school)*. Nikkei Business Publications, Tokyo.

Resnick, Mitchel, Maloney, John, Monroy-Hernandez, Andres, Rusk, Natalie, Eastmond, Evelyn, Brennan, Karen, Millner, Amon, Rosenbaum, Eric, Silver, Jay, Silverman, Brian, and Kafai, Yasmin. (2009) Scratch: Programming for all. *Communications of the ACM*. Vol. 52, No. 11, 60-67.

Analyzing Twitter Data using Snap!

Andreas Grillenberger, *andreas.grillenberger@fau.de*
Computer Science Education Research Group, Universität Erlangen-Nürnberg

Ralf Romeike, *ralf.romeike@fau.de*
Computer Science Education Research Group, Universität Erlangen-Nürnberg

Abstract

In this paper, we present a software tool which enables students to discover and learn about data stream systems (DSS) in a constructionist way. As DSS are part of the ongoing developments and innovations in data management, which are often summarized by the term *Big Data*, this tool shows in an exemplary way how these complex topics can be incorporated into school teaching. Therefore, we extended the block-based programming environment *Snap!*, so that it supports analyzing the Twitter data stream even without having any pre-knowledge on data stream analysis.

Keywords

data stream system; data analysis; block-based; software tool; big data

Introduction

Today, everyone is generating large amounts of data and is always confronted with the results and implications of data analysis. Not only smartphones and websites are producing and processing data continuously, but data are also relevant to many other parts of the daily life. Despite their relevance for everyone (cf. Grillenberger & Romeike 2015), these topics seem hard to understand because of their complexity. Nevertheless, various ideas that are central to these topics can be identified: for example, causality or correlation as analysis paradigm, the analysis methods used, (in particular categorization, clustering and association analysis), consistency, redundancy and parallelization.

For discussing these concepts at school, appropriate software tools are required in order to let students gather their own experiences with these systems. While in other teaching contexts, often professional tools are also used for teaching purposes, typical tools in the context of "Big Data" are too complex for providing them to students without extensive further support. Instead, finding ways for reducing the complexity is important for allowing students to build up own knowledge on such systems in a constructionist way.

Hence, we prepared the topic *data stream systems* in a way, that students can discover and understand the efficiency of this modern data analysis approach, the accompanying threats but in particular also the chances and possibilities that are opened up by these systems on their own. The software tool presented in the following supports the design of own data analyses without pre-knowledge on data stream systems, the underlying concepts and ideas or the tool itself. In this way, the presented approach is prototypical for incorporating the complex and innovative topics in data management / Big Data for (secondary) computing education.

The *Snap!* Data Stream Extension

For enabling students to conduct own data analyses, we extended the block-based programming environment *Snap! (Harvey and Mönig 2015)* in order to support working with and processing of data streams. We decided for using *Snap!* because of its low barriers to entry, but also because it provides diverse possibilities and is easily extendable. Hence, the students can not only use the provided functionalities, but also extend them on their own in order to provide additional possibilities. As data stream source, we selected Twitter, as this social network provides a huge amount of data easily accessible via a programming interface. Also, these data can be used without data privacy risks, as all the accessed data are also available via Twitter's public website; the students can hence not only access the tweets text, but also e. g. data on the user, followers or location data.

All new implemented blocks are based on original *Snap!* blocks only. Hence, they can be used in the original *Snap!* installation and most of its derivatives. Also, students can take a look behind the implementation and customize it, as the blocks may be edited as all other own blocks. For restricting the creative openness as less as possible, we allow accessing all the attributes provided by Twitter for each tweet. Thus, students have much creative freedom when analyzing the Twitter stream.

For visualizing the analysis results in an easy and clear way, we also implemented two visualizations: a bar chart (cf. fig. Figure 11), which is e. g. suitable for showing the results of categorization tasks, and a map (cf. fig. Figure 12) that can be used for visualizing spatial data. Both visualizations can also be used as example implementation for additional visualizations, which can also be implemented by the students on their own. In fig. Figure 10, we show an example program for categorizing all the incoming tweets by language; this code generates the image shown in fig. Figure 11.

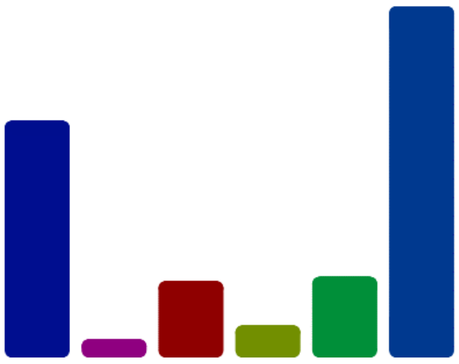


Figure 11. Bar chart visualization

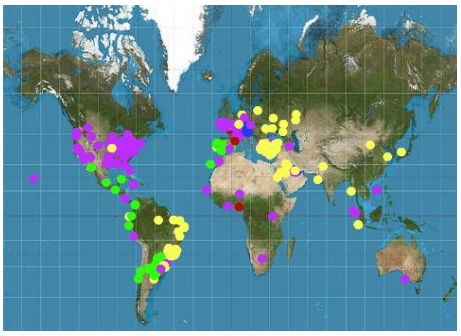
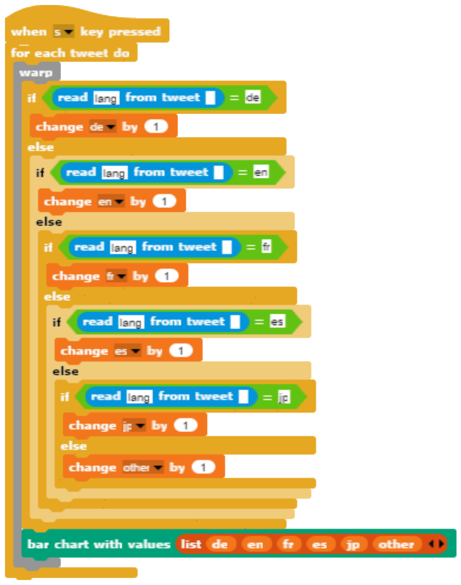


Figure 12. Map visualization

(map image ©2011 Strebe, CC-BY-SA 3.0)



```
when s key pressed
for each tweet do
  warp
    if read lang from tweet  = de
      change de by 1
    else
      if read lang from tweet  = en
        change en by 1
      else
        if read lang from tweet  = fr
          change fr by 1
        else
          if read lang from tweet  = es
            change es by 1
          else
            if read lang from tweet  = jp
              change jp by 1
            else
              change other by 1
    bar chart with values list de en fr es jp other
```

Figure 10. Bar chart visualization

Conclusions

With the simple-to-understand programming environment *Snap!* and by restricting to the basic ideas and concepts, we can break down the complex topic data stream systems to a level suitable for students. The extensibility of *Snap!* provides the advantage that despite the low entry level, also complex analyses are possible. This is further supported by basing on the real and infinite data source Twitter. The relevance and attractiveness of topics like data stream system can be further emphasized by incorporating other innovations: e. g. with the upcoming *Internet of Things (IoT)*, huge amounts of data are generated and analyzed—and it can be assumed that data stream systems will also have high importance there. By combining the described tool with for example objects created in physical computing projects, the *IoT* can be reconstructed in small-scale.

References

Grillenberger, Andreas; Romeike, Ralf (2015) *Analyzing the Twitter Data Stream Using the Snap! Learning Environment*. In Brodnik, Andrej; Vahrenhold, Jan (Eds.) Informatics in Schools. Curricula, Competences, and Competitions. Springer International Publishing.

Harvey, Brian; Mönig, Jens (2015) *Snap! Programming Language*. http://snap.berkeley.edu

JumpSmart: A Platform for Communal Making and Physical Engagement in Programming

Mikhaela Dietch, *mikhaela@students.olin.edu*
EASE lab, Olin College of Engineering

Bryanne Leeming, *bleeming1@babson.edu*
Master of Business Administration, Babson F.W. Olin School of Business

Amon Millner, *amon.millner@olin.edu*
Assistant Professor of Computing and Innovation, Olin College of Engineering

Abstract

In this demonstration we present our work-in-progress: a platform for creating called JumpSmart. We are developing JumpSmart to inspire new explorations in patterns and programming logic through open-ended activities for young learners (ages 7-14).

JumpSmart is designed to help young learners acquire new knowledge, abilities and insights through the construction of personally meaningful artifacts. The digital and physical technology of our system is built so that learners can engage in technical processes which resonate with their non-technical interests. Interactions that incorporate novel forms of making can bring computing to more communities and broaden who participates. Programming needs many methods for entry (Turkle, 1992) (Resnick, 1998) and we posit that physical activity is a worthwhile gateway to explore.

We intend for JumpSmart to promote learning through physical activity and extend programming to a wider array of young learners. The kinds of play possible with JumpSmart further the idea of using computers as chalk to rethink how computation can be used in making, through engaging, expressive, and transformative experiences (Millner,2012).

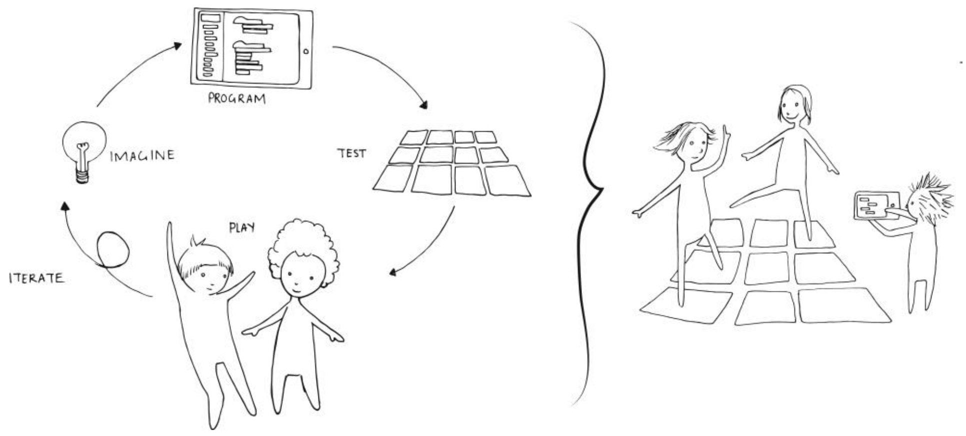


Figure 1. Above is a sketch of JumpSmart interactions. Young learners code interactively while exploring physical and communal play. The JumpSmart matrix of modular floor tiles is shown sensing pressure and lighting up according to the rules of a program. Players design programs for the tiles using a blocks-based environment, creating new patterns of interaction with their peers. They participate in a continuous cycle: imagine, program, test, play, and iterate.

Keywords:

Physical programming; Toolkit; Whole-body interaction; Blocks-based environments

Playing with JumpSmart

To highlight the intended uses of our platform, we have created a programmable physical simulation of the playground game 'foursquare' as a prototype of future interactions. This game, called Four Squares, shows how JumpSmart could afford socially meaningful contexts and imaginative play.

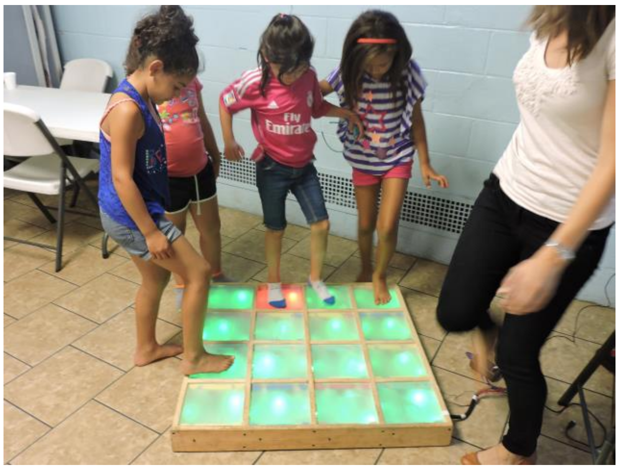


Figure 2. Four Squares can be played with two to four players, each owning a square (a 2x2 matrix of tiles) of the playing surface. A ball, digitally represented as a red LED-lit tile, randomly bounces from square to square while the players attempt to catch it by stepping on the ball. If a player misses the ball in any round, they are out. The last player to miss the ball is considered the winner, and speed increases in the next round while everyone tries again.

Four Squares

Four Squares is programmed in a Scratch-like interface, and is a starting point to initiate the programming process. By playing with the physical and digital components of the game, young learners can begin to explore concepts of Boolean logic, for loops, and other foundational computing concepts by iterating on Four Squares or beginning a new program.

Communal play creates an engaging context for creating programs. Through social engagement, players iterate together to further their ideas and realize their imagined game. In our testing, young learners have designed everything from new interpretations of well-known games to visual mazes. JumpSmart is intended for the expression of imagination. It is transformative as it affords iterative design, beginning with basic interactions and evolving as the young learners are empowered to consciously engage with their learning process to create new digital and physical experiences.

References

Millner, A. (2012). Proceedings from Constructionism '12: Computer as Chalk: Supporting Youth as Designers of Tangible User Interfaces, Athens Greece

Resnick, M., & Rusk, N. (1996). The computer clubhouse: helping youth develop fluency with new media. Paper presented at International Conference on Learning Sciences (ICLS '96): International Society of the Learning Sciences (285-291). Edelson, D.C., Domeshek, E. A. (Eds.)

Turkle, S., & Papert, S. (1992). Epistemological Pluralism and the Revaluation of the Concrete. *Journal of Mathematical Behavior*. 11, 1, 3-33.

Learning About Complex Systems with the BeeSmart Participatory Simulation

Yu Guo, *bguo@u.northwestern.edu*
Learning Sciences, Northwestern University

Uri Wilensky, *uri@northwestern.edu*
Learning Sciences and Computer Science, Northwestern University

Abstract

We propose to demonstrate the third iteration of a design-based research study—BeeSmart—on students' learning about complex systems. Previously, we presented the second iteration—a microworld for honeybees' hive-finding behavior (Guo & Wilensky, 2014). We found that students had difficulties with random interaction between bees in this complex phenomenon (Guo & Wilensky, 2016). Here we introduce our design of a whole-class participatory simulation activity consisting of a HubNet model and a set of inquiry activities to scaffold students' learning about random interaction. This work enriches the literature of designing constructionist learning environments to scaffold students in learning complex systems.

Keywords

participatory simulation; constructionist learning environment; complex systems; randomness

Introduction

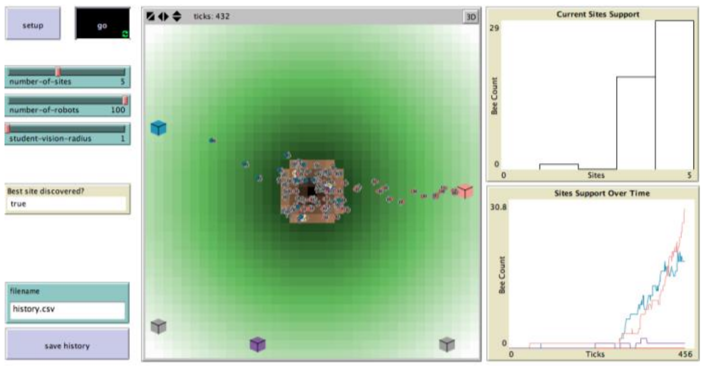
Complex systems are hard for students to understand. One of the major sources of difficulty is randomness involved in these systems (Wilensky & Resnick, 1999). When explaining complex phenomena, students tend to attribute agents' interactions as intentional and consistent with the systems' goals. In fact, agents follow very simple rules, which are not directly aligned with the systems' goals. For example, the overall goal of a swarm of honeybees is to find the best hive site nearby. However, the individual bees' do not care about this goal. When a bee discovers a potential hive site, she reports it to the swarm through a figure-8 dance. The on-watching bees in the swarm randomly follow a dance to further investigate the site advocated by the dance. When they return, they in turn dance for the site to recruit more watchers to further investigate the site. Dances for higher quality sites last longer, so they have a better chance to be followed. Iterations of such process can eventually lead the whole swarm to agree on the best hive site.

In our previous study, we found that students thought that individual bees needed to look for the longest dance in the swarm in order to ensure they end up with the best hive. Students did not know that bees do not have the abilities to observe the whole swarm and pick out the longest dance. They did not see that patterns can emerge from random interactions (Guo & Wilensky, 2016). To help students overcome these obstacles, we designed a participatory simulation.

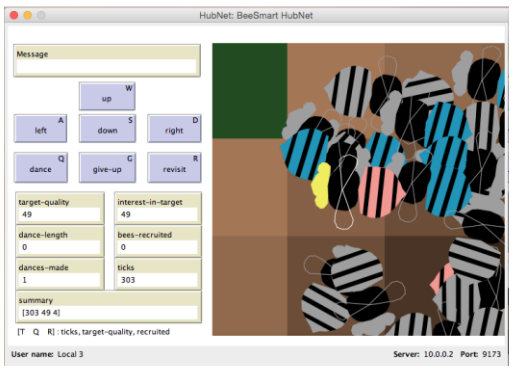
BeeSmart Participatory Simulation

Participatory simulations allow students to project themselves into models that can be executed. In a participatory simulation, students act out the roles of individual elements of a system and then observe how the behavior of the system as a whole can emerge from these individual behaviors (Wilensky & Stroup, 2000). Participatory simulations for honeybees' behavior are not new. For example, BeeSim (Peppler et al., 2010) is designed for young children to learn about honeybees' nectar collecting behavior through digitally augmented wearable bee puppets. Our work is different in that we target a different audience (high school students) with a different phenomenon (hive finding) through a different technology (HubNet). HubNet is 'an open client-server architecture, which enables many users at the "Nodes" to control the behavior of individual objects or agents and to view the aggregated results on a central computer known as the "Hub".' (Wilensky & Stroup, 2000, p. 286). This technology fits our goal of investigating students' difficulties with random interaction in complex systems.

In our BeeSmart participatory simulation, students act out the role of individual bees seeking for hive sites in the surrounding virtual landscape. The "nodes" are computers from which students control their avatar bees. The "hub" is the teacher's computer hosting the virtual world where the bees interact. The world, along with plots of aggregated data from individual bees, is only projected to the whole class during discussions (Figure 1). The class activity starts with a driving question: How can a swarm of bees pick the best hive site from many choices? In the first round, the teacher sets the bees' vision radius to maximum, so the entire virtual world is visible from a bee's perspective. Students can control their bees' movement and drive them directly to potential hive sites, inspect them, and dance to report to the swarm. In subsequent rounds, the teacher continuously reduces the vision radius to limit the information a bee has access to (Figure 2), which provokes students to think about real bees' limited abilities and the rules they follow. The histogram and line graph show aggregated data of all students' play. During discussions, students use their experience as bees to make sense of these data and patterns.



number-of-sites 5
number-of-robots 100
student-vision-radius 1
Best site discovered?
true
filename
history.csv



Message

target-quality
49
interest-in-target
49
dance-length
0
bees-recruited
0
dances-made
1
ticks
303
summary
[303 49 4]
[T Q R] : ticks, target-quality, recruited
User name: Local 3
Server: 10.0.0.2 Port: 9173

Figure 1. Teacher's view. Scouts are flying out from the swarm at the center to explore two hive sites (blue on the left and pink on the right)

Figure 2. Student's view. The pink bee at the center is controlled by the student at this node. It is dancing to advocate for the pink site

A pilot run with graduate students shows that changing the radius of students' vision changes their experience and the dynamics of their conversations. This design will be implemented in high schools in the near future to further investigate students' difficulties with random interaction.

References

Guo, Y., & Wilensky, U. (2014) Beesmart: a microworld for swarming behavior and for learning complex systems concepts. Proceedings of the Constructionism 2014 Conference. Vienna, Austria. August 2014.

Guo, Y., & Wilensky, U. (2016) Students' difficulties with randomness in complex systems—A design-based research study. Poster to be presented at the AERA Annual Meeting (SIG: Learning Sciences), Washington, DC. April, 2016.

Peppler, K., Danish, J., Zaitlen, B., Glosson, D., Jacobs, A., & Phelps, D. (2010). BeeSim: leveraging wearable computers in participatory simulations with young children. In Proceedings of IDC 2010 (pp. 246–249), Barcelona, Spain. June 2010.

Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and Technology*, *8*(1), 3–19.

Wilensky, U., Stroup, W. (2000) Networked gridlock*:* Students enacting complex dynamic phenomena with the HubNet architecture. In Proceedings of the Fourth Annual International Conference for the Learning Sciences (pp. 282-289). Mahwah, NJ. June 2000.

Pyonkee: A Scratch compatible visual-programming environment running on iPad

Kazuhiro Abe, *abee@si.aoyama.ac.jp*
School of Social Informatics, Aoyama Gakuin University
SoftUmeYa, LLC

Masashi Umezawa, *ume@softumeya.com*
SoftUmeYa, LLC

Abstract
Pyonkee is a visual-programming environment running on iPad. It is based on Scratch 1.4 from the MIT Media Laboratory. Since Pyonkee is fully compatible with Scratch 1.4, millions of existing Scratch projects can be used for reference. The Pyonkee user interface is optimized for touch devices. We do not need cumbersome typing, even a mouse. Just write programs wherever you like. Pyonkee nicely supports pinch-in/out, and font scaling for small devices. Moreover, sound recorder and camera are provided for importing your sounds and pictures into the projects. We can mix various media on Pyonkee and program them.

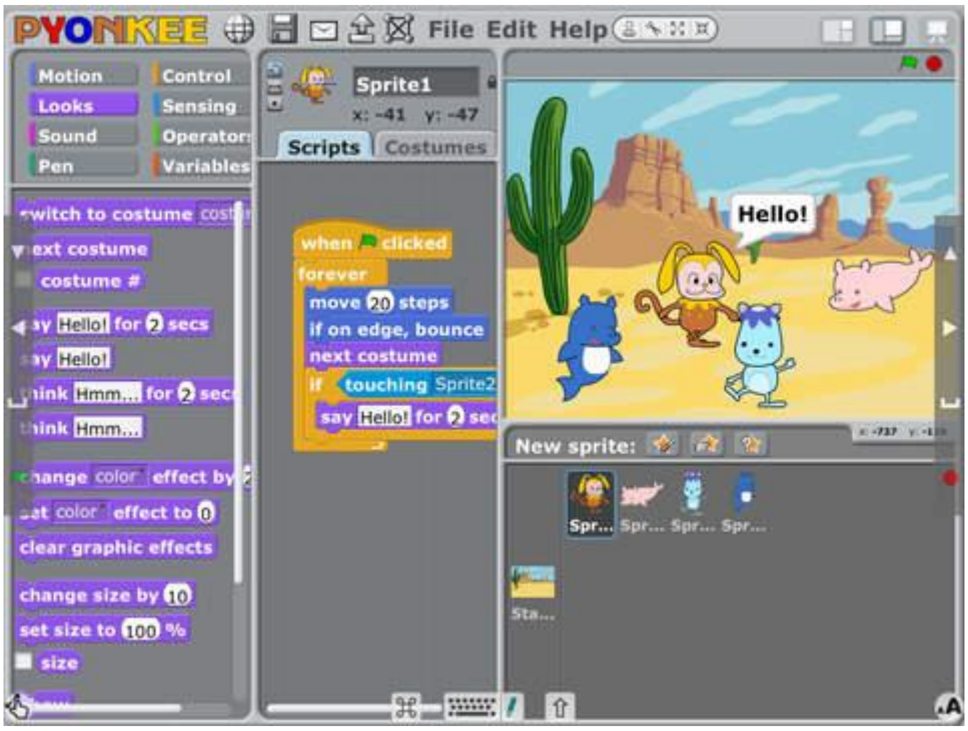


Figure 1. Pyonkee running on iPad

Keywords
Scratch, visual programming, tablet, STEM, physical computing, iPad

Introduction of Pyonkee

Scratch [1] is a popular visual-programming environment for children developed by the Lifelong Kindergarten Group of MIT Media Lab. It runs on Windows, Mac OS X, and Linux. The main user of Scratch is eight years old and older children. Millions of users are using Scratch in the world, and they are sharing ten millions of projects.
In recent years, tablet device became an important tool for STEM in the school. However, Scratch did not support tablet devices for many years. MIT Media Lab, Tufts University, and PICO released ScratchJr in 2014 for iPad and Android.
ScratchJr [2] is a subset of Scratch made for five to seven years old children. ScratchJr has very limited command blocks. For example, it lacks turtle graphics and conditional branching blocks. Its simple iconic blocks without text label are easy to understand for young children. But it does not substitute for Scratch to make more complex projects.
So we developed Pyonkee [3] for iPad. It is based on the open-source code of Scratch 1.4. And we got many ideas from the open-source code of Scratch.app for iOS [4] written by John M. McIntosh.
We cannot use the name of Scratch, logo, and Scratch cat under the provision of the Scratch Source Code License from MIT. However, it is a virtually Scratch. It can run any projects made from Scratch 1.4.
Pyonkee is available on AppStore for free of charge. And its source code is on GitHub as an open-source [3].

Advanced features of Pyonkee

Pyonkee has some advanced features in addition to the function of Scratch 1.4.

- Optimized GUI for the tablet, e.g. pinch-in/out, font scaling, large buttons, software keys, etc.
- Project sharing by AirDrop including Dropbox. The teacher can distribute samples and pupils can share their projects easily in the classroom.
- Access to built-in sensors of iPad from sensor blocks, e.g. accelerometer, gyro, compass, light, sound, camera, etc. It makes possible to make tangible and physical computing things.
- Supporting Mesh [5], an experimental distributed computing mechanism, hidden function of Scratch 1.4. You can send messages and share global variables with multiple devices connected Wi-Fi.
- Additional costumes and backgrounds from PeKay [6] donated by Tamie Asakura.
- Any device which supports Scratch remote sensor protocol works with Pyonkee without modification via Wi-Fi, e.g. Romo, Sphero, Kinect, Raspberry Pi GPIO, etc.

Practices of Pyonkee

Many schools in the world adopt Pyonkee for their classroom. The followings are examples.

- Grade 3 & 4 Games - Con, Mildura West Primary School, Australia
- How to program their own “health” quizzes, Rajabhat University’s Demonstration Primary School, Thailand
- Children’s Technology Challenge, ITOCHU Techno-Solutions Corporation, Japan
- Personal Computer Club, Komae Daisan Elementary School, Japan
- Creative Coding, Ferndown First School, United Kingdom

References

[1] Scratch - Imagine, Program, Share, https://scratch.mit.edu/
[2] ScratchJr – Home, http://www.scratchjr.org/
[3] Pyonkee - Visual Programming with iPad, http://www.softumeya.com/pyonkee/
[4] Scratch, a Story for iPhone & iPad, https://mobilewikiserver.com/Scratch.html
[5] Mesh - Scratch Wiki, http://wiki.scratch.mit.edu/wiki/Mesh
[6] PeKay’s Little Author for iPad, http://www.pekay.jp/pkla/ipaden

Tatrabot - a mobile robotic platform for teaching programming

Pavel Petrovič, *ppetrovic@acm.org*
Dept of Applied Informatics, Comenius University, Bratislava

We have developed a low-cost robotic platform that we use to teach first grade undergraduate students programming in C language. After the study program in Applied Informatics has been redesigned, our students learn programming in high-level programming language Python, which has automatic memory management, advanced data-structures and algorithms. However, some of our more advanced courses, such as Operating Systems show that the students lack the understanding for the bits and bytes, low level programming and miss the connection between the actual processes that occur in hardware and their software being executed on a machine. We think it is important to understand this level too, if the student is to program efficiently and correctly. Furthermore, playing with robots stimulates and motivates freshmen students, makes the study program more attractive, and allows more playful learning, improves the overall experience and diminishes the shock from entering the University. In this demonstration, we will show you the robot, its capabilities and how we have used it in our course on Computer Hardware.

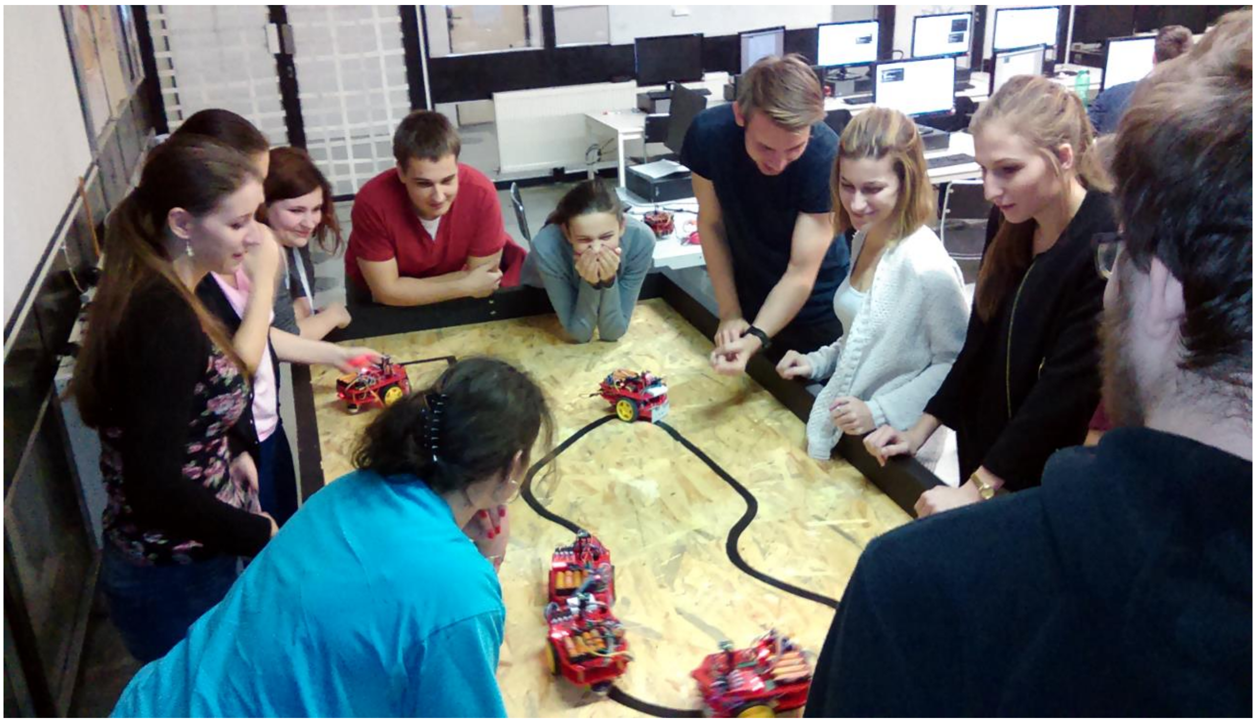


Figure 1: Student exercise, participants programmed their robots to follow line and navigate a maze.

Keywords

robotics platform, constructionism, C language

The platform

The base of the robot consists of two plastic plates with many small holes that allow easy mounting of additional equipment. The plates are connected with usual metal spacers giving the construction a solid and robust appearance. Simple wheels are propelled by two DC motors, base is supported by a supporting rolling sphere. All of this comes as 2WD Beginner Robot Chassis V2 from DAGU China, available from RobotShop for about 15 Eur. The control platform is a tiny board with 32-bit microcontroller STM32F103 with a bootloader. It provides many more features than a popular 8-bit Arduino platform, it is available for less than 4 Eur, its CPU runs on more than 3-times higher frequency, and it can be programmed in standard C using free, but very convenient development environment ChibiStudio based on Eclipse IDE. The included library ChibiOS – provides interfacing with most of the devices that are already available on the single chip microcontroller such as timers, PWM controllers, input capture units, I2C bus, CAN bus, SPI, serial ports and more, making it very easy to attach sensors and actuators of all kinds with less technical effort. In our case, we have connected the following: SHARP distance sensors, two IR obstacle sensors, 4-channel bottom line sensor, motor encoders with wheels cut by laser in a local FABLab since those provided by DAGU are IR transparent, a panel with six signal LEDs, a reset and program button, simple piezo sound output, two user buttons, and an inertial unit that includes 3DOF accelerometer, gyroscope and magnetometer, temperature sensor and barometer. Programs are downloaded over USB cable that allows communication in terminal or any other application that can communicate over serial port such as Imagine Logo. The robot also has a BlueTooth allowing for remote control, easy debugging, and remote data collection. A small-size breadboard allows easy plug & play attachment of further sensors, such as ultrasonic distance sensor. Everything is powered by a set of 4 AA cells, giving at least 5 hours of operation. The complete system is built for about 65 Eur requiring less than 10 hours for assembly. We have purchased parts for 30 units, and we are about half through the assembly process, allowing for a use in a group of 20 students.

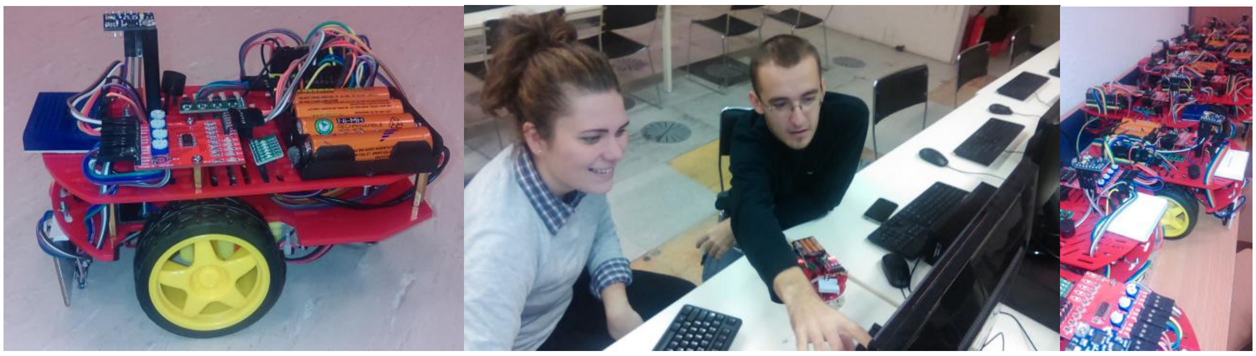


Figure 2: Tatrabot from side (left), students programming Tatrabot (middle), Tatrabots resting (right).

Experiences and Conclusions

The robots have been used in one freshmen course, a robotics seminar, where students made a program to solve sokoban puzzles and in an individual work with talented students. We have a set of functions that provide easy interfacing with the installed sensors. For instance, the current value of the left and right wheel encoders are in the global variables countA, countB, the current reading of the SHARP distance sensor is always available in the variable distance, to control speed of motors, one just needs to call a function set_motor(motor, speed), where the speed is 0-10000. We find the platform easy to use, easy to understand, but its advantage is that it is not a toy. We give the students a real thing. Both software and hardware is completely open, it is built from parts that can be used in industrial applications. All the learned material allows students to immediately continue developing their own embedded applications for a very low price. Yet, we adhere to the principles of constructionism. Students learn by doing and playing, they can come with creative designs, solutions, projects, interact in groups and have fun doing it. The documentation site for this robot is dai.fmph.uniba.sk/projects/tatrabot/

And now for something completely different: ToonTalk - a programming language that is not textual, block-based, or procedural

Ken Kahn, *toontalk@gmail.com*
Academic IT Services, University of Oxford

Abstract

The motivation behind this workshop is to introduce some different ways of thinking about computation. As Marvin Minsky wrote “If you understand something in only one way, then you don't really understand it at all” (Minsky, 1984).

Nearly all constructionist programming languages are based upon the idea of users defining procedures. In some cases those procedures are associated with sprites or objects but the range of computation models is narrow. Many languages support concurrent computation but the ways in which these parallel computations communicate and synchronise is limited. Logo pioneered a syntax that was easier for beginners to read, write, and understand. Scratch and other block-based languages introduced a new kind of syntax that makes the construction of programs simpler and much less error-prone. Text and blocks, however, are not the only kinds of syntax possible.

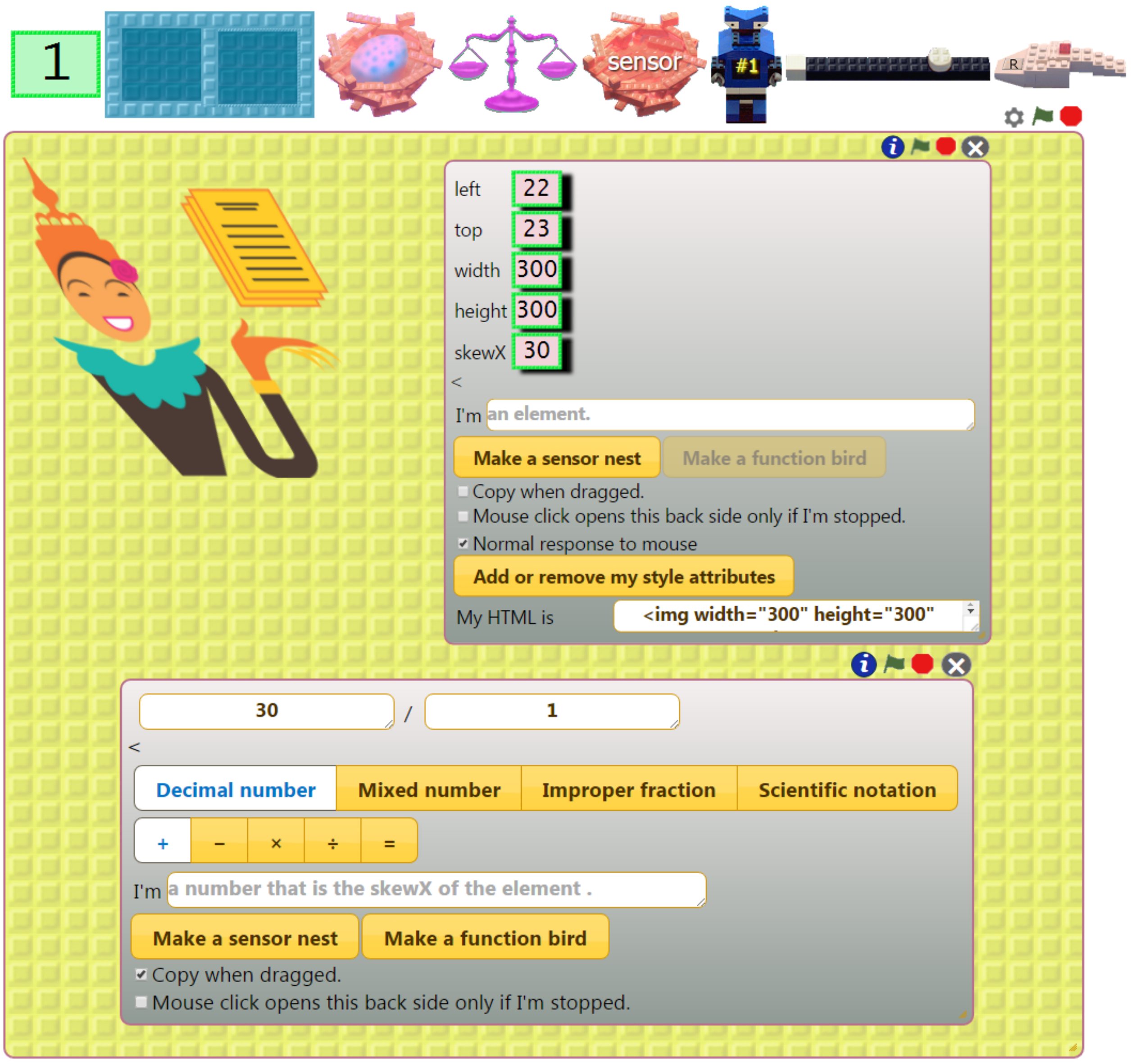
Seymour Papert once described Logo as the result of “child-engineering” the best ideas in computer science. In 1967 the Lisp programming language exemplified these ideas and strongly influenced the design of Logo. Computer science today contains many other ways of describing computation based upon logic, constraints, actors, higher-order functions, data flow, or rules.

ToonTalk (Kahn, 1996) grew out of the goal to once again “child-engineer” the best ideas in computer science. Concurrent constraint programming (Saraswat, 1993) was chosen as the framework for the design of ToonTalk. Another set of computer science ideas that were crucial to the design of ToonTalk was the work on programming by demonstration or by example (Kahn, 2001). Another idea is that the syntax of a program could be not only visual but animated.

This workshop will introduce participants to ToonTalk Reborn (Kahn, 2014) a new web-based implementation of ToonTalk. It runs on any device with a modern web browser, though it is easier to use on a laptop than with a touch device. More details in the paper *Integrating programming languages with web browsers* in the conference proceedings. No prior programming experience required. Any number of attendees welcome. A projector required. A good Internet connection is very desirable but not necessary.

Keywords

ToonTalk Reborn, constructionist programming languages, animated programming, programming by demonstration, concurrent constraint programming



left 22
top 23
width 300
height 300
skewX 30
<
I'm an element.
Make a sensor nest
Make a function bird
Copy when dragged.
Mouse click opens this back side only if I'm stopped.
✓ Normal response to mouse
Add or remove my style attributes
My HTML is <img width="300" height="300"

30 / 1
<
Decimal number | Mixed number | Improper fraction | Scientific notation
+ | − | × | ÷ | =
I'm a number that is the skewX of the element .
Make a sensor nest
Make a function bird
✓ Copy when dragged.
Mouse click opens this back side only if I'm stopped.

Figure 13. A ToonTalk screen shot

References

Kahn, K. (1996) *ToonTalk - An Animated Programming Environment for Children*. Journal of Visual Languages and Computing, June.

Kahn. K. *(2001) Generalizing by Removing Detail: How Any Program Can Be Created by Working with Examples*, Communications of the ACM, March 2000 and full version in *Your Wish Is My Command: Programming by Example*, edited by H. Lieberman, Morgan Kaufmann, February.

Kahn, K. *(2014) TOONTALK REBORN - Re-implementing and re-conceptualising ToonTalk for the Web.* Constructionism 2014*.* Vienna*.* August*.*

Minsky, M. *(1984) Will Robots Inherit the Earth?* Scientific American*.* October

Saraswat, V. (1993) *Concurrent Constraint Programming*, MIT Press, March.

Considering approaches to research through the lens of constructionism

Carina Girvan, *girvanc@cardiff.ac.uk*
School of Social Sciences, Cardiff University, Wales, UK.

Nathan Holbert, *holbert@tc.columbia.edu*
Department of Mathematics, Science and Technology, Teachers College, Columbia University, USA.

Celia Hoyles, *c.hoyles@ioe.ac.uk*
London Knowledge Lab, Institute of Education, University College London, London, UK.

Chronis Kynigos, *kynigos@ppp.uoa.gr*
Department of Pedagogy, University of Athens, and CTI Diophantus, Greece.

Richard Noss, *r.noss@ioe.ac.uk*
London Knowledge Lab, Institute of Education, University College London, London, UK.

Abstract

This workshop provides an opportunity for the community to discuss and debate the methodologies and methods used in constructionist research, critiquing existing approaches in relation to both pedagogy and philosophy of constructionism. While there may be numerous texts on conducting educational research, there has been little consideration of how pedagogical theory or philosophy should inform a researcher's approaches. In this workshop, constructionism is the lens through which we explore methodology, methods and tools. We explore several ways in which the process and outcomes of learning through constructionist activities can be uncovered, considering the role of approaches such as design research, randomised control trials and case studies, and looking at data sources such as interviews, data logs, artefacts and observations. The aim is to construct both individual knowledge and a communal understanding of the opportunities and tensions of the approaches we use and the process being studied and is suited to all those with an interest in researching constructionist learning experiences.

Keywords

Constructionism; pedagogy; philosophy; research; methodology; methods

Workshop

While there may be numerous texts on conducting educational research, there has been little consideration of how pedagogical theory or philosophy should inform a researcher's approaches. In this workshop, constructionism is the lens through which we explore methodology, methods and tools. As a core commitment of constructionism is that learning is best represented in the process, rather than the product, studying or evaluating constructionist environments and experiences requires methodologies that provide detailed descriptions of both physical *and* mental constructions in process. We explore several ways in which the process and outcomes of learning through constructionist activities can be uncovered, considering the role of approaches such as design research, randomised control trials, case studies and action research, and looking at data sources such as interviews, data logs, artefacts and observations.

Format

The workshop will be delivered in three phases, beginning with a panel who will discuss some of the approaches used in their own research, their strengths and their limitations as viewed through a constructionist lens. This discussion will provide the initial stimulus for small group discussions facilitated by each of the panel members. Small group discussions will be organized around open questions as well as methodological and research design challenges currently faced and posed by participating members. In the final phase each group will present ideas and solutions that emerged during their discussions.

Participants and Outcomes

This workshop is designed for all those involved in researching constructionist learning with intended outcomes for both individuals and the community. At an individual level it will provide participants with an opportunity to find out about and develop their understanding of different methodologies, methods and tools. Specifically, we hope participants will come away with actionable ideas for employing a variety of methods and techniques to study personally relevant research questions. As a community we aim to build a shared knowledge of the opportunities and tensions in using these approaches within the frame of constructionism, considering how and to what extent constructionist philosophy and pedagogy can or should inform a researcher's methodological choices and modes of analysis. To extend the dialogue beyond the conference a summative report on the session will be written for the broader constructionist community and available via the conference website.

No limit.

Equipment

Participants: No equipment is necessary to participate, however we may use online tools to support back-channel conversations and record ideas within groups.

Presenters: Presenters will require a digital projector, screen and internet access. It would also be helpful if post-it notes, flipchart paper and marker pens were available for the group activities.

Room: Flexible seating.

Constructionist Archaeology - Digging into Papert Papers Lost and Found

Gary S. Stager, Ph.D. *gary@stager.org*
CEO: Constructing Modern Knowledge

Abstract

This Constructionism 2016 workshop will read and discuss overlooked or recently discovered speeches, articles, and papers by Seymour Papert. It is hoped that this will help extend his legacy and invite other scholars to participate in the process of editing, exploring, and examining Papert's early and late period work.

Keywords

Seymour Papert, constructionism, school reform, Logo, learning

Digging into Papert Papers Lost and Found

This community claims Seymour Papert as the father of constructionism and although his constructionist activity ranged from 1968 - 2006, much of the discussion at the past few Constructionism conferences has been focused on the "Mindstorms period" (Papert, 1980) of the early 1980s. This workshop represents an attempt to broaden the lens on Papert's work to the last two active decades of his career and his seminal early publications from the 1960s and early 70s.

Over the past several years, the presenter has transcribed more than a dozen Papert speeches and interviews from 1990-2006. Many of the Papert speeches began as video recordings now being published online. Once the transcripts of these "publications" are fully edited, the text and associated media files will be archived online at The Daily Papert, (Papert, 2000; Stager, 2015) a site curated by the presenter.

In addition to the newly discovered Papert documents being made available, the presenter has been searching for and sharing elusive papers and articles chronicling Papert's earlier foundational work. In addition to academic settings, these documents chronicle Papert's engagement in public discourse - at education conferences and civic meetings.

Participants in this workshop will have the opportunity to explore, read, watch, and discuss "new" pieces of the Papert oeuvre. Powerful ideas gleaned from the works may be shared online or during the presenter's related paper presentation.

It is critical to have fresh eyes approach these documents, not just for the sake of constructionism research and sustaining the legacy of Dr..Papert, but to ensure accuracy and make these artefacts available to the entire world.

The following is a list of the recently unearthed documents available to workshop participants in addition to old documents available in print form.

Document	Year	Video?	Edited?	Audio quality
Keynote address - New Jersey Educational Computing Conference	1990	Yes	no	Poor

Document	Year	Video?	Edited?	Audio quality
Keynote address - New Jersey Educational Computing Conference	1992	Yes	no	Poor
Keynote at National School Boards Conference	1994	Yes	Yes	Excellent
Logosium 1994 speech - MIT	1994	Yes	In progress	Fair
American Prospect panel discussion about impact of Internet on education with Howard Gardner, Mitchel Resnick, Sherry Turkle, and Mitch Kapor	1996	Yes	Yes	Excellent
Looking at Technology Through School Colored Spectacles from The Logo Exchange	1997	No	yes	no audio
Seymour Papert on Generation YES and Kid Power	1998	Yes	In progress	Excellent
Keynote address at MIT Mindfest	1999	Yes	Yes	Fair
A keynote about online learning for a conference in Italy that Seymour didn't attend, but sent this short video	1999	Yes	Yes	Excellent
Keynote at California Computing Educators Conference - Palm Springs, CA	2000	Yes	Yes	Fair
Keynote at National Montessori Society Conference	2000	Yes	Yes	Good
Papert talks about middle school mathematics education - Palm Springs, CA (Mathstar project)	2000	Yes	Yes	Excellent
Millennial Lecture at the Muskey Archives - Bates College, Maine	2000	Yes	Yes	Excellent
Seymour Papert speaks informally with audience of educators at Pepperdine University reception - Palm Springs, CA	2000	Yes	No	Fair
Seymour Papert speech to civic leaders on behalf of a laptop for every child in Maine	2001	Yes	Yes	Excellent
Keynote at 1:1 Computing Conference - Sydney	2004	Yes	Currently being transcribed	Excellent
Sunday Profile with Geraldine Doogue on Australian ABC Radio	2004	No	Yes	Excellent
Afterword: After How Comes What From:	2006	No	Yes	print only
State Department Online Conversation about One Laptop Per Child	2006	No	Yes	none
Speech at Conference in Mathematics Education in Hanoi	2006	No	Yes	Text only

Document	Year	Video?	Edited?	Audio quality
Speech at American Educational Research Association in San Diego	circa 2002	Yes	Yes	Good
Math and Logo Lecture at the University of Maine	circa 2006	Yes	Yes	Good

A full bibliography is in the process of being created for this collection.

References

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. New York: Basic Books.
Papert, S. (2000). Seymour Papert's CUE Conference Keynote Address (transcription). Palm Springs, CA: DailyPapert.com.
Stager, G. (2015). The Daily Papert. Retrieved from http://dailypapert.org

Developing Computational Thinking by Using Constructionist and Deconstructionist Learning

Valentina Dagienė, Vilnius University, Lithuania

Gerald Futschek, Vienna University of Technology, Austria

Gabrielė Stupurienė, Vilnius University, Lithuania

Keywords: Bebras challenge, computational thinking, informatics concepts, learning through a contest, task solving, constructionist and deconstructionist learning.

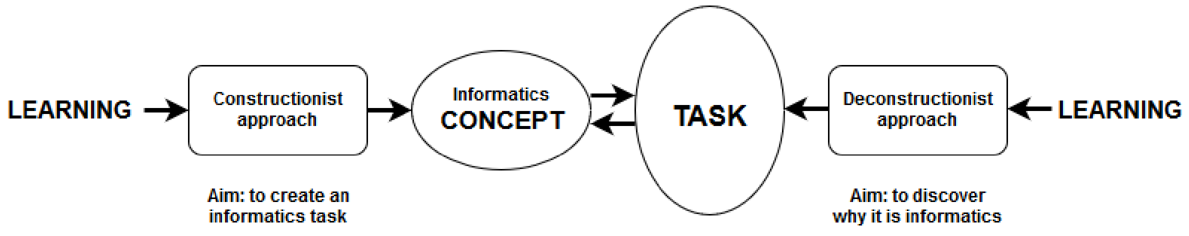
This workshop addresses all educationists and education scientists who are interested how school students can learn informatics (computer science) concepts and Computational Thinking through a contest.

The International Bebras Challenge on Informatics is the world's largest contest on Computational Thinking. In the 2014 contest more than 925,000 students participated in 36 countries of all continents.

The students have to solve from 15 to 21 tasks within 40 to 60 minutes. To solve these tasks, students do not need specific pre-knowledge. Tasks are developed for different age groups, from primary school to upper secondary school students.

The tasks contain concepts of about nearly all areas of informatics. Usually a short story introduces a task and states a problem, termini technici are not used, but to solve the task some kind of computational thinking must be applied. There are tasks about concept categories of information representation, algorithms, programming, logic, encryption and many other.

The other point is teachers' constructionist and deconstructionist learning by creating and explaining Bebras tasks.


LEARNING
Constructionist approach
Aim: to create an informatics task
Informatics CONCEPT
TASK
Deconstructionist approach
Aim: to discover why it is informatics
LEARNING

Items discussed in the workshop

- Operational definition of computational thinking
- Why Bebras tasks can convey computational thinking?
- Which concepts of informatics can be introduced through Bebras tasks?
- How to teach computational thinking using Bebras tasks?
- Teachers' constructionist and deconstructionist learning by using Bebras tasks

In the workshop the participants will learn more about the Bebras Challenge, how the tasks are created, which kind of tasks were produced, what are the effects on learning and teaching.

Participants practically will try to deconstruct the Bebras tasks and find what informatics concepts are hidden. The participants will experience wow-effects while solving Bebras tasks and how thinking is directed to solving strategies that are typical for informatics and computational thinking.

We will discuss how the Bebras Challenge should be performed in a school context and how the teachers may use the Bebras tasks in their teaching activities.

Resources

Each participated country has established its own Bebras website. The international Bebras website www.bebras.org focuses on general information about the contest and tasks. A list of scientific papers and methodological publications related to the Bebras contest is provided (http://bebras.org/?q=publications).

Interactive tasks are the core of the contest. These tasks can be made using different techniques: a tool for making interactive tasks, BebrasLodge, has been developed. Countries use different contest management systems developed by themselves or helpers. Training sessions are available using various systems (and different languages), for example, Lithuanian Bebro varžybų laukas: lt.bebras.lt (training sessions are available in English as well); USA Bebras Challenge: http://www.bebraschallenge.org/; Finnish Majava Kilpailu: http://www.majava-kilpailu.fi/ (available in Swedish as well).

Valentina Dagienė is professor and head of the Informatics Methodology Department at Vilnius University Institute of Mathematics and Informatics. She has published over 150 scientific papers and the same number of methodological works, has written more than 50 textbooks in the field of informatics and ICT for high schools. She has been working in various expert groups and work groups, organizing the olympiads in informatics among students, also engaged in localization of software and educational programs, and problem solving. She is an Executive Editor of international journals "Informatics in Education" and "Olympiads in Informatics". She has participated in many national and EU-funded R&D projects.

Gerald Futschek is a professor at Institute of Software Technology and Interactive Systems at Vienna University of Technology. He is researching and teaching in the fields of Software Engineering and Informatics Didactics. He is highly involved in a variety of national and international contests and competitions that aim to raise the interest of students in the field of informatics. In the field of informatics didactics he tries to answer the question how concepts of informatics such as algorithms and programming may be introduced to students with a low threshold and a high learning success.

Gabrielė Stupurienė is a PhD student in the informatics engeneering program at Vilnius University Institute of Mathematics and Informatics. Her PhD research is focused on informatics concepts. Since 2010 she has been working on Informatics contest Bebras tasks. She has developed and defended her Master thesis “Conceptualisation of Informatics Fundamentals through Tasks” in 2011.

LevelSpace: Constructing Models and Explanations across Levels

Arthur Hjorth, *arthur.hjorth@u.northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Dave Weintrop, *dweintrop @u.northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Corey Brady, *cbrady@northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Uri Wilensky, *uri@northwestern.edu*
Center for Connected Learning and Computer-based Modeling, Northwestern University

Abstract

In this hands-on workshop, we will introduce participants to the recently released LevelSpace NetLogo extension. By using LevelSpace, it is possible to programmatically open NetLogo models from inside NetLogo, essentially treating models like agents. This has a wide and interesting applications for modellers, curriculum developers, and researchers interested in eliciting and studying complex systems and how people reason about them. We will introduce the LevelSpace extension's programming primitives and how to use them by building connected model ecologies. We will talk about the different kinds of ways that phenomena might be connected, and how to model that using LevelSpace. We will also discuss our experience with using LevelSpace in classrooms, and discuss best practices for using it as a tool for studying student reasoning *within* and *between* complex phenomena.

Keywords

Agent based modeling; NetLogo; complex systems thinking; Multi-Level Linked systems

Introducing LevelSpace

NetLogo (Wilensky, 1999) offers learners and modelers a low-threshold, high-ceiling environment for constructing models of, and reasoning about, complex phenomena. NetLogo is widely used in education and research on complex systems (Goldstone & Wilensky, 2008; Wilensky & Rand, 2015) and is one of the most often cited languages in social sciences research (Thiele, Kurth, & Grimm, 2012).

One question we have often been asked – and wondered ourselves – while building models is, "how does the phenomenon in this model relate to the phenomena in other models?" For instance, the NetLogo library already contains two models that seem closely related in the real world: The Wolf Sheep Predation model shows how we get stable ecosystems in which wolves, sheep, and grass interact in a complex food web. The Climate Change model shows how the greenhouse gas effect happens as the interactions between clouds, soil, CO2, and energy from the sun in either the shape of infrared light, or visible light. There are obvious ways in which these two phenomena interact in the real world. For instance, the Climate Change model has a temperature indicator which could affect how quickly grass grows back in the Wolf Sheep Predation model. The Wolf Sheep Predation model both produces greenhouse gases (animal flatulence) and absorbs them (respiration of grass), which would affect the amount of greenhouse gases in the atmosphere in the Climate Change model. But so far it has not been possible connect them computationally. That is, until now!

With LevelSpace (Hjorth, Head, & Wilensky, 2015), it is possible to connect the any number of models – we have run as many as a couple of thousand models simultaneously. LevelSpace is built as a NetLogo extension, using NetLogo's Extensions API and the Controlling API. It is released as Open Source, and full documentation and source code are freely available on GitHub. It extends the NetLogo language with a set of primitives that allow modelers, learners, and curriculum designers to control NetLogo models from inside NetLogo models, essentially treating models like agents – "turtles all the way down."

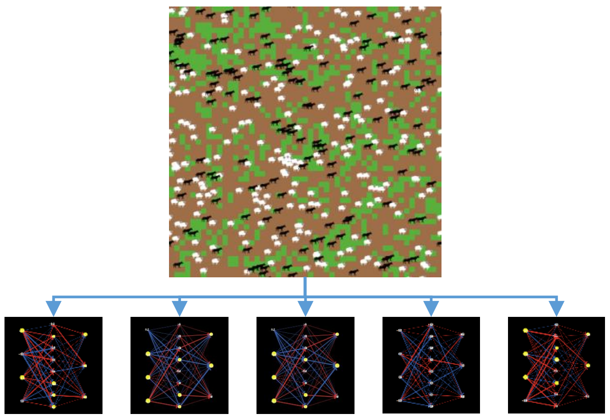
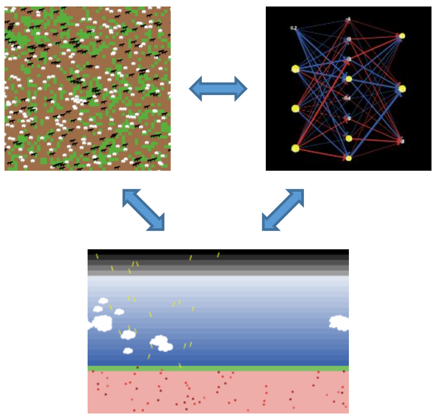



Figure 14: **Multi-Level Linked Systems**. **Each wolf and sheep run their own neural net model which acts as their** '**brain**'. [**Left**] **An ecosystem, a polluting traffic system, and the environment interact** [**Right**]

By using LevelSpace it is possible to encourage and elicit thinking not just *within* complex phenomena, but also *between* them. We think this has interesting implications for modelling as a scientific method, and for modelling as a process of thinking-with external, manipulable representations. We have designed and implemented early prototypes of GUIs and curriculum activities using LevelSpace (Hjorth, Brady, Head, & Wilensky, 2015; Hjorth, Head, Brady, & Wilensky, 2015), and we will discuss our experiences with implementing LevelSpace-based curriculum and discuss best practices for using LevelSpace in classrooms, and for studying student reasoning.

References

Goldstone, R. L., & Wilensky, U. (2008). Promoting Transfer by Grounding Complex Systems Principles. *Journal of the Learning Sciences*, *17*(4), 465–516.

Hjorth, A., Brady, C., Head, B., & Wilensky, U. (2015). Thinking Within and Between Levels: Exploring Reasoning with Multi-Level Linked Models. In *Exploring the material conditions of learning: opportunities and challenges for CSCL*. Gothenburg, Sweden.

Hjorth, A., Head, B., Brady, C., & Wilensky, U. (2015). LevelSpaceGUI: scaffolding novice modelers' inter-model explorations. In *Proceedings of the 14th International Conference on Interaction Design and Children* (pp. 453–457). ACM.

Hjorth, A., Head, B., & Wilensky, U. (2015). *LevelSpace NetLogo Extension. http://ccl.northwestern.edu/levelspace. Center for Connected Learning and Computer-Based Learning. Evanston, IL.* Retrieved from www.github.com/NetLogo/LevelSpace

Thiele, J. C., Kurth, W., & Grimm, V. (2012). Agent-Based Modelling: Tools for Linking NetLogo and R. *Journal of Artificial Societies and Social Simulation*, *15*(3), 8.

Wilensky, U. (1999). NetLogo: Center for connected learning and computer-based modeling. *Northwestern University, Evanston, IL*.

Wilensky, U., & Rand, W. (2015). *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. MIT Press.

This article is released under the CC-BY-ND license.

Music Blocks Workshop

Walter Bender, walter@sugarlabs.org
Sugar Labs, a member project of the Software Freedom Conservancy

Devin Ulibarri, devin@devinulibarri.com
New England Conservatory Preparatory School and YMCA Malden
Cynthia Solomon, cynthia@media.mit.edu
MIT

Claudia Urrea, callaurrea@gmail.com
MIT

Abstract

This workshop is a hands-on introduction to Music Blocks, a microworld of music designed for teachers and learners to explore the fundamental concepts of music in a visual-coding environment (http://walterbender.github.io/musicblocks). Music Blocks is both innovative and beneficial to music education in a number of ways: On the one hand, it is a new method for understanding the fundamental concepts of music; on the other, it is a tool for learning coding and logic skills. It integrates both music and STEM fundamentals in a fun, scalable, and authentic way.

Keywords: music, programming, reflection, Logo

Introduction

Music Blocks (See Figure 1) provides a collection of technological tools specific to a microworld of music, starting with pitch and rhythmic note values, but also providing affordances for repetition, transposition, etc. Music Blocks is more than an interface to a synthesizer and more than a transcription/engraving tool—it is a scalable and modular collection of essential building blocks, which are at the crux of all powerful ideas in music.

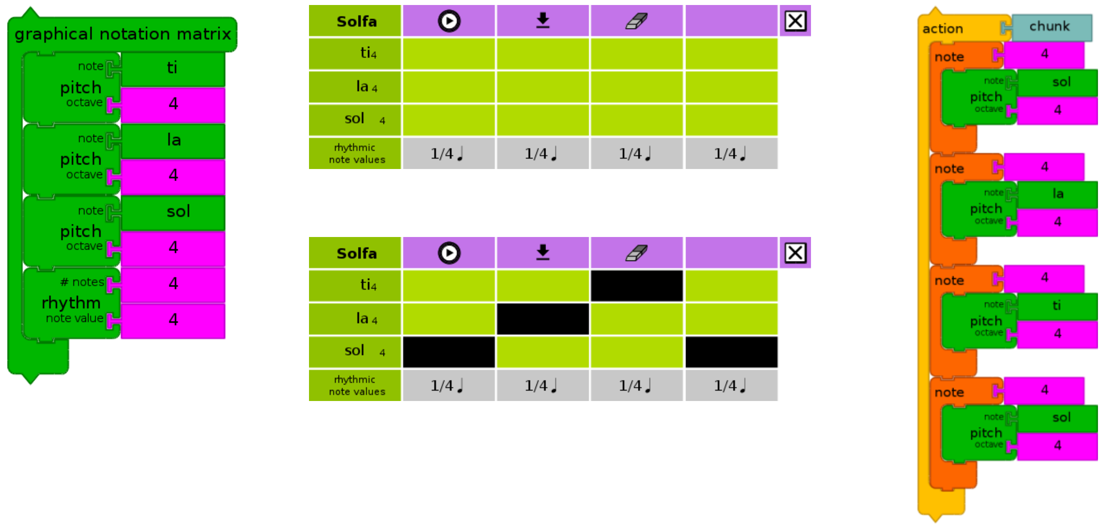


Figure 1: The Music Blocks programming environment running in a Web browser

The workshop will be organized around the concept of a "power piece". A power piece is a melody or a song that is taught because it is powerful and becomes more powerful as it is taught. A power piece is authentic, archetypal, and timeless, yet historically relevant, integrative, and infinitely malleable. *Twinkle-Twinkle Little Star*, a popular English lullaby sung to the tune of the French melody *Ah! vous dirai-je, maman,* published in 1761, is a classic power song. Mozart recast it as a highly embellished theme in his set of *Twelve Variations*. Since its inclusion as a Suzuki-method staple, it has become even more widely known and transformed, further expanding its potency. Its simple and elegant design allows for embellishment, refinement, and pedagogical repurposing (e.g., to teach new rhythms, as a vehicle for learning harmony, etc.). In this case, we will work with a power piece that would be familiar to children in Thailand.

The *power piece* approach differs from just learning more repertoire. Musical pieces are chosen, studied, and manipulated as a means to deepen one's understanding of musical concepts. For example, studying even the first section of the jazz standard *Autumn Leaves* (originally *Les feuilles mortes,* published in 1945 by Joseph Kosma), in all twelve keys in various ways (melody, harmony, guiding tone lines, with improvisation) would deepen a learner's understanding of the archetypal and important minor cadence in jazz.

Music Blocks brings the building blocks of music to the fore. It starts with music's most fundamental parameters: pitch and rhythmic note values. Combining the two parameters creates a note with a definite pitch and duration. Ordering notes in a sequence creates melody, and stacking different notes of the same rhythmic note value at the same point in time creates harmony. Once a melody has been created, "chunks" can be created and combined to create higher-level musical forms. Building in complexity, Music Blocks software provides blocks to manipulate pitch and rhythmic note values through repetition, transposition, note value augmentation and diminution—just to name a few. These tools, because of their inherently fundamental nature, become a powerful metaphor when scripting power pieces.

When exploring a power piece with Music Blocks, a learner might engage in a type of problem solving that is not very different from what a student of music theory might do to analyze a piece of music. Music Blocks places all of the analytic tools in the foreground and makes them easily accessible with minimal music-specific jargon.

Who should come to this workshop?

Children (ages 8+), educators, makers, musicians

Number of participants

15

AV needs

LCD projector, Internet, speakers
Participants should bring a laptop with either Chrome, Firefox, or Safari installed.

Recommendation for participants

Spending some time exploring Music Blocks (and Turtle Blocks) beforehand would be useful, but not required.

http://walterbender.github.io/musicblocks
https://github.com/walterbender/musicblocks/blob/master/guide/README.md

Detailed agenda

Introduction to the Music Blocks Microworld (15 minutes)
Guided exploration of Music Blocks (30 minutes)
Power piece exercise (30 minutes)
Presentations and discussion (15 minutes)

NetLogo Web: Bringing Turtles to the Cloud

David Weintrop, *dweintrop@u.northwestern.edu*
Learning Sciences, Northwestern University

Arthur Hjorth, *arthur.hjorth@u.northwestern.edu*
Learning Sciences, Northwestern University

Corey Brady, *cbrady@northwestern.edu*
Learning Sciences, Northwestern University

Uri Wilensky, *uri@northwestern.edu*
Learning Sciences, Computer Science and Complex Systems, Northwestern University

Abstract

This workshop will be a hands-on introduction to the recently released NetLogo Web platform. NetLogo is a widely used agent-based modelling (ABM) environment that has widespread use in classrooms and research laboratories around the world. The NetLogo Web project brings this popular platform into the cloud, making it fully accessible through any modern browser. This means netbooks, tablets, and even smart phones, can now serve as platforms for designing, implementing, and running powerful agent-based models. The goal of this workshop is two-fold: (1) introduce learners to the next iteration of NetLogo and (2) demonstrate the pedagogical and expressive power of having a fully functional ABM environment in the browser.

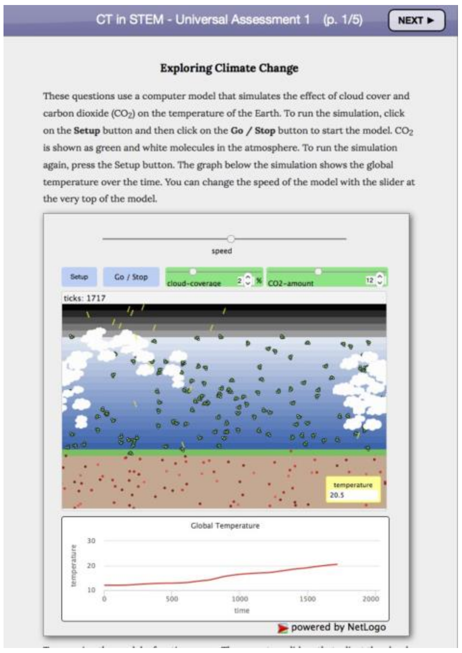
Keywords

NetLogo; Agent-based Modelling; Complex Systems; Online Learning Environments

The currents of educational technology are steadily pushing learning environments off of hard drives and standalone platforms and lifting them into the cloud. By providing browser access to learning tools hosted on the Internet, the technological barrier to entry can drop dramatically, as web-browsers that support that latest technical specification are becoming increasingly ubiquitous and available on diverse and inexpensive devices. Along with wider and easier access, moving learning environments online also allows integration with existing web-based frameworks and learning environments and access to other cloud-based resources, including online data repositories, distributed computing networks, and multimedia libraries with embeddable videos and images. In this workshop, we introduce the recently released NetLogo Web platform (Wilensky, 2015), a JavaScript and HTML5-implementation of the NetLogo agent-based modelling language and environment (Wilensky, 1999), that supports the authoring and running of NetLogo agent-based model inside a browser.

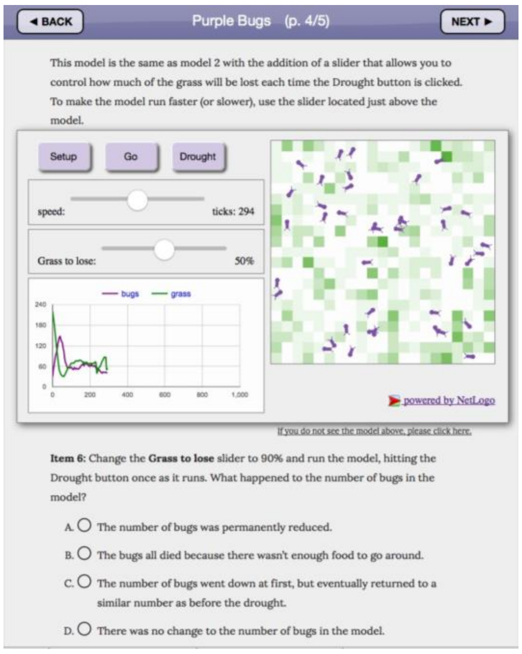
NetLogo is one of the mostly widely used agent-based modeling languages, and provides modelers a “low threshold, high ceiling” environment for modeling complex systems across a wide range of domains (Tisue & Wilensky, 2004; Wilensky & Rand, 2014). It is currently used for educational purposes from as early as middle schools and for “serious research” in laboratories around the world. With NetLogo Web, this powerful learning and research environment moves into the cloud, opening up a new set of possibilities for the environment and its users.

This workshop is for researchers, designer, and educators who currently use NetLogo or are considering using NetLogo in the future. We will focus on how you can benefit from the existing NetLogo models library and library of extensions, while also benefitting from having a purely cloud-based application. During the workshop, we will introduce attendees to the basics of agent-based modeling with NetLogo, including featuring popular models from the NetLogo Models Library and activities that modify and extend existing models. After covering basic NetLogo functionality, we will highlight a number of the powerful new capabilities that are possible due to the environment being situated in the cloud. For instance, because NetLogo Web relies completely on standard web technologies, including HTML, CSS, and JavaScript. This means that NetLogo Web models can be restyled with new CSS rules. More importantly for both teachers and for educational researchers, models in NetLogo Web can be controlled with custom JavaScript calls, and embedded into existing webpages and online educational frameworks. For example, Figure 1 shows a pair of customized NetLogo Web models that have been embedded into a larger an online assessment framework designed to measure learners' computational thinking capabilities (Weintrop et al., 2014). This approach makes the collection of student responses and reflections as easy as clicking button. The workshop will also demonstrate examples of NetLogo Web models that can fetch real-time data from online databases and use that data to control the runtime behavior of the model. Similarly, we will show how NetLogo Web can be used to publish data generated by a model in real-time, so it can be interpreted by other online data analysis tools. Finally, we will show how the new NetLogo Web platform integrates with NetLogo making it possible to take advantage of features only available in the original NetLogo (like access to the large NetLogo Extensions library), as well as how you can convert existing NetLogo models into online NetLogo Web models.



Exploring Climate Change

These questions use a computer model that simulates the effect of cloud cover and carbon dioxide (CO_2) on the temperature of the Earth. To run the simulation, click on the **Setup** button and then click on the **Go / Stop** button to start the model. CO_2 is shown as green and white molecules in the atmosphere. To run the simulation again, press the Setup button. The graph below the simulation shows the global temperature over the time. You can change the speed of the model with the slider at the very top of the model.



This model is the same as model 2 with the addition of a slider that allows you to control how much of the grass will be lost each time the Drought button is clicked. To make the model run faster (or slower), use the slider located just above the model.

If you do not see the model above, please click here.

Item 6: Change the **Grass to lose** slider to 90% and run the model, hitting the Drought button once as it runs. What happened to the number of bugs in the model?

A. ○ The number of bugs was permanently reduced.

B. ○ The bugs all died because there wasn't enough food to go around.

C. ○ The number of bugs went down at first, but eventually returned to a similar number as before the drought.

D. ○ There was no change to the number of bugs in the model.

Figure 1. A pair of NetLogo Web Models embedded within the Computational Thinking in STEM Assessment Framework

References

Tisue, S., & Wilensky, U. (2004). NetLogo: A simple environment for modeling complexity. In *International Conference on Complex Systems* (pp. 16–21).

Weintrop, D., Beheshti, E., Horn, M. S., Orton, K., Trouille, L., Jona, K., & Wilensky, U. (2014). Interactive Assessment Tools for Computational Thinking in High School STEM Classrooms. In D. Reidsma, I. Choi, & R. Bargar (Eds.), *Proc. of INTETAIN 2014, Chicago, IL, USA* (pp. 22–25).

Wilensky, U. (1999). *NetLogo*. Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University. http://ccl.northwestern.edu/netlogo.

Wilensky, U. (2015). *NetLogo Web*. Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University. http://www.netlogoweb.org.

Wilensky, U., & Rand, W. (2014). *Introduction to Agent-based Modeling*. Cambridge, MA: MIT Press.

New Frontiers in Educational Robotics with the Raspberry Pi and the GoGo Board

Arnan Sipitakiat, *arnans@eng.cmu.ac.th*
Learning Inventions Lab, Faculty of Engineering, Chiang Mai University, Chiang Mai, Thailand

Paulo Blikstein, *paulob@stanford.edu*
Transformative Learning Technologies Lab, School of Education, Stanford University, USA

Abstract

This workshop will offer a hands-on demonstration of how small embedded-computers such as the Raspberry Pi can offer new possibilities in educational robotics. We aim to demonstrate (A) how off-the-shelf peripherals can be utilized, allowing learners to use devices like webcams, GPS, SMS Air cards, and Wifi in ways that would have been too technical before. (B) We will present the GoGo Board, which snaps on top of the Raspberry Pi and allows the Raspberry Pi to control motors and read sensor values. (C) Programming will be done using Tinker, a new graphical environment that integrates the various components into a single straightforward platform.

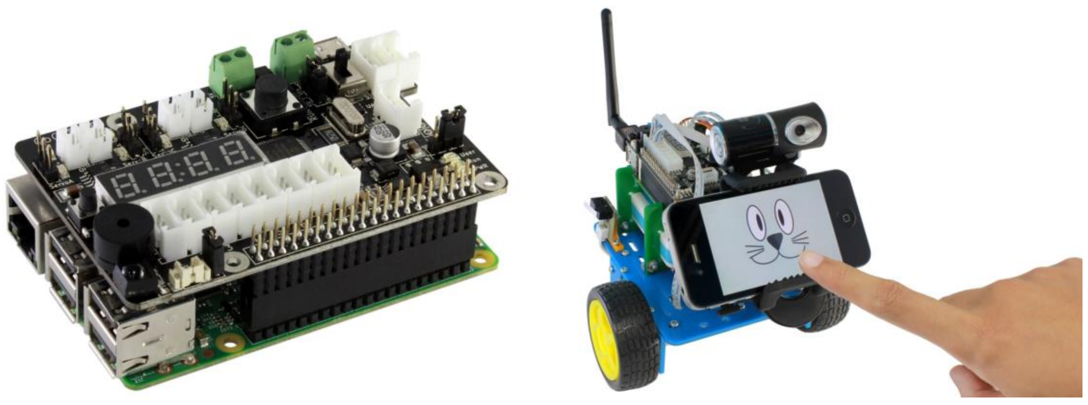


Figure 1. Left: The GoGo Board on top of a Raspberry Pi. Right: a robotic pet utilizing smartphones and off-the-shelf USB peripherals

Keywords

Robotics, Embedded Computer, Programming

Scope

This workshop is aimed at educators and researchers who are interested in using physical computing with learners. It highlights a set of tools that elevates the project possibilities for novice programmers. Participants will be engaged in creating a mini project that involves both constructing a physical object and programming a desired behaviour. Experiences in using robotics tools such as Lego Mindstorms or the Arduino would be helpful but not required.

The workshop will start with a demo of the platform. The participants will then divide into small groups. We will then offer a walk-through of the environment along with a number of quick exercises. The majority of the workshop time will be spent on a mini project where participants will be able to put together laser-cut parts and decorate. Participants will also program behaviours for their creation making use of the tools available. Participants should be able to perceive how this activity can be applicable to their students or their research work.

Workshop Tools

Raspberry Pi

The Raspberry Pi is a credit-card sized embedded computer that runs the Linux operating system. What makes the Raspberry Pi special for physical computer activities is that it can be run off of a set of batteries and that it has general purpose extension ports. It also runs the Linux operating system and support the Python programming language. The miniature hardware together with a powerful programming language open up tremendous opportunities for creating innovative learning tools.

GoGo Board

The GoGo Board is a robotics add-on board that snaps onto the Raspberry Pi. It allows one to use the Raspberry Pi to easily control motors and use analog sensors, which are fundamental requirements for a physical computing system.

Tinker Programming Environment

Tinker is a graphical programming system that offers abstractions suitable for novice programmers to program a desired behaviour utilizing the rich resources available to them via the Raspberry Pi and the GoGo Board without needing to learn the nitty-gritty details of the underlying operating system.

Participants and Requirements

- This workshop can accommodate approximately 15 participants (five groups of three persons each).
- Each group is expected to bring at least one laptop computer (PC is preferred but Mac is also ok).
- Each group requires one sizable table and a power outlet with at least two plugs.
- A projector and a screen is needed for demonstrations.

Videogame construction with models of physical parameters

Angel Pretelín-Ricárdez, *apretelin@ipn.mx*
Instituto Politécnico Nacional, UPIITA, México
Centro de Investigación y de Estudios Avanzados (Cinvestav-IPN), México

Ana Isabel Sacristán, *asacrist@cinvestav.mx*
Centro de Investigación y de Estudios Avanzados (Cinvestav-IPN), México

Abstract

In this workshop, participants will construct videogames that require the use of models of basic physical parameters (e.g. friction, force, velocity, gravity, density, linear damping, angular damping) in the game mechanics within the story. It illustrates an approach where students can gain a deeper understanding of those physical concepts, their functioning and possible applications in realistic systems or situations (RSS) from physics, computer science, etc., in a fun constructionist (Papert & Harel, 1991) way (the videogame construction).

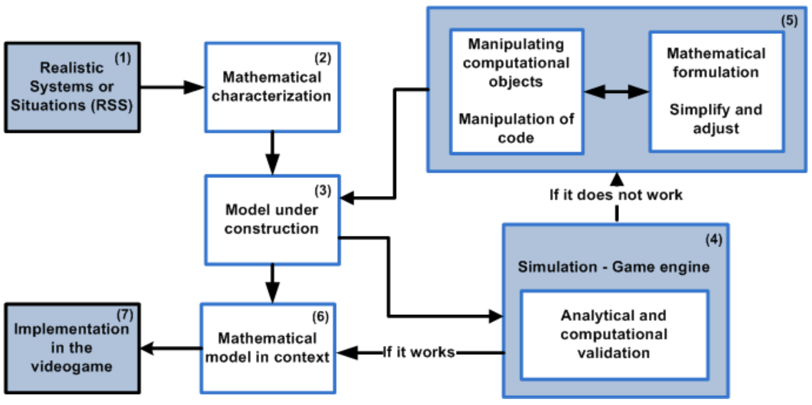

(1)
Realistic Systems or Situations (RSS)
(2)
Mathematical characterization
(3)
Model under construction
(4)
Simulation - Game engine
Analytical and computational validation
(5)
Manipulating computational objects
Manipulation of code
Mathematical formulation
Simplify and adjust
If it does not work
If it works
(6)
Mathematical model in context
(7)
Implementation in the videogame

Figure 1. Construction cycle and model implementation in the videogame

The workshop consists of a learning sequence with three activities (see further below) where the following construction cycle is carried out (see Figure 1): (1) A RSS is proposed related to engineering; participants will have to model this RSS for implementing it into the game mechanics of a videogame. (2) Participants will need to identify the physical parameters in those RSS and describe them through algebraic formulas or equations (help cards will be provided for this): this is a first mathematical characterization. (3) That first mathematisation of the RSS will need to be converted (abstracted) into a working model in the game engine computer code, in order to: (4) test it and validate it, in a first simulation. Here are two possibilities: If the implementation does not work, then (5) the model needs to be simplified or adjusted –i.e. abstracted again through the manipulation of computational objects and programming code, repeating steps (3) and (4). If the simulation runs properly, then we go to we have a model in context (6) that is ready for the final implementation in the videogame (7).

Keywords (style: Keywords)

Constructionism, videogames, physical parameters, mathematical modelling

The videogame-construction activities

This workshop is inspired by a research project (see Pretelín-Ricárdez & Sacristán, 2015), where engineering students construct (design and program) videogames that require applying mathematical models from phenomena (e.g. from realistic systems or situations –RSS— from physics, computer science, etc.) that are usually part of the practice of an engineer. Modelling is an important activity in the life of an engineer, because it is the first step in developing and designing any device, system, product or process. It requires applying mathematical skills in order to build appropriate representations and models. Thus it is important for engineering students to understand the mathematical concepts that underlie a certain phenomenon, as well as the particular context where they will be applied, and to adapt them in a model.

The aim of the workshop is to give participants the experience of this approach where students can gain a deeper understanding of how to apply and contextualize mathematical parameters and models in engineering-like situations, through videogame construction. Moreover, if the basic physical concepts are well understood, defined, and applied, they will be the basis for the modelling and implementation (e.g. in a videogame) of much more complex phenomena.

The workshop is structured through an introduction followed by a sequence of three activities in order to encourage participants to apply mathematical-modelling concepts in a real engineering situation or system. These activities are:

- Introduction: Description of the workshop and the purpose of the videogame construction. Familiarisation with the game engine. Distribution of the workshop material, pre-designed images and sounds, quick help cards and half-baked programs. (20 minutes)
- Activity 1: *Modelling – Mathematisation* (20 minutes). Each participant will be given a RSS and they will need to find a correct mathematical model (either propose their own or choose among game cards provided) that describes the parameters involved in that situation. This will require some previous basic knowledge of mathematics and physics in order to identify the main mathematical variables and relationships.
- Activity 2: *Modelling - Programming simulations* (20 minutes). Each participant will attempt to adapt the mathematical model by programming it into the videogame engine and thus creating a simulation of the RSS.
- Activity 3 (Collaborative): *Construction of the videogame* (30 minutes). In this activity, teams of two participants will develop collaboratively an idea for a videogame that requires solving puzzles and completing levels. They will create a brief "storyboard" of up to 4 images, adding a brief description of the gameplay and mechanics. The game mechanics will require, at the core, using the models built in the previous activity.

Requirements

- Target audience: People with basic programming skills (some experience with any structured programming language or object-based programming) and some basic (high-school level) knowledge of mathematics and physics.
- Maximum number of attendees: 14
- Equipment required: Windows 7 operating system or higher.
- Software required: The free version of Game Maker Studio which can be downloaded and pre-installed from http://www.yoyogames.com/studio/download .

References

Papert, S., & Harel, I. (1991). Situating constructionisn. In: S. Papert & I. Harel (Eds.), *Constructionism*. (p. 1-12). New Jersey, NJ: Ablex Publishing Corporation.

Pretelín-Ricárdez, A. & Sacristán, A. I. (2015). Videogame Construction by Engineering Students for Understanding Modelling Processes: The Case of Simulating Water Behaviour. *Informatics in education*, 14(2), 265-277. doi:10.15388/infedu.2015.15

World Peace Song Project

Yoshiro Miyata, *miyata@sist.chukyo-u.ac.jp*
School of Engineering, Chukyo University, Toyota, Japan

Mihoko Kamei, *kamei@sugiyama-u.ac.jp*
School of Culture-Information Studies, Sugiyama Jogakuen University, Nagoya, Japan

Abstract

We use products and energy from all around the world. We have constructed a global community connected physically through the products and energy we use. However, we have not connected our hearts globally and created many global problems concerning food, energy, environment, often resulting in conflicts (Miyata et al. 2015). Collaborative music making has been a powerful means to connect our hearts. We would like to connect our hearts globally by making music together globally.

We propose a workshop in which the participants will be able to experience the process of creating music in collaboration with many partners around the world, all singing lyrics they have created expressing peace in their own languages. We welcome participants with many different nationalities, native tongues, ages and genders, to express their feelings and thoughts about world peace in their native languages, and sing together in a global chorus with World Museum partners around the world who will be listening to us online and send messages to us in real time. The participants will experience playing instruments constructed with Scratch on Raspberry Pis.

This workshop demonstrates a new way of music making based on constructionism. In contrast to the commercial music making where a very few professional musicians produce music for millions of passive listeners, we create a community in which anyone can make music collaboratively with the help of professional musicians.

Figure 1. World Peace Song was premiered in the Workshop & Concert in Nagoya, March 2015

Keywords; Global Collaboration, World Peace, Music, Creative Expression

World Peace Song Workshop

World Peace Song Project is a global collaboration project to create a song of peace in which people around the world can contribute lyrics about peace in their own languages, and sing their lyrics with all other participants together in a global chorus. The music for the song was composed by Women of the World (a) for the partner children of the World Museum Project (b) to collaboratively make music of peace. The song was premiered in "Women of the World in Nagoya - Workshop and Concert" in March 2014. It was performed together with recordings of the lyrics contributed and sung by children from around the world, the workshop participants and concert audience, singing and playing musical instruments created with Scratch on Raspberry-Pis. The performance was streamed online for the partners around the world to view and send comments which were then displayed on the stage in real time. The event was quite successful and some of the participants have tried or will try peace song workshops in their own schools.

The workshop in Constructionism 2016 will proceed with the following plan.

1. A brief introduction to the project, explaining its purpose and short video recordings of the peace song performances in Nagoya, Amsterdam, and Boston. (5 min.)
2. We will perform the peace song as a collaboration between live performance by some musicians on the stage and recorded performance by partner children from around the world. (5 min)
3. We will make groups of 3-5 participants. Each group will choose one melody from four simple melodies, and create lyrics of peace in their own languages to go with the melody. Each group will practice singing their song of peace. (15 min)
4. All the groups will practice singing the melodies in a chorus. Then the recordings by the children around the world will join the chorus. (10 min)
5. Finally, the musicians, the participants, the partners will all sing the peace song together. The performance will be streamed online for the partners around the world to view and send messages to us, which will be displayed on the stage in real time. The singing by the participants will be recorded and will be included in the future World Peace Song performances and workshops. (10 min)
6. Reflections and discussion. (10 min)

We are now developing an integrated system for global collaboration and performance so that many people can do the workshop anywhere in the world. The current system is a collection of separate tools for recording of individual songs, editing and mixing them, playing the edited chorus, live streaming, viewing the streaming, sending messages, receiving and displaying the messages, etc.. By integrating these tools as a single web application, anyone with internet access will be able to try the workshop with their local participants without special knowledge nor software. As a result, many children with different languages and backgrounds can contribute lyrics and sing in their own languages together with partners around the world.

(a) Women of the World is a renowned a cappella group based in Boston who sing music from around the world as well as original songs to promote world peace. They won the 2014 US a cappella championship in The Harmony Sweepstakes A Cappella Festival.

(b) World Museum Project is a network connecting many educators around the world to help children living in different places around the world to collaboratively create many artworks by drawing, making animations and music. Since it started in 2010, more than 1000 children from more than 35 countries in five continents collaborated on various themes.

References

Miyata, Y., Kamei, M. and Ueshiba, T. (2015) *Connected Design - Toward New Design Principles from an Evolutionary and Global Perspective*. A keynote lecture in the 4th International Conference on Life Science & Biological Engineering, Nagoya, Japan

Constructionism in Action 2016

About Suksapattana Foundation

With a start-up grant of Baht 5.3 million in 1996, F.R.E.E. (Foundation for Research Education and Enterprise), to which His Majesty the King later graciously bestowed the name Suksapattana Foundation, started its work to improve education in Thailand.
Suksapattana entered a collaboration agreement with MIT under which Professor Seymour Papert and his team from the MIT Media Lab introduced constructionism to education to Thailand.
The effects of the ensuing and path breaking 'Lighthouse Project' are still felt today. The foundation has worked with three main sectors: Schools, Villages, and Industries.
(*Text modified from The Education & Public Welfare Foundation website http://www.epwfoundation.com*)

About “Constructionism”

Constructionism is a learning philosophy created by Seymour Papert at MIT. The idea was built upon Jean Piaget's Constructivism, where learning was seen not as a transmission of knowledge from an expert to the learner. Instead, learning is a process where a learner goes through a continuous process of making his or her own meaning about the world. At MIT, Papert has shown how constructing personally meaningful artifacts using a computer can allow Piaget learning process to take place “especially felicitously”. Papert and his learning ideas became widely known in the eighties especially when “Logo”, a computer programming language designed for children, became a popular learning tool on computers in that era.

มูลนิธิ
ศึกษาพัฒน์
Suksapattana
Foundation

ISBN 978-616-92726-0-1