SLurtle Soup: a conceptual mash up of constructionist ideas and virtual worlds

Carina Girvan, *girvanc@tcd.ie*
Centre for Research in IT in Education, School of Education and School of Computer Science & Statistics, Trinity College, Dublin, Ireland

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

Timothy Savage, *Tim.Savage@cs.tcd.ie*
Centre for Research in IT in Education, School of Education and School of Computer Science & Statistics, Trinity College, Dublin, Ireland

Abstract

In *Mindstorms*, Papert (1980) advocated "*the construction of educationally powerful computational environments that will provide alternatives to traditional classrooms and traditional instruction.*" (p. 182). At the same time he identified that the technology of the day was limited in its capabilities and functionality. Since then considerable work has been done to create tools ranging from Logo, Mindstorms, Scratch, ToonTalk, etc which embody a constructionist approach to learning. In this paper we introduce a new embodiment of constructionist theory, SLurtles (programmable Turtles in *Second Life*).

The virtual world *Second Life* provides a high floor high ceiling building and programming interface difficult for the novice to master. To address this problem, SLurtles leverage concepts from Turtle and Turtle geometry; *Mindstorms*; and *Lego*, and combines them with *Scratch for Second Life* (a low floor, high ceiling programming environment for *Second Life),* to create a low floor, high ceiling programmable building tool for *Second Life*.

This paper presents a use case example and description of ideas which underpin the SLurtle conceptual mash up. This is followed by an outline of our initial trial with learners using SLurtles to experience constructionist learning in a virtual world, with an overview of our research aims.

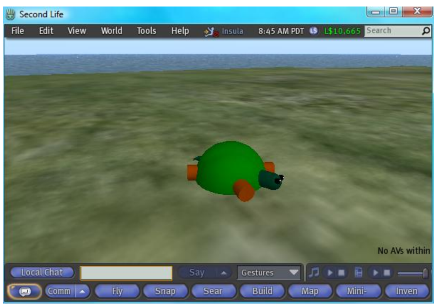

Figure 1. SLurtle, a programmable Turtle in Second Life

Keywords

Turtle geometry; Mindstorms; Scratch for Second Life; Constructionism; Virtual Worlds; Second Life

Introduction

In *Mindstorms*, Papert (1980) advocated "*the construction of educationally powerful computational environments that will provide alternatives to traditional classrooms and traditional instruction*." (p. 182). While he accepted that the tool he was using, Logo, was limited by the capabilities and functionality of the technology of the 1970s, it nevertheless went on to give rise to a rich vein of research based on the notion of 'objects-to-think-with', with *Scratch* (Maloney et al., 2004) and *ToonTalk* (Kahn, 2001) prominent examples. These 'objects-to-think-with' provide the learner with an easy to access, 'low floor', entry to programming, whilst allowing the more experienced user create highly complex algorithms, resulting in a 'high ceiling' (Sheehan, 2000).

Virtual worlds, such as *Second Life*, provide three-dimensional, persistent and flexible environments that can provide an alternative to traditional educational locations and approaches. While the automatic reaction to replicate what has gone before in a new technology (Winn, 2005) has been observed in virtual worlds with the creation of in-world lecture theatres, etc, some educators have begun to explore the potential of this technology and the new educational opportunities it can support.

Much like the Lego/Logo 'behaving machines' that could be created in the real world (Resnick, 1993), artefacts which are interactive and exhibit behaviours can be created in the virtual world. Content creation in the virtual world of *Second Life* begins with a simple object (prim), which can be manipulated in numerous ways and combined with others to create sophisticated structures. However the skills required to create even a simple artefact, such as a staircase, results in a high step to entry. Furthermore, in order for this simple artefact to exhibit behaviours requires the object to be programmed. Programming within *Second Life* requires the use of Linden Scripting Language (LSL), a high floor, high ceiling programming language with C style syntax.

Based on *Scratch* (Maloney et al., 2004; Resnick et al., 2009), Eric Rosenbaum (2008) developed *Scratch for Second Life* (S4SL) as a low floor programming environment for *Second Life*. S4SL provide the learner with an opportunity to programme and introduce behaviours to otherwise static objects in *Second Life*, much like the Lego/Logo 'behaving machines'.

This paper presents SLurtles (programmable Turtles in *Second Life*) as the embodiment of constructionist theory within a virtual world. SLurtles leverage concepts from Turtle and Turtle geometry, *Mindstorms* and *Lego*, combining them with S4SL within the virtual world of *Second Life*, to provide new 'objects-to-think-with' within the new learning environment of virtual worlds. S4SL provides the learner with a low floor, high ceiling programming environment to programme the SLurtles. The SLurtles in conjunction with S4SL provide the learner with a constructionist low floor, high ceiling tool for the construction of objects in *Second Life*.

Following a brief discussion on virtual worlds and the Logo heritage, this paper presents a use case example and description of SLurtles. This is followed by an outline of our initial trial with learners using SLurtles to experience constructionist learning in a virtual world, with an overview of our research aims.

Background

Virtual Worlds

Virtual worlds are typically characterised as persistent, three-dimensional, immersive environments (Castronova, 2005) which provide opportunities for users to collaborate and share experiences without the need for physical co-presence. Users are represented by avatars which can interact with their environment and communicate with others through a range of communication tools. This supports the user's perception of immersion and co-presence with other users. Some virtual worlds also provide the opportunity for users to create content. The construction of static objects which can also be programmed is available to users of *Second Life*, limited for users of *There* but unavailable to users of *Club Penguin*.

Replication

As Winn (2005) describes in his work on virtual reality, the natural reaction of the early adopters of a technology is to replicate what has gone before in the new environment. Thus in virtual worlds we see replicated lecture theatres and university campuses. While this may provide some advantages for distance learners, there is a need to move beyond what can be replicated and begin to explore and innovate in our use of virtual worlds for educational purposes.

Increasing numbers of learning experiences reported in the literature are making links between the features of virtual worlds and the learning approaches adopted. For example, role play can leverage the sense of immersion arising from the use of avatars, communication tools and the 3D environment (Jamaludin et al. 2009). There is also opportunity for experimentation without real-world repercussions (Dede, 1995; Burmester et at., 2008) as well as the creation of learning experiences which could not easily be achieved in the real world (Good et al., 2008).

Despite the increasing number of reported learning activities which are beginning to explore beyond simple replication, Savin-Baden (2008) notes that these learning experiences often lack pedagogical underpinnings. We posit that to design learning experiences that have strong pedagogical underpinnings we first need to identify those pedagogies which can strongly leverage the unique combination of affordances that virtual worlds can provide (Girvan & Savage, 2010). These pedagogies then need to be explored in action, as part of a carefully designed learning activity experienced by several groups of learners, to identify how the pedagogy manifests and leverages the affordances of a virtual world.

Affordances

Within the literature on virtual worlds, ‘affordance’ is a widely used but often undefined term. The affordances of an object are the actions that an individual can conceive of as possible when interacting with that object based upon a visual impression (Gibson, 1979). For example, based on the perception of a sharp edge an individual can envisage that a knife affords cutting. We specifically focus on Norman’s (1999) description of ‘perceived affordances’ which requires us to acknowledge that what may be perceived as an affordance may differ from person to person. Those affordances which may support educational activities are described by Kirschner (2002) as ‘educational affordances’. Thus, those affordances, as perceived by users, which could be leveraged for educational purposes can be described as ‘perceived educational affordances’ (Girvan & Savage, 2010).

Our current exploration of pedagogies within virtual worlds focuses on *Second Life*. Based on the literature and our experience as educators within virtual worlds, we can identify that the three-dimensional landscape and representation of avatars within *Second Life*, combined with the opportunity for interaction through communication tools, affords a sense of self and presence. This sense of self and presence can in-turn result in immersion and support socialisation and collaborative learning (Kemp & Livingstone, 2006; Cross, O’Driscoll & Trondsen, 2007; Minocha & Roberts, 2008). *Second Life* also provides tools that afford build and rebuild opportunities for the construction and programming of objects and environments (Delwiche, 2006) which are persistent (Castronova, 2005), interactive and flexible. However, these tools, while providing a high ceiling, present the learner with a high floor to access.

Second Life and constructionism

With regard to the perceived educational affordances of *Second Life* outlined above, constructionism appears to be a potentially appropriate pedagogy to underpin the design of learning experiences in virtual worlds. Central to constructionism is the designing and construction of shareable artefacts as an opportunity for the learner to actively explore, experiment and extend their understanding. Hoyles et al. (2002) also note that programmability remains an essential facet of constructionism. These core features can leverage the building and scripting tools in *Second Life* which afford build and rebuild opportunities in the construction and programming of objects and the environment. As virtual worlds are persistent, any object created by a learner can remain in the space it was created, without requiring the user's presence. This provides an opportunity for learners to create artefacts which can be returned to and shared.

The sense of self and presence, resulting in an opportunity for socialisation and collaborative learning, can also be leveraged to support the sharing of artefacts. We suggest that immersion may support the creation of personally meaningful artefacts and result in a sense of interacting and observing artefacts more directly. However, there is little understanding of how the use of an avatar or the variety of communication tools and sense of presence and immersion could impact the pedagogy in action.

Based on our current understanding of the literature on virtual worlds, there have been few studies which have placed constructionism at the heart of the learning experience. Both Dreher et al. (2009) and Good et al. (2008) report on learning experiences within *Second Life* that use constructionism but lack a description of how constructionist learning was supported during the learning activities. There is limited discussion as to how the specific features of the technology support the constructionist learning activities and a lack of results or findings that enhance our understanding of the pedagogy in action within virtual worlds.

Within the literature on tools which facilitate constructionist learning, Kahn (2007) considers the potential of virtual worlds such as *Second Life*. His particular focus is on constructing programmes within these environments similar to *ToonTalk*. The potential additional benefits he identifies are based on the opportunity to leverage the impression of a three-dimensional environment, realistic physics, collaboration and 'inhabited spaces'.

From Turtle to Scratch

Logo was designed to be an easily accessible (low floor) and at the same time, powerfully expressive (high ceiling) programming language providing opportunities for learners to construct, explore and investigate (Feurzeig, 2007). Before Turtle, Logo was used to write poetry, create translators or construct strategy games (Papert, 1980). The Turtle provided an opportunity to engage learners across a much wider age spectrum with an 'object-to-think-with', whether sharing a physical presence with the learner or as an image on the screen which produces colourful lines. 'Objects-to-think-with' provide a focal point for the "intersection of cultural presence, embedded knowledge, and the possibility for personal identification" (Papert, 1980, p.11), whilst being shaped and even split due to differing cultural and political contexts (Agalianos et al., 2006)

Turtle geometry was proposed as a computational style of geometry, unlike Euclid's logical style (Papert, 1972). Within Euclidian geometry there are a number of concepts, one of which is a 'point' which has no properties other than 'position'. Instead of a point, Turtle geometry uses a 'Turtle', whether physical or on a screen, to draw lines. Similar to a point, a Turtle has a position but it also has a 'heading' resulting from the direction it is facing (Papert, 1972; 1980). The Turtle 'object-to-think-with' thus provides the entry point to Turtle geometry which is dependent upon both position and heading.

Since the early days of Logo and Turtle geometry, the concept of low floor, high ceiling programming languages has continued to result in the development of languages and micro-worlds such as Scratch (Maloney et al., 2004). Scratch provides a good example of how these later instantiations of constructionist 'objects-to-think-with' began to widen the walls, supporting a variety of projects that could be realised dependant on the interests and learning styles of the user (Resnick et al., 2009).

The design of a constructionist tool for a virtual world: SLurtles

SLurtles were designed and created to provide a constructionist 'object-to-think-with' for use within *Second Life,* which leverage the perceived educational affordances of virtual worlds. SLurtles are designed as a low floor, high ceiling tool for the creation of objects, used in conjunction with *Scratch for Second Life* (S4SL).

This section presents a SLurtle use case example to provide context to the following detailed description of the appropriation of Eric Rosenbaum's (2008) S4SL and modification of 'lineSegment' in the design and creation of SLurtles.

SLurtles in action: A use case

SLurtles can be programmed to create and place virtual blocks in patterns and sequences to construct sophisticated artefacts within the virtual world of *Second Life*. This programming is undertaken using the low step interface of *Scratch for Second Life* and imported into the SLurtle. A range of SLurtles have been constructed each of which contains and lays a unique style of block

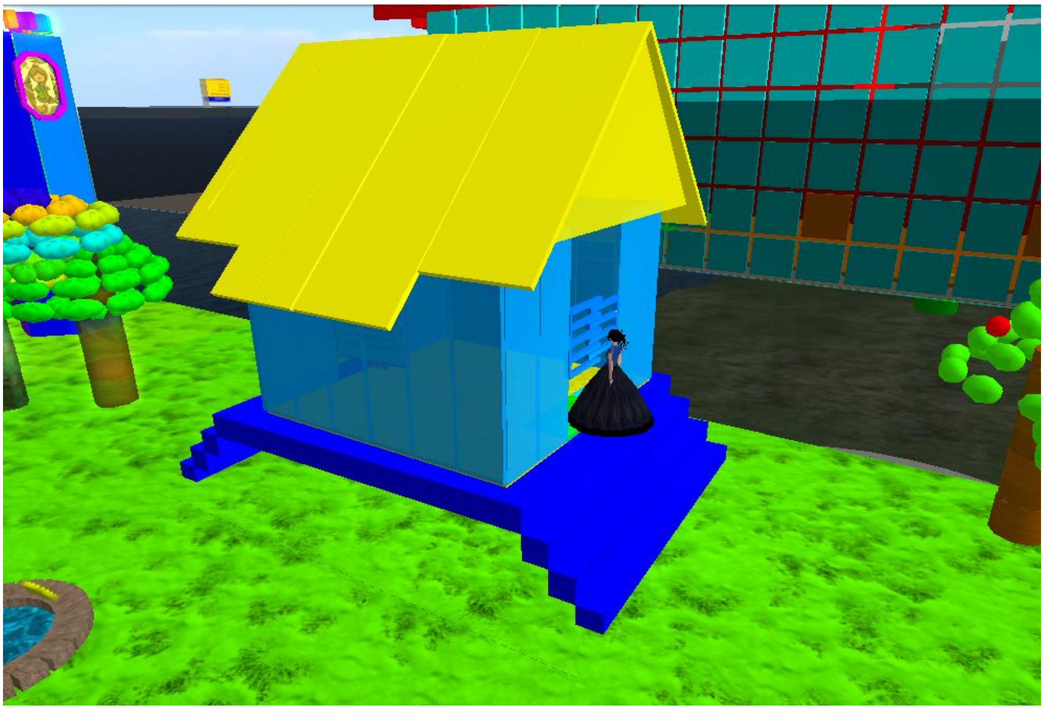

Figure 2. Witch's house in the Enchanted Forrest installation

The witch's house and trees shown in figure 2 were each created using programmable SLurtles. The house was constructed using six SLurtles, each which created different shaped blocks. Each SLurtles was programmed using S4SL to position, colour and place each block, with the length of each block determined by the distance the SLurtle travelled. In a similar way each of the trees shown was created using two SLurtles. The first SLurtle created cylinders and was programmed using S4SL to create the trunk of the tree. The second SLurtle created spheroids and was programmed using S4SL to create the green leaves and randomly place red spheres. Both the witch's house and trees, once created were then programmed to exhibit behaviours.

For example the red spheres, which represented apples on the trees, were programmed using S4SL to shrink and change colour when touched, accompanied by the sound of a witch's cackle.

Scratch for Second Life (S4SL)

As previously stated the tools which support the creation and programming of objects in *Second Life*, while powerful, can be difficult to use for those without previous experience. Object creation is essentially an exercise in 3D modelling and the Linden Scripting Language (LSL), with its C-style syntax and structure, has a high barrier to entry, particularly for someone with no previous programming experience.

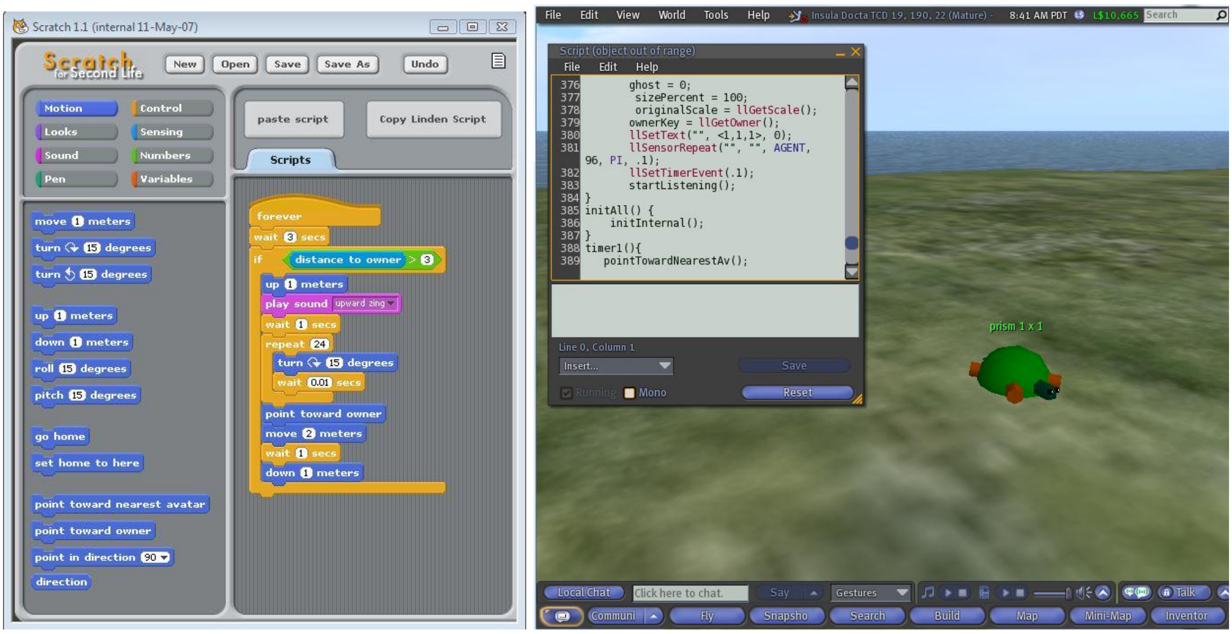

Figure 3. Scratch for Second Life interface and Second Life client with open LSL script and SLurtle

Based on the graphical programming language, *Scratch* (Maloney et al., 2004; Resnick et al., 2009), *Scratch for Second Life* (S4SL) was designed by Eric Rosenbaum (2008) as a low floor programming environment to help lower the barrier to adding behaviours and interactivity to otherwise static objects in *Second Life*. Unlike Kahn's (2007) focus on constructing programming tools within the virtual word, S4SL is a separate programme which runs outside of the virtual world. It provides the traditional Scratch visual programming environment in which graphical blocks are snapped together to create a programme which is then exported into equivalent LSL code. All that is required of the user is that they select the "Copy Linden Script" button to generate the LSL code and then paste that into a new script within an object created in *Second Life.* Figure 3 shows the S4SL environment and a section of the LSL code generated which is placed inside a SLurtle.

Pen down

The S4SL blocks which control behaviour originate from *Scratch,* are fairly intuitive and include categories for 'motion, 'control', 'looks', 'sensing', 'sound', 'pen', 'variables' and 'numbers'. The following discussion will focus on the ones that control the "pen".

In the traditional *Scratch* programming environment the 'pen down' command results in a 2D line drawn on the stage, behind the sprite as it moves. This feature in turn is influenced by Turtle geometry (Papert, 1972) and the ability to create programmes which direct the actions of a physical or on-screen Turtle to create line drawings. In S4SL the 'pen down' legacy continues however LSL and *Second Life* have no ‘pen’ equivalent. Instead, Rosenbaum has created a work-around for S4SL which leverages the affordances of the building tools in *Second Life*. A ‘lineSegment’ object, a 0.1 meter wide by 0.1 meter high cuboid, can be placed within a *Second Life* object. This containing object can be scripted using S4SL and when ‘pen down’ is called this causes an instance of the 3D ‘lineSegment’ to appear (be drawn) within the virtual world. The location and length of the new object are determined by where the parent object was when the ‘pen down’ command was issued and how far it travelled in one action. For example, figure 4 shows some S4SL code and what happens in *Second Life* after an avatar has clicked on the object (the wooden cube) which contains both the LSL script generated by S4SL and a ‘lineSegment’.

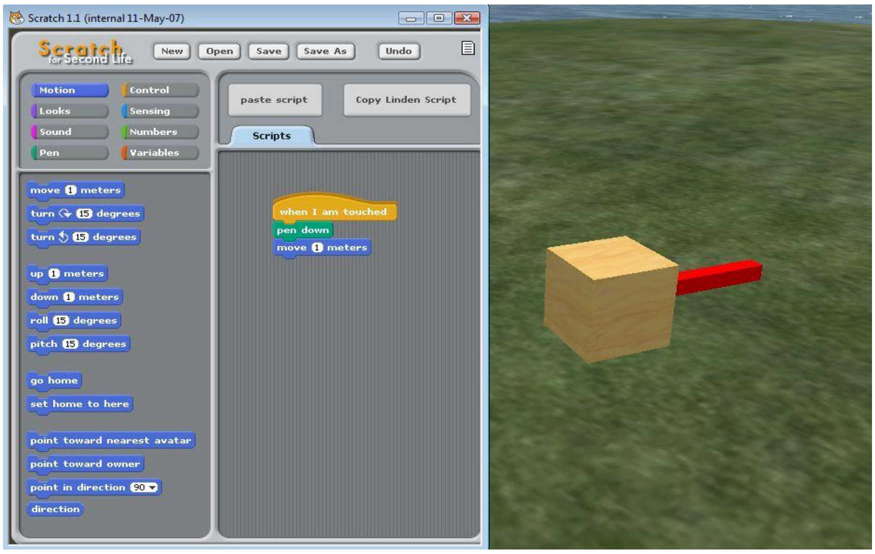

Figure 4. S4SL blocks and result of the programme in Second Life, object producing a ‘lineSegment’

Each time the cube is clicked it will create a new instance of ‘lineSegment’. Unlike the *Scratch* or Turtle pen marks which merely draw lines, each shape created using the pen in S4SL is an object, although only temporal.

SLurtles

To the learner, the cube in figure 4 when first created is much like the Euclid point. It can be observed to have a position but no obvious heading. When programmed to move forward it will move forward but to the learner it is not clear what direction this will be until the programme is executed. In addition, S4SL users must explicitly embed a ‘lineSegment’ in an object so that those objects can respond to ‘pen up/down’ commands. We have taken S4SL and the notion of ‘lineSegments’ in objects one step further by making more explicit the link to Turtle and by drawing on ideas from Lego building blocks and *Lego Mindstorms*.

We have created SLurtles (programmable Turtles in Second Life, shown in figure 1) which behave in a similar way to Turtle. A SLurtle has an explicit notion of heading and responds to ‘pen up/down’ commands by placing a SLurtle block. The learner does not need to know about the embedded object but by programming SLurtles using S4SL, sophisticated objects can be created in the virtual world.

The SLurtle block is an adapted ‘lineSegment’. As stated above, the original ‘lineSegment’ was temporal lasting a short time before disappearing. However, by leveraging the persistence affordance of *Second Life* the SLurtle block is a permanent object, thus SLurtles provide an opportunity to lower the barrier to object creation in *Second Life*.

Borrowing from the idea of Lego bricks a variety of different SLurtles are provided which create different shaped blocks including cuboids and spheroids of different height and widths.

Current work

By combining S4SL with concepts from Turtle geometry and Lego bricks, SLurtles become an ‘object-to-think-with’ (Papert, 1980) within a virtual world. They also provide an opportunity to explore constructionism in action within a virtual world.

An initial user trial of SLurtles has recently been conducted with postgraduate students on a technology and education course. Students on the course have a wide range of backgrounds encompassing computer scientists with an interest in education and school teachers with an interest in using computers more effectively in their own classroom. As part of the course learners are introduced to a range of pedagogical theories and technologies.

This year, as part of their work on virtual worlds, learners were introduced to SLurtles following a brief introduction to *Second Life* as well as lectures and workshops on constructionism, using *Mindstorms* and *Scratch*. Following a workshop session with SLurtles which had an emphasis on using Kolb’s (1984) learning cycle to support the explorative process, students collaborating in pairs, used SLurtles to create an installation which in-turn was programmed using S4SL to be interactive as part of their course assessment.

The user trial has provided us with an opportunity to garner feedback on the learners’ experiences with SLurtles and their thoughts on its design. It has also provided us with an opportunity to begin exploring our research questions about constructionism in action within a virtual world. During the learning experience participants recorded their text based conversations using the chat logging feature in *Second Life*. Following completion of the assignment participants completed a questionnaire and were invited to take part in a semi-structured interview. Finally their interactive installations and reflections submitted for the assignment were collected.

Qualitative data analysis is currently underway and a number of positive and unexpected outcomes are beginning to emerge. Overall participants appear to have enjoyed using SLurtles to construct their installations within the virtual world, achieving much more than they had had first expected possible. The Witch’s house and trees shown in figure 2 comprise part of one group’s installation of an enchanted forest. Other groups created a piano, obstacle course, an interactive animation of the story of the Three Little Pigs, a bowling alley and an abstract optical illusion.

Discussion

The following discussion briefly considers some of the potential applications, opportunities and constraints of SLurtles as presented in this paper.

Potential applications

As presented in the previous section, SLurtles provide an opportunity for learners to experience constructionism in action for themselves. There is potential, through the use of closed access virtual worlds such as OpenSim, for using SLurtles to teach geometry within the K-12 education sector. Finally, SLurtles may provide a approach for learners to explore and learn about the abstract concepts of an initial computer programming course through creating concrete objects rather than focusing on the syntactic complexities of code.

Persistent objects

The original ‘lineSegment’ for use with S4SL was a temporal object once placed. This limits the opportunity to share artefacts and significantly restricts opportunities to observe and reflect on constructions and part of a constructionist learning experience. Leveraging the persistent nature of *Second Life*, we have altered 'lineSegment' so each instance is no longer temporal. The persistent instances of 'lineSegment' provide an opportunity for learners to observe and reflect on their constructions. They are able to log out of *Second Life* and their creations remain for others. In addition they can alter their S4SL code, return to *Second Life* and create a new object which can be compared side-by-side with their previous construction. As a result SLurtles can be programmed and reprogrammed to create persistent artefacts which can be shared with others.

Lowering the ceiling

SLurtles lower the floor for construction within virtual worlds. However, while it leverages the affordances of *Second Life* and S4SL it is also constrained by them. LSL, while a complex programming language for novices does provide a very high ceiling with a wide range of functions available. S4SL only provides the user with a selection of these functions, limiting the functionality that can be obtained and thus lowering the ceiling. It would, however, be possible to overcome this with further development of S4SL.

Widening the walls

While SLurtles lower the floor for construction, they may also widen the walls as evidenced by the wide range of installations created by learners in the initial user trial. The variety of shapes and forms of those shapes that can be created by the SLurtles, supports the creation of a wide range of objects in the virtual world by the novice user.

Acknowledgements

The authors would like to thank members of the Centre for Research in IT in Education, Trinity College, University of Dublin, as well as the first year students of the Masters in Technology and Learning, for their support during this work. This research is supported by 'Embark Initiative' funding from the Irish Research Council for Science Engineering and Technology.

For a copy of SLurtle, please contact Carina Girvan at the address above or Sleepy Littlething in Second Life.

References

Agalianos, A., Whitty, G. Noss, R. (2006) *The social shaping of Logo*. Social studies of science 36, 241-267

Burmester, A., Burmester, F. & Reiners, T. (2008) *Virtual Environment for Immersive Learning of Container Logistics.* In Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2008 (pp. 5843-5852). Chesapeake, VA: AACE.

Castronova, E. (2005) *Synthetic Worlds*. Chicago, The University of Chicago Press

Dede, C. (2004) *If Designed Based Research is the Answer, What is the Question?* The Journal of the Learning Sciences, 13(1), 105-114.

Dreher, C., Reiners, T., Dreher, N., & Dreher, H. (2009). *Virtual Worlds as a Context Suited for Information Systems Education: Discussion of Pedagogical Experience and Curriculum Design with Reference to Second Life.* Journal of Information Systems Education, 20(2), 211-224

Feurzeig, W. (2007) *Towards an culture of creativity: A personal perspective on Logo's early years, legacy, and ongoing potential*. In Proceedings of EuroLogo 2007. Edited by I. Kalaš, Bratislava, August 2007

Gibson, J. J. (1979) *The ecological approach to visual perception.* Boston: Houghton Mifflin Company

Girvan, C. & Savage, T. (2010) *Identifying an appropriate pedagogy for virtual worlds: A Communal Constructivism case study.* Computers & Education, 55(1), 342-349.

Good, J., Howland, K. & Thackray, L. (2008) *Problem-based learning spanning real and virtual worlds: a case study in Second Life.* ALT-J, 16(3), 163-172.

Hoyles, C., Noss, R. & Adamson, R. (2002) *Rethinking the Microworld idea.* Journal of Educational Computing Research, 27(1), 29-53.

Jamaludin, A., Chee, Y. S. & Ho, C. M. L. (2009) *Fostering argumentative knowledge construction through enactive role play in Second Life.* Computers & Education, 53(2), 317-329.

Kahn, K. (2001) *ToonTalk and Logo – Is TookTalk a collegue, competitor, successor, sibling, or child of Logo?* In Proceedings of EuroLogo 2001. Edited by G. Futschek. Linz, August 2001

Kahn, K. (2007) *Should LOGO keep going forward 1?* In Proceedings of EuroLogo 2007. Edited by I. Kalaš . Bratislava, August 2007

Kirschner, P. A. (2002). Can we support CSCL? Educational, social and technological affordances for learning. In P. A. Kirschner (Ed.), *Three worlds of CSCL: Can we support CSCL* (pp.7-47). Heerlen: Open University of the Netherlands.

Kolb, D. (1984). *Experiential learning: experience as the source of learning and development*. Prentice-Hall.

Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B. & Resnick, M. (2004) *Scratch: A sneak preview*. In Y. Kambayashi, K. Tanaka & K. Rose (Eds.), Proceedings of the Second Intyernational Conference on Creating, Connecting, and Collaborating Through Computing (pp.104-109). Kyoto: Kyoto University.

Papert, S. (1972) *On making a theorem for a child.* In Proceedings of the ACM annual conference, August, 1972, (pp.345-349) Boston, Massachusetts, United States

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

Resnick, M. (1993) *Behaviour construction kits.* Communications of the ACM, 36(7), pp.64-71

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastwood, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B. & Kafai, Y. (2009) *Scratch: Programming for all.* Communications of the ACM, 52(11), 60-67

Rosenbaum, E. (2008) *Scratch for Second Life.* In S. Veeragoudar Harrell (Chair & Organizer), Virtually there: Emerging designs for STEM teaching and learning in immersive online 3D microworlds. Symposium in Proceedings of the International Conference on Learning Sciences – ICLS 2008. Utrecht, The Netherlands: ICLS. Abstract retrieved February 1, 2010, from http://www.fi.uu.ul/en/icls2008/144/paper144.pdf

Savin-Baden, M. (2008) *From cognitive capability to social reform? Shifting perceptions of learning in immersive virtual worlds.* ALT-J, 16(3), 151-161

Sheehan, R. (2000, 10-13 Sept.). *Lower floor, lower ceiling: easily programming turtle-graphics.* Proceedings of the IEEE International Symposium on Visual Languages (VL'00), Seattle, WA, USA.

Winn, W. (2005) *What we have learned about VR and learning and what we still need to study.* In Proceedings of Virtual Reality International Conference 2005. Retrieved March 14, 2008, from http://depts.washington.edu/edtech/laval.doc