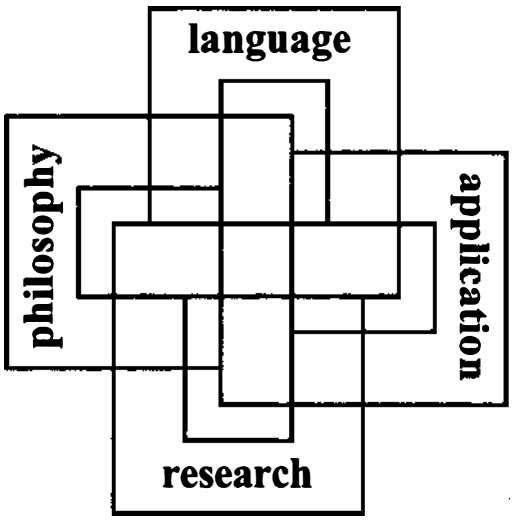
Computing in Education
Group of Victoria

Logo in Australia

Selected Readings



A collection of Australian papers on Logo
written during the period 1975 - 1996

Editor: John Oakley

Published by

Computing in Education
Group of Victoria

Statewide Resources Centre
217 Church Street
Richmond. Vic. 3121

National Library of Australia
Card Number and ISBN 0 9586879 0 0

Printed by Publicity Works

Preface

This book has been published as part of the "celebration" of 21 years of Logo in Australia. Its aims are twofold. Firstly, to record in a single volume, a selection of significant and / or seminal papers on Logo that have been written by Australians during the period 1975 to 1996, and secondly, through its very diverse chapters, to offer insights into Logo and what Logo has offered, and continues to offer education.

Logo has made an important contribution to educational thinking in Australia, a contribution that has flowed to thinking about the wider use of technology in education and, indeed, to education in general.

Throughout the past 21 years much has been written about Logo on topics that focus on the Logo language, the underpinning philosophy of Logo, classroom applications of Logo, and research on the effects of Logo in classrooms.

Australian educators have contributed significantly to each of these areas at national and international levels and *Logo in Australia: Selected Readings* is an attempt to recognise those contributions. Unfortunately, due to time constraints, difficulties in obtaining copyright clearances, and other factors, not all those authors whose papers I would liked to have included have been included.

The range of papers gives insights, not only into the historical development of Logo in Australia, but also into the underpinning philosophy and associated theories on teaching and learning and the application of those in classrooms contexts.

It is true that Logo is no longer "the flavour of the month" but it should be noted that where Logo is being used in

schools there seems to be a much more balanced and long term approach with outcomes, that in most instances, would gladden the heart of any educator. No longer are there comments such as "I used Logo last week and it didn't work!". The "bandwagon" approach to the use of Logo has disappeared and that must be a good thing.

It is hoped that serious users of Logo and those who are approaching it for the first time will find much within this book that will refresh their educational vision and encourage them to reflect on Logo in practice and what it has to offer children.

John Oakley
September, 1996

Acknowledgements

The following papers included in this publication are reprinted with permission and were originally published as indicated.

Adams, Tony (1985). Logo Environments: The Evolution of the Language. In B. Rasmussen (1985). The Information Edge: The Future for Educational Computing. Brisbane: Computer Education Group of Queensland.

Reprinted with the permission of the Author

Betts, Jenny (1992). Logo - A "Game" We Play. In W. Davey (Ed.) (1992). Computing the Clever Country? Proceedings of the 10th Annual ACCE Conference. Melbourne: CEGV.

Reprinted with the permission of the CEGV

Betts, Jenny (1992). A Child's Progression of Developing Information Handling Techniques in Logo for Databases. In S. Wawrzyniak & L. Samootin (Eds) (1992) Kids, Curriculum, Computers and... Proceedings of the 9th Annual Conference of the NSW Computer Education Group. Sydney: N.S.W. Computer Education Group.Sydney: NSWCEG.

Reprinted with the permission of the Author

Byrt, Pauline (1986). A Tool to Think With: A New Look at Some Old Problems. *Australian Educational Computing, 1*,1, pp.24-29.

Reprinted with the permission of the Author

Carter, Peter J. (1988). Stepping Out: Legged Robots in Lego. In P. J. Carter (Ed.) (1988). *POALL, 3*, 3.

Reprinted with permission of the Author

Carter, Peter J. (1989). recursion *n.* (*see* recursion). In P. J. Carter (Ed.) (1988). *POALL, 5*, 1.

Reprinted with the permission of the Author

Chambers, S. (1986). So Papert Was Right?: Evidence of General Skill Enhancement Due to Logo Experience. In A.D. Salvas & C. Dowling (1986). *Computers in Education: On The Crest Of A Wave?* Melbourne: CEGV.

Reprinted with the permission of the CEGV

Dowling, Carolyn (1992). Computing the Bossy Country? Logo as Language in the Primary Classroom. In W. Davey (Ed.) (1992). *Computing the Clever Country?* Proceedings of the 10th Annual ACCE Conference. Melbourne: CEGV.

Reprinted with the permission of the CEGV

Dowling, Carolyn (1995). What's in a Name?: Microworlds as Learning Environments. In R. Oliver & M. Wild (Eds) (1995). *Learning Without Limits*. Proceedings of ACEC'95 Conference. Perth: ECAWA.

Reprinted with the permission of the ECAWA

Gibbons, Pamela (1988). Procedurality, Modularity and Problem Structure: Introducing Logo Through Music. In *Proceedings of the ACEC'88 Conference*. Perth: ECAWA.

Reprinted with the permission of the ECAWA

Horton, Bruce. (1991) Integrating Logo into the Secondary Mathematics Curriculum. In *Proceedings of LME5*. Melbourne: ACER.

Reprinted with the permission of the Australian Council for Educational Research

Jones, Tony (1991). Teachers: Jewels in the Crown of Learning with Logo. In *Proceedings of LME5.* Melbourne: ACER.

Reprinted with the permission of the Australian Council for Educational Research

Malone, Terry (1983). Logo - A Learning Environment. In D. Blane (Ed.) (1983). *The Essential of Mathematics Education.* Parkville, Victoria: The Mathematical Association of Victoria.

Reprinted with the permission of the Author

McDougall, Anne & Adams, Tony. (1985). Teaching a Computer to Write Poetry. In J. Oakley (Ed.) (1985). *Computers in Education: Issues and Applications.* Sydney: NSWCEG.

Reprinted with the permission of the Author

McDougall, Anne (1991) Structure and Process Microviews: Partial Understandings of Recursion in Logo Programming. In *Proceedings of LME5.* Melbourne: ACER.

Reprinted with the permission of the Australian Council for Educational Research

Oakley, John (1986). Is There More to Logo Than Piaget and Powerful Ideas? In A.D. Salvas & C. Dowling (1986). *Computers in Education: On The Crest Of A Wave?* Melbourne: CEGV.

Reprinted with the permission of the CEGV

Richardson, Jeff. (1991). Interview with Brian Harvey. Melbourne: CEGV.

Reprinted with the permission of the CEGV

Robinson, Belinda. (1988). The Use of Logo in Mathematics. Unpublished.

Reprinted with the permission of the Author

Schibeci, Renato (1988). Logo in Preservice and Inservice Teacher Education. In *Proceedings of the ACEC'88 Conference.* Perth: ECAWA.

Reprinted with the permission of the ECAWA

Turner, John (1993). Logo: Has It Had Its Day ?: An Historical Analysis of Past and Present. In J. Mousley & M. Rice (Eds), *Mathematics: Of primary importance* . Proceedings of the thirtieth Mathematical Association of Victoria conference. 66-71. Melbourne: Mathematical Association of Victoria

Reprinted with the permission of the Author

Wills, Sandra (1983) Doodle Design Debug: Process vs Content Issues in Classroom Computing. A version of a paper originally presented to the *Second Annual NSW Computers in Primary Schools Conference,* Macquarie University, September 1983.

Reprinted with the permission of the Author

Yelland, Nicola, Clements, Douglas, Masters, Jennifer & Sarama, Julie (1996) Children, Computers, and Mathematical Ideas: Evaluating a Research-Based Version of Logo.

Printed with the permission of the Authors

Contents

Logo Environments: The Evolution of the Language

Tony Adams

Royal Melbourne Institute of Technology
Melbourne, Victoria

In 1982 a study of the early development and use of Logo was published, before Logo's widespread availability on microcomputers (McDougall and Adams, 1982). Some of that earlier work has been included here as context for more recent Logo developments. A companion paper (McDougall, 1985) examines the use of Logo in education and research.

Computational Models of Learning

Logo has a strong historic relationship with LISP, a major language of artificial intelligence (AI), but the connection with artificial intelligence is more than that of language. It springs from the view that AI models of learning may be used to teach children "to learn how to learn" (Papert, 1980a). Central to this view is that

> *Process models of psychology attempt to portray the individual as an intellectual agent, who constantly reformulates his knowledge, fits it together in various ways and constructs and tests theories about the way the world is structured.*
>
> (Howe, 1975, p. 296)

Howe further states that artificial intelligence addresses itself to the computational and hence detailed and precise modelling of cognitive structures. The philosophy of Logo is that tasks such as drawing and building can be modelled in computer programs. A child building a computer program to carry out a process develops a complete understanding of the process described.

This viewpoint developed by Minsky and Papert (1972) begins with the assumption that children do not know how to learn (Brady, 1975), and that computer programming provides a means for a child to overcome this by building various computational models of processes. These processes can be described in terms of procedures to carry them out, and they can be broken down into sub-processes and combined together in various ways (Solomon, 1979).

In building computational models the child will be expected to make mistakes, and seeking the bugs will lead to making new hypotheses, devising test strategies and explaining unexpected results. Typical of this approach is

> *. . . the idea of debugging itself, for example, is a very powerful concept—in contrast to the helplessness promoted by our cultural heritage about gifts, talents, and aptitudes. The latter encourages "I'm not good at this" instead of "how can I make myself better at it?".*
>
> (Minsky, 1970, p. 48)

Papert argues that skills such as juggling and riding a bicycle can be taught and discussed in terms of notions such as "state", "process" and "bug", and that these become a means both of representing and testing conceptual structures and of discussing and transferring them. He further

stresses the need to provide children with tools to identify and name the concepts needed to discuss thinking processes in a clear way. This view is seen by Howe as being in contrast to traditional problem solving activities within the classroom, which use well known and understood algorithms as recipes to solve relatively brief tasks.

Logo is seen by many workers as being consistent with artificial intelligence ideas. The following are given as reasons why Logo is preferable in this to other languages such as BASIC:

> *An interesting problem domain which doesn't rely on students having extensive formula knowledge from some other discipline.*
>
> *An obvious program trace which aids debugging, is a primitive measure of efficiency and so on.*
>
> *It encourages the notion of a process as a representation of a solution to a problem.*
>
> *Its primitive commands are simple to understand, being defined purely in terms of actions in the problem and not alterations to the state of the machine.*
>
> (Bornet and Brady, 1974:3)

Groen distinguishes between learning to program and the skills learned in Logo:

> *What is learned in Logo is not primarily a programming language. Its educational value, especially with children, comes from the fact that it provides a way of exploring microworlds. Powerful ideas are not generalised programming skills but ways of coordinating a microworld with its*

> *analogues in reality, or ways of coordinating between different representations of microworlds.... What the child is learning when involved in this [turtle geometry] microworld is not a programming language but a way of establishing correspondences between a concrete world and one of abstract representations.*
>
> (Groen, 1984:50)

According to Squires and McDougall (1985) there have been a number of interpretations of the term "microworld" and most of them have "tended to be descriptive in an attempt to convey how a microworld might be used".

Papert describes a microworld 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. 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, 1980b:204)

The aim of Logo, according to Solomon, was to build a computer culture that provided a new intellectual environment for students, to enhance the development of problem-solving skills through the writing and debugging of programming projects. Logo was not only to be a language to "talk to the computer", but an environment in which problem-solving was to take place and a meta language which includes images, metaphors, and a way of talking about projects.

a computer

a programming language and an operating system
a collection of computer peripherals usually including graphics and turtles
a collection of projects
a meta language - a consistent way of talking about the language, the projects, etc.
a relationship between teacher and learner, and
a collection of bridge activities, like juggling, puzzles, etc.
All of these components are interdependent and the special virtues of the environment follow from their coherence with one another

(Solomon, 1978, p. 21)

Early Logo Systems

The development of Logo can be seen as having gone through a number of phases.

Teletype graphics systems

Logo was developed at Bolt, Beranek and Newman Inc. (BBN) by Papert, Feurzeig, Bobrow and Solomon (Feurzeig and Lukas, 1972). In an interview (Reed, 1982), Feurzeig says that Logo was designed to provide a language modelled on LISP for children to program using strings and lists, but with a simpler syntax. At BBN, teaching sequences were developed for a number of topics including logic, algebra and problem-solving (Lukas, 1972). Teaching was carried out at both the elementary school and undergraduate level.

The introduction of turtle graphics

The turtle graphics microworld was introduced into Logo in 1970 when the first robot turtle was developed.

> *Turtles introduced many changes to the culture. It became infused with an attractive and powerful programming context, and a simple set of primitive commands.*
>
> (Solomon, 1979, p. 50)

Implementations of Logo were developed at MIT and the University of Edinburgh on PDP 10 and 11 equipment. Allison and Edmiston (1981) survey various implementations developed during this period, including a number of turtle graphics subsets. Other microworlds that were developed included music (Bamberger, 1974) and poetry (Sharples, 1978).

In the latter part of the 1970s raster graphic displays began to be used for providing turtle graphics on the screen (Abelson et al., 1976), but cost of computer equipment meant that Logo could not be used in normal school environments. During this period artificial intelligence groups working in education, including MIT, the University of Edinburgh and Xerox-Parc, were carrying out research in selected schools and were basing their work on the assumption that cheap reliable personal computers with raster displays would be available in the next decade (O'Shea, 1978).

Logo on eight-bit microcomputers

The introduction of eight-bit microcomputers in the early 1980s with minimal facilities to allow the implementation of full versions of Logo was responsible for the language moving from research environments to the classroom. Implementations of Logo by the MIT group on the Apple II and Texas Instruments (TI) 99/4 microcomputers and the widespread interest in the writings of Papert (1980a) created widespread interest leading to the development of

Logo implementations on most microcomputers in the home and educational market.

In the early part of this period, turtle graphics subsets of Logo were also in widespread use, often because classroom teachers with programming skills, but lacking access to commercial implementations, were able to create language subsets. With the commercial availability of full implementations these began to fall into disuse.

A second phase was entered in 1983 as commercial implementations began to become available. This phase has been characterised by greater acceptance by educational authorities of Logo and by the publication of Logo tests and materials aimed at classroom use.

Present implementations on eight-bit microcomputers have addressed the low-threshold aspects of Logo, providing relatively easy access to novices, at least in comparison with other commonly used languages. This has been reinforced by an extensive literature (McDougall, 1985) that has concentrated on the use of Logo with young children. The high-ceiling or advanced features of Logo have not been exploited so effectively. There are a number of reasons for this:

- The limited memory and power of eight-bit microcomputers has not enabled Logo to be used as an implementation language and has encouraged the low-ceiling aspects of the language.
- The LISP-based model of computing present in Logo is different from that of more conventional languages. Because this model is not an integral part of mainstream tertiary computing courses or of the culture of hobbyist uses of computers, it will be unfamiliar to most teachers of computing at senior secondary level.

- The developers of Logo created a low-threshold turtle graphics microworld, but the issues of providing access at this low threshold for other areas of the language (other microworlds) have only been addressed in a piecemeal fashion.

During this period no single Logo standard has emerged. Instead most Logo implementations have come from two companies, both of which have evolved from the MIT Logo group. The majority of implementations have been developed by Logo Computer Systems Inc. (LCSI), a Canadian company that was formed to develop Apple Logo (an Apple Inc. product) after licensing difficulties with MIT. Besides the "official" Apple product, LCSI implementations have included Logo for the Atari and the IBM PC, and Logotron Logo for the BBC Acorn. A smaller number of implementations have been developed by Terrapin Inc., who have distributed a version of Logo for the Apple II licensed by MIT (MIT Logo) and Commodore 64 Logo. The syntax adopted by both groups has been similar, with small differences arising from differing perspectives on language issues. More significant differences have arisen because of implementors taking advantage of machine differences and developments such as sprite graphics chips. The result of this process has been that Apple II versions of the language by LCSI and Terrapin have become default specifications for standard Logo. Most other implementors have conformed to one of these specifications to a great extent.

The Linguistics of Logo

The language described is that implemented by the MIT Logo group on the Apple II computer (Abelson, 1982). References to Apple Logo are to the version developed by Logo Computer Systems Inc. (LCSI).

Logo is a procedural list processing language, in which procedures may exist independently in a workspace and are defined separately. Procedures may be invoked at an outer command level from the keyboard by other procedures or recursively. No main program (or procedure) is required. Logo is extensible in that once defined a procedure may be used in the same manner as a language primitive. From the viewpoint of the programmer, data and procedures have the same form (a list), so that a procedure may be modified as data, and data may be executed as procedures. A procedure may modify itself and Apple Logo provides the ability for primitives to be redefined.

Procedures are of two types. A command causes something to be carried out, for example printing a result or moving the turtle. An operation returns (outputs) a result to a calling procedure. Procedures may contain parameters which are passed by value.

Procedures may contain local variables (inputs to the procedures); otherwise the most recent association of the variable is searched for. First the context of the calling procedure is searched, then the procedure that called that, and so on. Values may be assigned globally at command level.

The data structures that exist are numbers (integer and real), words (alphanumeric strings without spaces), and lists. A list is one or more numbers, words or other lists. A numeric word and a number are treated interchangeably, and data declarations are not required. Names may be assigned to data values (variables) and the values are called the "Thing" of the name. The "Thing" of a name may be a name of another value so that complex hierarchical data structures may be built up out of arbitrary combinations of numbers, words and lists.

Emphasis is on recursion for looping. A REPEAT command which runs a list of instructions a set number of times is provided, as is an unconditional transfer of control (a command very rarely used).

A number of predicates are provided (e.g. < =) which output a value of TRUE or FALSE. These may be tested singly or in combination by an IF THEN ELSE or TEST construct.

An assignment statement (MAKE) is provided but is not often used because of recursive procedure calls with modified values of inputs.

The major domain of the language is turtle geometry, and commands are provided to move and rotate the turtle (e.g. FORWARD, RIGHT). Turtle rotations are a relative number of degrees left or right of its present heading and turtle forward and backward movements leave a drawn line (its pen, that may be up—no drawing—or down— drawing). Operations exist to set the turtle at a coordinate position on the screen and to interrogate its present status.

A second domain is that of list processing. Operations are provided to add data to the beginning or end of a list (FPUT, LPUT), to take data from the beginning or end of a list (FIRST, LAST), and to take all but the beginning or end of a list (BUTFIRST, BUTLAST). A number of predicates allow for testing of an empty list or whether the data being examined is a number, word or list.

Standard arithmetic operations are provided including SIN, SQRT, RANDOM, etc.

Procedures are defined and edited with a full screen editor, and the workspace can be saved and retrieved to disk.

Apple Logo provides a number of additional facilities. These are the ability to create (define) a new procedure in a simple line editor, a startup file that allows specified procedures to be automatically loaded into the workspace at system startup, a wider range of predicates for the handling of words and lists, and property lists that allow properties to be assigned to variable names.

Logo Developments

Logo has proved to be a suitable medium in which to develop new microworlds, many of which relate to new hardware facilities or are a result of more powerful computer environments. The reason for this relates to the extensible nature of Logo syntax. New primitives can be provided at an appropriate level of abstraction and data can be executed as a program. These features (part of Logo's LISP inheritance) give Logo considerable expressive power to represent new ideas.

Logo developments can be characterised in several ways.

Towards a Complete Language

Versions of Logo implemented on 64K, 8 bit machines have not fully satisfied the conditions for a complete computer language in the sense of Pascal or even BASIC. Features that have been implemented on more powerful computers have addressed this issue particularly in regard to a wider range of file-handling facilities. The image of Logo that has dominated to date of a low-threshold language for children will be modified as versions appear on more powerful computers with substantial memories, allowing Logo to be used as a systems implementation language.

New Microworlds

The subject of microworlds has been central to Logo philosophy. The development of the sprites graphics chip by

Texas Instruments brought a limited parallel processing-animation microworld into early microcomputer implementations. These ideas have been extended by both Atari and Tandy, who with differing syntax models have provided interactive game-playing microworlds, suitable for both young children and senior secondary students. The design and programming of animated video games is a powerful idea that contains many important computer science fundamentals within an exciting and engaging microworld. In Adams (1985) I argue for the development of a logic programming microworld based on Micro-PROLOG primitives within a Logo environment. A first step towards this in the form of an interactive database is described in McDougall, Adams and Adams (1984). Such a microworld draws together the procedural versus declarative debate and provides opportunities for use of the microworld in a PROLOG sense, but also provides an opportunity for investigation of the issues.

At Atari research a number of projects have been undertaken (Solomon, 1984) including an "object oriented" version of Logo that "could be the unifying element of our play station of the future". Other work reported by Solomon includes music, gesture, robotics and animation environments.

At MIT, work has been proceeding on Boxer, a computational environment that is seen as a successor to Logo:

> *In Boxer, the language we are designing as a successor to Logo . . . computational objects such as programs are visual units (boxes) and are trivially manipulable as a whole, somewhat like a large character in a text editor. More profoundly in Boxer we have changed the conversational interaction paradigm to "looking at and directly*

> *altering the state of the system". Boxer is "editor top level": you are always able to directly change or use anything you have put on the screen. Not only does this automatically give you a simple form of concrete programming, but we can support the profitable illusion that the user directly sees the system itself on the screen.... Seeing your computational world is a much more important thing, especially in terms of long term development, than seeing only a recent audit trail of your conversation with the world.*
>
> (di Sessa, 1984:150-51)

ExperLogo, an implementation of Logo for the Apple Macintosh, introduces three dimensional turtle graphics, a microworld set out by Abelson and di Sessa (1982). A parallel processing microworld (Practical Robotics, 1984) is being developed by Cheung at the University of Edinburgh. Harvey (1984) explores a number of iterative language forms using Logo's extensible features. These provide a basis for a "computer language laboratory" suitable for upper secondary computer science studies within a Logo environment.

References

Abelson, H. (1982) *Logo for the Apple II* (Byte Books).

Abelson, H. and di Sessa, A. (1982) *Turtle Geometry* (MIT Press).

Abelson, H., Bamberger, J., Goldstein I. and Papert, S. (1975) *Logo Progress Report 1973-1975.*

Artificial Intelligence Laboratory (1976) *Logo Memo No. 22* MIT.

Adams, T. (1981) "Logo: Where to Next?" *Logo 85 Pre-Proceedings* (MIT, publication pending).

Allison L. and Edmiston, P. (1981) "A Logo Survey" , *Australia Computer Bulletin,* October 1981.

Bamberger, J. (1974) "The Luxury of Necessity", *Logo Memo No. 12* (Artificial Intelligence Laboratory, MIT).

Brady, J. (1975) Introducing Logo Education, *Computer Education,* no. 19, pp. 24-27.

Bornet, R. and Brady, J. (1974) "The Linguistics of Logo", *Memo No. CSM-4* University of Essex.

di Sessa, A. (1974) "Notes on the Future of Programming", in *Logo 84,* ed. R. Sorkin, (1974) PreProceedings of the National Logo Conference, MIT, pp. 149-55.

Feurzeig, W. and Lukas, G. (1972) "Logo: A Programming Language for Learning Mathematics", *Educational Technology,* March 1972, pp. 39-46.

Groen, G. (1984) "Theories of Logo", in *Logo 84,* ed. R. Sorkin, Pre-Proceedings of the National Logo Conference, MIT, June 1984, pp. 49-54.

Harvey, B. (1984) "Iteration in Logo", in *Logo 84,* ed. R. Sorkin, Pre-Proceedings of the National Logo Conference, MIT, pp. 113 -20.

Howe, J. "Artificial Intelligence and Education", in *Computer Assisted Learning in the U.K.,* ed. R. Hooper, Council for Educational Technology, London, pp. 295-317.

Lukas, G. (1972) "Logo Teaching Sequence on Strategy in Problem Solving and Story Problems in Algebra", *BBN,* Cambridge, Mass.

McDougall, A. (1985) "Logo: Its Use in Education and Research", in *The Information Edge: The Future for Educational Computing,* ed. B. Rasmussen, Computer Education Group of Queensland.

McDougall, A. and Adams, T. (1982) "Logo Environments: The Development of the Language and Its Use in Education and Research", in *Proceedings of the Ninth Australian Computer Conference,* Australian Computer Society, pp. 116-32.

McDougall, A., Adams, T. and Adams, P. (1984) *Learning Logo on the Commodore 64* Pitman.

Minsky, M. (1970) "Form and Content in Computer Science", *Journal of the ACM,* vol. 17, no. 2, pp. 40-58.

Minsky, M. and Papert, S. (1972) "Artificial Intelligence", *Memo No. 252,* Artificial Intelligence Laboratory, MIT.

O'Shea, T. (1978) "Artificial Intelligence and Computer Based Education", *Computer Education,* no. 30, pp. 25-28.

Papert, S. (1980a) *Mindstorms: Children, Computers and Powerful Ideas* Harvester.

Papert, S. (1980b) "Computer Based Microworlds as Incubators for Powerful Ideas", in *The Computer in the School: Tutor, Tool, Tutee,* ed. R. Taylor, Teachers College Press, pp. 203- 10.

Practical Robotics, September/October 1984, p. 8.

Reed, K. (1982) "An Interview with Wallace Feurzeig", *COM-3,* Computer Education Group of Victoria, no. 29, p. 9.

Sharples, M. (1978) Poetry From Logo", *Working Paper No. 30,* Department of Artificial Intelligence, University of Edinburgh.

Solomon, C. (1978) "Teaching Young Children to Program in a Logo Turtle Culture", *Sigcue Bulletin,* vol. 12, no. 3, pp. 20-29.

Solomon, C. (1979) "Language in the Logo Computer Culture", *Proceedings NECC,* Iowa City.

Solomon, C. (1984) "Logo: Past and Future", in *Logo* 84, ed. R. Sorkin, Pre-Proceedings of the National Logo Conference, MIT, pp. 125-30.

Squires, D. and McDougall, A. (1985) "What is a Computer Based Microworld?—An Interpretation to Aid Design", publication pending.

Computing the Bossy Country? Logo as Language in the Primary Classroom

Carolyn Dowling

Australian Catholic University (Mercy campus)
Ascot Vale, Victoria 3032

The traditional presentation of Logo as 'mathland', faithful as it might be to Papert's intentions, can be discouraging, even intimidating, to teachers who lack confidence in this area. While the development of LogoWriter has very obviously legitimised an increasingly language-oriented approach, such a focus has always been possible, and has frequently provided an alternative, more accessible gateway into Logo for both teachers and students (Dowling 1985; Goldenberg & Feurzeig,1987).

One way in which this can be achieved is, of course, through an early emphasis on list processing rather than on graphics. Programs can be written which encourage reflection on language issues such as the dynamics of conversation (interactive 'chat' programs), what constitutes a 'poem', why certain sequences of words are grammatically more acceptable than others (templates for the generation of particular sequences of word forms), and so on. Examples of such activities can be found in many Logo books and manuals.

More broadly, however, both the structure and the overtly metaphorical terminology of Logo in all its forms invite an

approach which draws heavily on a range of analogies with natural language. In this regard, what sorts of connections might be made in the early stages between Logo and children's other language experiences? What are the less obvious implications of a focus on Logo as language? What does this focus suggest on the one hand about Logo and computers, and on the other hand about language? How and what can children be said to learn from the 'languageness' of Logo?

Computer programming languages hover uneasily around the edges of our common-sense understanding of what a language is. Though they share some features of natural language they also lack many of its qualities and attributes, while possessing others of their own. The question as to whether the shared aspects justify the unproblematic characterisation of Logo as a language, or whether the expression, 'the Logo language', should generally be regarded more as a figure of speech, depends on the context in which the classification is being made. While writers such as Ong (1982) and others might deny the validity of regarding "so-called 'computer languages'" as commensurate with natural languages by virtue of their being consciously constructed systems rather than having organically 'grown', there may well be purposes for which a degree of analogy might shed light on interesting aspects both of language and of computers.

Papert (1980) certainly expresses his vision of the 'languageness' of Logo in a variety of ways at different conceptual levels. For him Logo is primarily the language of 'mathland', with a vocabulary and syntax strongly influenced by aspects of the English language, tempered by the constraints of the computational facilities available and also, to a degree, by the conventions of programming. Children are described as learning Logo much as they learn

a natural language - by a combination of immersion and purposeful use. He writes of a group of student Logo-learners that, "...like children who have spent a vacation with foreign-speaking cousins, they were clearly on their way to 'speaking computer'." He suggests that, "Children working with an electronic sketchpad are learning a language for talking about shapes and fluxes of shapes, about velocities and rates of change, about processes and procedures. They are learning to speak mathematics...". (On the other hand, while the Logo environment in general is replete with language imagery, the question of how the computer, (or is it the turtle?), comes to understand and to 'speak' Logo is rarely addressed either by Papert or by other writers in more than a cursory fashion.)

The validity in detailed terms of this analogy with natural language learning is not at issue here; what is of interest is the initial setting of a scene in which Logo is to be regarded as commensurate in a number of important senses with natural language.

In the interests of accessibility, the vocabulary and surface structures of the original versions of Logo were certainly very strongly based on those of the English language, to the extent that some of the underlying structures of Logo were in fact rendered extremely complex through the need to achieve a resemblance to English syntax (du Boulay et al, 1980). Thus one would reasonably expect it to resonate most closely with the broader language experience of children whose first language is English, and to a greater or lesser extent for native speakers of other languages according to the degree to which those languages are similar to English. In the case of translations of Logo, this depends on the strategies chosen by the translators. In the case of

Hungarian Logo, for example, Turcsanyi (1985) writes: "Since Logo is based on the English language, it cannot be a natural means of expressing thoughts for a Hungarian child, due to the grammatical and alphabetical problems of a foreign language. We tried to 'nationalize' it by thoroughly studying the philosophy of Logo and implementing it to our specific culture".

This does, of course, beg the question of the degree to which even English speaking children find Logo a 'natural means of expressing thoughts'. This is not easy to assess, since it involves not simply details of vocabulary and syntax, but also the child's acceptance of the numerous other levels at which language-related images and concepts operate within Logo, such as the idea of a procedure as a 'new word', the concept of the LogoWriter 'page', the status of program as text, the role of the turtle as language-user and so on. In addition to these considerations, the level of natural-language experience brought to the situation by individual users, including the extent to which they are print-literate in their native tongue, and the type of programming activities undertaken, clearly have a significant bearing on the degree to which Logo is perceived as an analog of natural language.

The development of LogoWriter has, of course, made both an initial and a continuing emphasis on language much easier to achieve. In addition to the increased use of language-related imagery, particularly that concerned with the conventions of more familiar print-based environments ('scrapbook', 'page' etc.), the capacity to integrate the 'language' of Logo with the manipulation of text written in the child's natural language opens the way to an enormously increased flexibility in the ways Logo can be used as language in the classroom, both in the early stages and

for more advanced work. LogoWriter makes more apparent the qualities of Logo as interactive electronic text. Contrary to a general perception, reinforced by the titles of a number of Logo books, that turtles 'speak' Logo, it is, in virtually all its manifestations, a purely textual code embodied in an electronic medium. The obvious ways in which, for instance, the LogoWriter 'page' (three-sided since the advent of Logo Ensemble!) differs from its hard copy equivalent, points to a range of more subtle distinctions between this and the products of more traditional print media. Apart from the questions this raises concerning the status of Logo procedures (or indeed programs in any other computer language) as text, the integration of LogoWriter with an electronic mailing system in Logo Express links text created within this environment to more general concerns as to the place of this form of textual experience within the whole gamut of language experience (Ong, 1982; Dowling 1987; Poster 1990; Dowling 1991).

Although perhaps not entirely in accordance with the tenets of the whole language approach, it can be argued that children may well gain a certain degree of insight into a more abstract concept of language in general through the experience of communicating in a code which is very clearly simplified and restricted, but which is perceived to share some features of their native tongue. This is similar to the rationale often given for the study of additional natural languages by children. Arguably more useful than what is learnt of the second language itself, in this case Logo, is the attention paid, through consideration of both similarities and differences, to aspects of how languages in general 'work'. For instance, quite young children working with Logo quickly become aware of issues critical to text-based language use such as the significance of spaces in delineating

the boundaries of printed 'words', or the importance of standardised spelling - at least when addressing a computer!

But while obvious similarities between Logo and the English language do exist, it may well be that the most interesting and pedagogically fruitful points of comparison between the child's natural language and Logo will spring from the differences between them. Many of these differences which a child might perceive are apparently consequent upon the fact that Logo has been designed as a language of communication between human being and computer, rather than between one human being and another. A simple example is the turtle's literal and limited response to the command *LT whatever*, as compared with the context-dependent assumptions which would determine our reaction to such an instruction. Having students use Logo to communicate with one another, that is, with one student 'playing turtle', clearly brings out some of these differences. In addition to the importance of an emphasis on 'difference' in concept formation, such a focus militates against those aspects of Logo which are not consonant with natural language being incorporated without consideration into children's models of language in general.

But the common perception that most of the differences between Logo and natural language are simply the consequence of one of the parties to the act of communication being a computer rather than a human being, becomes a more complex issue when considered in the light of the different levels of personification which operate within the Logo environment. An examination of a number of the early Logo books and manuals on which our present philosophies and practices are based, reveals multiple levels of anthropomorphism. Depending sometimes on context and intention, and at other times apparently on mere whim, the

turtle (Turtle?), the computer and even Logo itself are variously accorded a range of human features and capabilities. These ideas are intimately connected with the ability of a computational object to respond to and through language - traditionally capacities regarded as largely definitive of a human being. Thus the degree to which the principle of personification is accepted by children is bound up in a circular fashion with the perceived 'languageness' of Logo, and so also with the extent to which its features are incorporated into their general understandings about language.

One of the key principles on which Logo was founded was that of empowerment - of giving the user control over the capabilities of the computer. (Although the user most typically envisaged is of course the child, this type of experience has been acknowledged as desirable in other contexts. As one writer comments: "Logo will give learners experience of success and control, both sadly lacking in a prisoner's inventory of experience" (Stone 1985).)

This aspect receives extensive emphasis both in writings about Logo and the so-called 'Logo philosophy', and within the terminology and structures of the language itself. While at many levels it can only be applauded as a counterbalance to a number of other perceptions of the child's rightful place and role in the world, the issue of personification mentioned above raises questions concerning the precise status of what is being controlled, and of the language through which the control is exercised. How does this type of control and empowerment relate to perceptions of what it takes to be 'clever' in the electronic age?

Western thought and culture traditionally support a clear disjunction between the animate and the inanimate. While power and control over the inanimate is generally condoned,

and in fact constitutes the goal of many major fields of endeavour such as most aspects of western science, power over the animate, and particularly over one's fellow human beings, is a different matter. It runs counter to the major thrust of most of our generally accepted philosophies. It is widely perceived, for instance, as 'undemocratic'. To exercise power over other people is generally to be regarded with mistrust and suspicion. While the exercise of authority over an electronic device may be acceptable, power over a significantly 'personified' computational object is more equivocal. Where does the act of commanding a Logo turtle situate us within such a context?

Themes of power, control and domination, characteristic of what is often described as the masculine or hard-science approach to the world, pervade the Logo environment, from descriptions of its aims and mode of operation through the syntax of its 'commands' , the metaphor of the LogoWriter 'command centre', the self-deprecating error messages of the hapless turtle (or turtles), the particular model of 'teaching' often utilised as a metaphor for the creation of procedures, and so on. One version of Logo (Lego TC Logo) even incorporates this ideal into its name and trademark.

Although rarely discussed in such terms, a significant manifestation of language as power is embodied in the apparently straightforward act of 'naming' a procedure. The degree of power and control traditionally implied by the naming process, in many cultures by the mere *knowledge* of a person or object's true name, reflects the significance of such symbolic representations (Salmond 1982). Thus while at one level the metaphor of procedure writing as 'teaching the computer a new word' supports the general analogy between Logo and natural language, and while it can provoke quite sophisticated consideration of the nature and

limits of definition and the means of creation of new linguistic items in natural language, in the wider human context this image involves far deeper issues concerned with the relationship between language and power.

With these ideas in mind, if we are to regard Logo even for limited purposes as commensurate with natural language, what styles of utterance does it support? Although several modes of language use are possible (Kieran 1985), at the level of use of most primary school children it is a procedural language, a medium for the delivery of instructions, and its mode of interaction is most typically the imperative mood. Through the language of Logo, children command. This style of language use is certainly more typical of child to child or less happily of adult to child than of adult to adult in our society, precisely because we go to great lengths to discourage it. In a culture governed at least in theory by notions of negotiation, compromise and consensus the undisguised imperative is indicative of a generally unacceptable level of verbal assertiveness. As one teacher protested recently, "I don't like Logo, it's such a *bossy* language." From the point of view in particular of the younger child, this mode of address, legitimised within the Logo environment, might well seem attractive, but is it a style of language use which we are happy to see so strongly endorsed by association with the high status activity of programming a computer?

There are several possible responses to this concern. On the one hand this very aspect is sometimes applauded by teachers as providing an opportunity for children to practice a type of language use which is not encouraged in most classroom situations, but which might be 'useful' in later life. Some writers and practitioners quite clearly elevate this feature of Logo to an end in its own right. One early introductory text,

for instance, in claiming that Logo activities will 'increase children's expertise in commanding', talks quite explicitly in terms of 'exercising their wills' and 'gaining a sense of power' (Bitter & Watson 1983).

Perhaps the most common response is that this is a wholly appropriate way for a human being to convey his or her wishes to a mere machine, in fact that this style of language forces and reinforces a desirable separation and hierarchical distinction between user and machine. One is not, after all, addressing a person. The computer being a particularly strong contemporary symbol of power and authority, the experience of control over even a limited range of its formidable capacities should certainly be significant in terms of the user's self-image, and desirable as a means of supporting the continued dominance of humanity over the world of things. However this attitude, and the aspects of Logo which support it, would seem to be significantly at odds with the lengths to which all of the developers of Logo have gone to encourage varying but in all cases substantial degrees of personification within the Logo environment.

Just as it is difficult to judge the degree to which Logo is accepted as being equivalent to a natural language, so it is not easy to ascertain in the case of individual users the degree to which the turtle, the computer, the language itself, even the procedures, all of which partake of varying degrees of personification within the literature of Logo, are actually anthropomorphised as part of the Logo experience. Particularly in view of such insights as those of Sherry Turkle concerning the confusion experienced by younger children regarding the boundaries between the animate and the inanimate (Turkle 1984), it seems over-optimistic to expect that a simple caution such as that quoted below would suffice to set the record straight. "When we talk

about Logo as if it were a *person* that 'knows', 'understands', 'wants to help you', etc., we do so because it helps us think about solving problems with the computer. We know that Logo isn't *really* a person, but if we visualise Logo as a personality, it helps us understand its behaviour at a number of tricky points" (Watt 1984). As mentioned earlier, the involvement of language capacities in addition to the more traditionally recognised criterion of movement in regard to the distinction between the animate and the inanimate increases the complexity of this issue.

The whole matter is further complicated by the degree to which the mind/machine metaphor of AI, a conception which involves yet another once definitive feature of our humanity, has become part of everyday discourse. The computer as mirror and metaphor both reflects and shapes our image of ourselves, to the extent that the much exploited science fiction theme of the human/computer partnership as cyborg now features in a growing number of more scholarly contemporary writings, not necessarily in obviously 'computing' contexts (Hayles 1990; Bigum 1991; Spinrad 1991).

If we are to accept that, in the Logo computing environment, the boundaries between the animate and the inanimate, the person and the thing, are blurred, whether consciously or not, what are we to make of the emphasis on power, command and control as manifested in the Logo language? Is it a 'what' or a 'who' over which the child is exercising such peremptory authority? Does such experience of language use go beyond promoting a desirable degree of influence over life's circumstances? (In connection with the ubiquitous 'little people' metaphor in Logo programming, Solomon writes, "We can conjure up these little people whenever we need help. We can simulate them,

that is, be them for a while, talk to them, dispose of them!" (Solomon 1986).) However we express the issue, whether as a question of tone, register, style, or in other terms altogether, the bottom line seems to be that this is the language of uncompromising domination which defines or at the very least reinforces a particular and very limited type of relationship between computer and user; a relationship which, moreover, sits most uncomfortably with the complexities suggested by a computing environment so saturated with anthropomorphic imagery.

Are there alternatives to 'bossing' the turtle? Within the constraints of the language as given, these are not obvious. Short of rewriting the primitives, little can be done to soften the immediate impact of the language of command. The adoption of a more open-ended, 'discovery' oriented attitude to Logo programming offers some degree of amelioration of the overall ethos of command and control, although many writers and practitioners are wary of the possible excesses of such approaches.

While the ethos of power and domination sits comfortably with the more simplistic characterisations of the computer as a 'tool', more recent conceptions of its role incorporating ideas of partnership and collaboration, including themes of mutual and reflexive definition implied by the image of the cyborg and much of current thinking in regard to artificial intelligence, suggest that this may not, in fact, be a 'clever' way to relate to computers. Papert himself has acknowledged in a number of contexts the need for flexibility in relation to how computers might be conceived. In *Mindstorms* (Papert 1980) he refers to the computer as "the Proteus of machines", and elsewhere says of Logo: " If there is anything that Logo promises, it's the protean ability to take different forms ... and, if you use it right, to become

a kind of mirror in which you can see reflections of yourself (Papert, 1985). In his introduction to Brian Silverman's 'The Phantom Fishtank', an exploratory 'microworld' written in Logo, he writes that: ".. a whole software industry is committed to the idea that programmers will make computers do what they are told - no more and no less. But there are times when the thrill comes from the computer doing something that one did not anticipate and perhaps could not have imagined even if one has tried. The excitement is not [in] what was specified but in what emerges" (Papert 1987). This is the expression of a very different attitude to the relationship between programmer and computer from that initially fostered by Logo; an attitude more in keeping with contemporary developments in various fields of computing, and certainly potentially more responsive to the varied 'thinking styles' of human beings (Thornburg 1985; Dowling 1991).

Is the apparent tension within the Logo environment between a tendency on the one hand to anthropomorphise the turtle/computer and on the other to accentuate the sense of power and domination of the human user over the technology counter-productive, in that each of these tendencies works powerfully towards undermining the other, or does it, on the contrary, provide children with a richly ambivalent experience of language and computing which embodies very real contradictions and complexities in the developing relationship between people and computers? Acceptance of the latter point of view would suggest that Logo has an even more interesting role to play in the classroom than might have been envisaged; one which goes beyond deliberate intent, to ambiguities inherent in the way in which we currently relate to computers within a culture which the technology both serves and shapes significantly.

In this sense the experience of Logo as language in the primary classroom might be seen as making a unique contribution, albeit it in an indirect but perhaps nonetheless effective way, to the unsettling, even the disruption, of some of the narrower but commonly held perceptions of the role of computers in our lives. Through the embodiment in its mixed images of apparently contradictory elements in our relationship with computing technology, experience with Logo has the potential to leave children more receptive than they might otherwise be to a wide range of possible modes and styles of interaction between human and computer. If the conception of a 'clever' country embraces long-term adaptation rather than short-term conformity, such consequences should be applauded.

References

Bigum, C. (1991), 'Schools for cyborgs: Educating aliens', in Proceedings of the Australian Computers in Education Conference, *Navigating the Nineties*, Brisbane.

Bitter, G.G & Watson, N.R., (1983), *Apple Logo Primer,* Virginia: Reston Publishing Company Inc.

Dowling, C. (1985), 'Logo and Language' in conference proceedings, *Logo in Australia ... Ten Years On,* Melbourne.

Dowling, C. (1987), 'Mind Your Language: Computers and Communication', in Proceedings of the Australian Computers in Education Conference, *Tomorrow's Technology Today,* ed. Hancock, J., Adelaide.

Dowling, C. (1990), 'Exploring the Unpredictable: the computer as a catalyst for creative thinking in the classroom', in Proceedings of the 5th World Conference on Computers in Education, *Computers in Education*, ed. McDougall, A. & Dowling, C., Amsterdam: North Holland.

Dowling, C. (1991), " 'Writing' with computers - a metaphor?", in *Australian Educational Computing,* Vol. 6, No. 2, pp. 38-40.

du Boulay, B., O'Shea, T. & Monk, J. (1980), 'The Black Box Inside the Glass Box: Presenting Computing Concepts to Novices', paper submitted to the *International Journal of Man-Machine Studies.*

Goldenberg E.P. and Feurzeig W.(1987), *Exploring Language,* Cambridge, Mass: MIT Press.

Hayles, N.K. (1990), *Chaos Bound: Orderly Disorder in Contemporary Literature and Science,* Ithaca: Cornell University Press.

Haraway, D.J. (198?), *Simians, Cyborgs and Women: The Reinvention of Nature*, FA books.

Kieran, T.E (1985), 'On Logo, Language Use and Thinking', precis of presentation for World Logo Conference, Vancouver.

Ong, W.J. (1982), *Orality and Literacy: The Technologizing of the Word*, London: Routledge.

Papert, S. (1980; this edition 1982), *Mindstorms: Children, Computers and Powerful Ideas,* Brighton, Sussex: Harvester Press.

Papert, S. (1985), 'Different Visions of Logo' in *Computers in Schools,* Vol. 2, nos. 2/3, excerpt reprinted for World Logo Conference, Vancouver.

Papert, S. (1987), Foreword in Silverman, B., *The Phantom Fishtank: An Ecology of Mind,* Canada, Logo Computer systems Inc.

Poster, M. (1990), *The Mode of Information: Poststructuralism and Social Context,* Cambridge UK: Polity Press.

Salmond, A. (1982), 'Theoretical Landscapes : On Cross-Cultural Conceptions of Knowledge', in *Semantic Anthropology,* ed. Parkin, D., London: Academic Press.

Solomon, C. (1986), *Computer Environments for Children: A Reflection on Theories of Learning and Education,* Cambridge, Mass: MIT Press

Spinrad, N. (1991), 'Virtual People', in *Isaac Asimov's Science Fiction Magazine,* Vol.15, No.15, pp. 162-174.

Stone, A. (1985), 'Logo in Corrections Education', in conference abstracts, *Logo '85,* MIT.

Thornburg, D.D. (1985), 'The Left Hand of AI: Bringing Design Students to Logo', presentation for World Logo Conference, Vancouver.

Turcsanyi, M. Sz. (1985), 'Logo in a University Education: Experiments and Plans', in conference abstracts, *Logo '85,* MIT.

Turkle, S. (1984), *The Second Self: Computers and the Human Spirit*, London: Granada.

Watt, D. (1984), *Learning with Apple Logo,* New York: McGraw Hill.

Interview with Brian Harvey

Jeff Richardson

Faculty of Education, Monash University
Clayton, Victoria. 3168

Introduction

Brian Harvey is a Professor of Computer Science at the University of California, Berkeley. He is the author of the influential and respected 'Computer Science Logo Style' (MIT Press 1985). Brian was in Australia during the autumn of 1991 for a series of conferences. The following is a transcript of an interview Brian gave to Jeff Richardson of Monash University College Gippsland for ABC Radio National's "Education Now".

The Interview

Jeff: Brian, in your preferred world, what preparation would you like for students entering the University who are planning to study Computer Science?

Brian: Well actually I don't care very much. I would be perfectly happy to have students coming in not knowing anything about computers. That would be OK. We can handle it. I would like to see students with a mathematical sense; not so much knowing any particular mathematical thing but feeling comfortable with the idea of formal notation, and thinking about manipulating symbols in a formal

way and both being able to do it and liking doing it too. One thing that I find is that students come into computer science courses thinking that computer science is just the study of programming techniques, and more programming techniques, and maybe lots of programming languages or something. Then they are very surprised when they hit theoretical computer science and we start introducing them to mathematical theory about computing, and they say "what has this got to do with it?". If they are going to learn to program I'm happiest if they have some exposure to the ideas of functional programming. That is to say, the idea of a program as representing a function. Certainly input/output functions; like for example, starting with some values, cranking them through and getting a result. And I want them to have the ability to combine functions to take the result of one function and use that as the input to another function. I see this in contrast to the more traditional, sequential, first do this, then do this, then do this style of programming. But even that isn't such a big deal.

Jeff: Is there anything in students' high school experiences that's bringing them to a certain viewpoint about computing that gives them inappropriate expectations about tertiary computer science?

Brian: Well, I don't know if it's their school experiences so much as their kind of street experiences. They have computers at home and the computer probably came with a BASIC interpreter and they've been playing around with it. And then they

read in a magazine about other programming languages and maybe they've gone off and learnt Pascal or they've learnt C or they've done some combination of things. Or they may have even learnt something I actually like, like Lisp. But this is all about programming and the thing is that I don't want to be understood as thinking that that's bad. I think it's great for kids to program. It's a lot of fun and they should do it. It's just that in addition to that they need to enjoy more formal mathematics.

Jeff: You like children to write computer programs. In certain educational circles that's a very unfashionable statement at the moment. There's a conventional wisdom that you don't have to learn to program the computer, that programming is the wrong use of computers in the pre-tertiary school setting. Could you expand a bit more on why you would like children to write programs?

Brian: Yeah. The conventional wisdom has gone back and forth several times on this. The first idea was that programming computers was too hard. And then programming computers was the greatest thing in the world and furthermore everybody was going to have to do it. And now the idea is that nobody is going to have to do it. I think all of those points of view are much too narrowly focused on vocational training and trying to second guess what people are going to need in their jobs twenty years from now. One thing is, I don't want to argue that every kid should do anything in particular in high school. I can see some things like that in primary school, you know, everyone should learn to read. I think

it's good if everyone has some exposure to beginning programming, back in elementary school. As far as high school is concerned my own feeling is that it's a great time for apprenticeship. I talk about that word a lot. This idea goes really against the grain much more than anything specifically about programming. Kids, teenagers, very often get tremendously hooked on something or other, and I think it almost doesn't matter what it is. Our job is to encourage that interest and try to encourage them to take the interest seriously, whatever it is. It turns out that not every kid, but a lot of kids, get excited about computer programming. Instead of either saying "well that's right because every kid should program" or "it's wrong because nobody should program", what we should do is take those kids who enjoy programming and give them access to expertise. First off so that they learn to do it right. Then we should provide an environment in which they have serious work to do. That is, not doing exercises to prove to the teacher that they can do it, but solving real problems. A lot of kids in school get the kind of experience I'm thinking of in sports. They join a team of some kind and their work is taken seriously. They're not just getting a grade for something. They're doing something that people really care about. Some kids get it in drama, some kids get it in the school newspaper. It's very very rare for a kid to have that kind of experience in anything scientific or technical or mathematical.

Jeff: In Language Arts, students refine their work for publication as a standard and proven part of the

curriculum. Are you proposing a similar strategy for programming in particular and science in general?

Brian: I can think a lot of examples of public performance, that's true and that is part of it but what I'm really on about is the realness of the work. When I was at school they used to have kids who went around and made sure the movie projectors worked when a teacher wanted to show a movie. That wasn't a terribly creative thing to do or anything but it was real work. Now because it wasn't terribly creative it would be foolish for a kid's entire school experience to be about making a movie projector work, but nevertheless that really stood out for a lot of people as something much more real than all the stuff that they were learning. It would be good to have that kind of experience about something technical so that kids who might develop an interest in that direction have the opportunity. And I don't think it matters if they end up being professional computer programmers when they grow up or not. Maybe they'll end up being a physicist or something instead. But it's pretty hard for a kid to have that kind of apprenticeship experience doing physics in school because although you can learn about physics that somebody else did, it's hard to do new physics unless you have cyclotrons and electron microscopes and all that kind of stuff. Whereas it's pretty easy, relatively speaking, to give kids that kind of experience about programming. Although you need computers, mostly what you need is to be at the cutting edge in software, and software is

affordable and easy. So I think that it's the realness of the experience of being able to write a program that somebody else uses that really counts. When I taught high school I had a bunch of kids who spent close to all their time in the computer centre, and I would encourage them, even if they didn't do so well in their other classes as a result. I thought it was more important. They could learn the other stuff some other time.

Jeff : What have you got to say to the social critics of computer use? Those who on hearing a story of the students spending all of their time in the computer centre see an image of children addicted to the machine.

Brian: Yes, there are really two different criticisms like that. One is about the whole idea of concentration. The first thing I would want to know is, suppose instead the kid were passionately interested in art? They went around drawing everything all the time, keeping sketch books, taking art classes. Would the critic have the same objection? Some would and some wouldn't. OK, if they would, if it's a question of kids shouldn't concentrate - it's kind of funny because when it comes to college students or undergraduates, which is who I teach now, I take the opposite view. I think undergraduates should have a broad liberal education and this is one of the things I argue about with my colleagues. I teach a lot of students who are signed up for engineering degrees and they take nothing but engineering courses. I feel that they are not getting an education. But I think that high school educates teenagers at a time when it just makes a lot of

sense to do a specialisation. It fits in with what they're ready for educationally and socially at that age. The kind of intensity and idealism that high school kids display fits in well with focussing on something and at the same time they are not ready yet for some of the broader things. Kids take high school subjects even if they are not really interested in them, and even if they get A's in the course, they don't see it as connected with their life in any way. They will be ready for that a little later. I sort of see both sides. I would agree that people shouldn't get so narrowed down permanently that they don't get a broad education, and if that's what the critics' fear I agree with it. But that sort of temporary narrowing down at the high school level isn't problematic for me. I tend to push it maybe to an extreme because I'm reacting to the kind of lock step, 'everyone has to learn these six subjects through high school' curriculum that is much more common. OK, now the other kind of objection is specifically about computers. It's the notion that there's something about machinery that's dehumanising. All that I can say is that it can be and it doesn't have to be. Actually, I think, when it looks like that it's probably more the other way around. Somebody who somehow isn't getting along well with human beings finds the possibility of working with computers as a relief, at least they can do something right.

Jeff: So it's a symptom rather than a cause?

Brian: Yeah, but I've mixed feelings about that too. I think it requires sensitive understanding on the

part of the teacher. It's something for teachers to watch out for. I think it can turn out to be a good thing. With some of my high school students I've had parents of kids come to me and say "the computer centre is the best thing that ever happened to my kid because he used to not have any friends and now he does". It can be a way in to human contact. If you find people who are the same kind of person you are, you can branch out after that and feel a little more confident with people. One of the things that I used to see that was very amusing was a boy and a girl at opposite ends of the room typing frantically at a computer terminal. They were talking to each other through the medium of the computer because there were things they wanted to say to each other that they just couldn't say face to face. That was sort of an interesting experience - I don't know whether it's for everybody. Certainly I think that the idea that if you use machinery as a tool you have to be like a machine, that's just not so.

Jeff: What is for everybody? Aside from those students who have either aptitude or interest for programming that you obviously want to encourage. Pragmatically schools have to face up to some sort of core curriculum. A vogue at the moment is to describe computer use in schools as Information Technology. Have you got a view on what is an acceptable core for computer use for all children?

Brian: My feeling is that a lot of the fuss about this is just an immediate temporary problem coming from the fact that the grown ups, the people who do the

curriculum planning, grew up in a world without computers. It's all new to them and they're a little panicked about the situation. Kids grow up with computers and are much more relaxed about the whole thing. Certainly computers are a terrific tool for everybody. I don't understand how anybody ever wrote anything before there were word processors. I've written a series of books and I certainly could not have done it with typewriters and so on. Every kid ought to have that opportunity starting in the very earliest grades. By the time they get to high school they should just know. If they don't know, we should teach them quickly. I don't like word processing courses, spending a whole semester learning to do word processing. There's just not that much to it.

Jeff: How about the other software environments that tend to have courses about them, data bases and spread sheets?

Brian: Much the same thing. I think that they should be available to everybody from early grades. One thing that there is an argument about is whether a kid should have the same software that grown ups have or whether we should design special simplified software for kids. I tend to lean to real software for kids. Without making it a big fuss kids should just have these tools available. A ninth grade class spending two weeks learning how to use a word processor or learning how to use a spread sheet or whatever is quite enough. I just don't think it's a big subject to make a big fuss about.

Jeff: What's your advice to any student who does have the interest and aptitude to go on beyond

secondary school with computing? What should they focus on?

Brian: Well, I teach a course at Berkeley that was developed at MIT by Hal Abelson and Jerry Sussman. The course is called "Structure and Interpretation of Computer Programs". It's taught in Scheme which is a little unusual. It's a language that comes from artificial intelligence work, but the point isn't about the language. The course talks about big ideas, about different ways of thinking about organising a computer program. There's functional programming which I mentioned before, object oriented programming which is a very hot buzz word these days, logic programming and ideas like that. The more typical course that you get even at the university introductory level spends a lot of time on the particular grammatical rules of some programming language, and I think that's a terrible waste. The kind of programming that you do if you are a kid and you are on your own with a computer really is perfectly good as long as you are writing programs that can sort of fit in your head all at once. When you want to deal with very large complicated systems you start needing help at being able to deal with one piece of it at a time, so you can isolate one piece and work on that separately and still have them all fit together properly. Those structuring ideas are really what we are trying to get across in computer science.

What's in a Name? Microworlds as Learning Environments

Carolyn Dowling

Australian Catholic University (Mercy campus)
Ascot Vale, Victoria 3032

The release of LCSI's Microworlds in 1993 has contributed to a resurgence of interest in the image of the 'microworld' as a description of certain computer-based learning environments. Further, it is a concept which resonates with a range of current philosophical and pedagogical concerns relating to the 'virtuality' of computing environments in general, and in particular to roles which computer based simulations and more recently virtual reality might play in educational contexts.

The term 'microworld', deriving initially from work in the area of artificial intelligence, is ubiquitous in early writings about Logo. It is generally used in two senses, firstly as a metaphor for the overall Logo environment, ("Logo is a microworld that is meant to be shared" (Tobias 1984, p. 95)), and secondly in reference to programming environments defined within the larger context of Logo itself, which display particular characteristics and have specific pedagogical purposes (Papert 1982; diSessa & White 1982; Lawler 1982, 1985, 1986; Squires & Sellman 1985;

Feurzeig 1986; McDougall & Squires 1986). It is in this sense of the term that Pufall writes, "In general, if successful, microworlds are environments in which we can think more clearly and, presumably, develop intellectually more efficiently" (Pufall 1988, p. 26). It is his opinion that much of the enthusiasm for microworlds as environments for fostering learning can be traced to Bruner's work in the 1960s, analysing mind in terms of different modes of representation. Within a microworld, modes of representation can be more closely specified, controlled and varied than in the outside world.

Logo microworlds characteristically foster the learning of procedures and principles rather than facts. They are practical environments or task domains incorporating "neat phenomena" and "powerful ideas" (Lawler 1982), so designed that available activities, while relating well to users' existing mental models, are predisposed to structure further understanding in particular directions. In the words of Papert:

> *Briefly, 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 ... the role of the microworld is to provide an intellectual environment that allows learning through interaction with it.*
>
> (Papert 1980b, p. 204)

Ideally, certain principles are so deeply embodied in every aspect of the microworld that the learner who interacts with

that world is strongly constrained to 'learn' in terms of those principles. For example Charischak, writing of Logo as a mathematical microworld, states that:

> *A mathworld is a structured environment that is designed to increase the probability of students bumping into powerful mathematical ideas. It is a place where people are engaged in mathematical activities in any topic area. The learning mode is exploration.*
>
> (Charischak 1985, p. 67)

Contrasting microworlds with CAI programs, Feurzeig writes:

> *Logo microworlds do not teach. Like real worlds they do not give away their secrets or explain themselves to passers by ... they simply exist and behave. ... Microworlds are designed as learning environments. In interactions with a Logo microworld a student strives to acquire or construct knowledge through active exploration and experiment.*
>
> (Feurzeig 1986, p. 44)

While the precise details of the necessary and sufficient conditions pertaining to the classification of a Logo program or selection of programs as a microworld have provided fuel for some debate, particularly in the early years of Logo, for the purposes of this discussion these aspects are perhaps less important than the question of the efficacy or otherwise of the term 'microworld' as a metaphor for computer based learning environments. While particular practitioners have their own very specific understandings of the word, to the less informed reader or listener it certainly

encompasses a wide range of potential metaphorical entailments, some of which may have unlooked for implications. A number of concerns in relation to this concept are shared with the more recently developed computing environments which constitute 'virtual reality' or 'cyberspace'.

In one early Logo textbook it is suggested to the learner that the room in which the computer is situated should be kept tidy so as to minimise distractions since, "You want to stay in the turtle's world" (Sharp 1984, p. 1). This suggests a level of one to one involvement with the computer more characteristic of arcade games than we generally associate with traditional Logo environments. To what extent is such a degree of engagement in fact suggested by the term 'microworld'? A 'world' as generally understood is an environment sufficient in itself. 'Microworld' carries with it the idea of diminution, but the level at which this can be assumed to operate is not clear. Is a microworld smaller than a 'world', but still self-sufficient as an environment? Is it to be understood as a complete model of the real world, but in miniature, or does 'micro' imply a reduction in the number of constituent elements? Is it a world constituted by a limited number of objects, relationships and characteristics? From a constructivist point of view, it could be argued that any individual at a given point in time inhabits a 'microworld', that is, a 'world' selected and constituted from the total available stimuli in accordance with his or her own particular stage of development, needs and metaphors. These are only some of the ways in which the term can be understood (or misunderstood) without recourse to an examination of particular examples of microworlds.

While microworlds are traditionally an important feature of Logo, the idea of "... constructing artificial realities that intersect enough with students' ideas that they can immedi-

ately begin to manipulate them, but whose 'deep structure', if you like, leads inevitably beyond those initial perceptions and conceptions" (diSessa 1988, p. 62), clearly has broader application in regard to learning. While not denying the general advantages of children being deeply and prolongedly engaged with learning activities, the degree to which not only children but adults accept computer simulations of many types as being at least equivalent in value if not in some cases preferable to 'raw' experience raises concerns regarding the characterisation of any educational computing environment as a 'world'. In fields ranging from entertainment to the dealings of multi-national corporations, there is ready acceptance of the benefits of simulation over the 'real thing'. (Benedikt 1991; Gelernter 1991; Rheingold 1992; Shapiro & McDonald 1992; Sherman & Judkins 1992). Simulations of varying degrees of verisimilitude are currently employed in a range of situations where exposure to the 'real' might be considered too dangerous or too costly, or where the original events are inaccessible because of constraints of time, distance or size. Simulations provided by both the visual and the performing arts have valid and important roles to play in most cultures. While a range of practical considerations support the usefulness of simulations in many contexts, the enthusiasm with which certain groups and individuals embrace the concept of substitute realities suggests the existence of motivations which go beyond the purely utilitarian. The embracing of these possibilities is very much in keeping with postmodern thinking in regard to acceptance of the intrinsic 'reality' of simulations and copies, as opposed to granting them simply the status of representations. Further, recent work by Gelernter suggests that human beings generally have a strong intellectual and emotional fascination with microcosms, which he relates to qualities such as the intensity of tightly

focussed and encapsulated experience, and also to the will to dominate (Gelerntner 1991, p. 181). The characterisation of a computer based environment as a 'world', certainly carries with it a significant suggestion of empowerment.

The relationship of this form of empowerment over an essentially symbolic environment to that which may be achieved in the real world is important in considering the role of simulations in many contexts, including education. While consistent with the sense of power and control over the computer which is an important element in the Logo philosophy, in relation to Logo microworlds it is doubtful whether this particular extension of the metaphor was deliberately intended. While Setzer alludes to the "spurious" sensation of power achieved through programming "... in a environment which is totally alienated from reality" (Setzer 1989, p. 32), this would depend at least in part on the place of the programming activity within the classroom situation. Other aspects which are less than trivial to a child, such as the necessity for the correct spelling of commands, may actually be seen as diminishing the sense of user's power in the interests of conformity to the requirements of the computing environment.

While the arguments of the proponents of microworlds concerning the pedagogical effectiveness of selecting and isolating certain carefully chosen elements of reality in the interests of concept formation can be appreciated, the relationship of this type of learning to the construction of knowledge within a 'whole world' social, cultural, intellectual and practical context must also be questioned. The Dreyfus brothers, for instance, state quite uncompromisingly that, "Microworlds are not worlds, but isolated meaningless domains, and it has gradually become clear that

there is no way they could be combined and extended to arrive at the world of everyday life" (Dreyfus & Dreyfus 1988, p. 32). At issue for them is the extent to which microworlds, isolated from a 'whole world' context, can properly be described as embodying 'meaning'.

A number of specific concerns relate to the practical aspects of limited domains of experience. For example, one of the features of both microworlds and virtual reality is the diminution of danger, whether intellectual or physical. While this characteristic is purported to encourage intellectual 'risk taking' in students as in other users, the degree to which the concept of risk itself is cheapened within this context is of some concern. Risk, whether to one's physical, intellectual or emotional well-being, is an ever-present element of 'whole world' existence, and is closely bound to our choices of values and behaviour. As human beings in the 'real' world, our minds, bodies and emotions are inextricably entwined. Along with an awareness of risk, for instance, comes strong commitment to ideas and actions. The separating of the intellectual component of experience from other aspects, and the distancing of the individual from the more complex consequences of action, may be seen as not necessarily being in the student's best interests. While it is often suggested that micro worlds provide a 'safe' environment in which children can explore and experiment, this very aspect removes an important element from the process of learning and developing within the wider social context.

In an interesting discussion of microworlds in the context of constructivism, Pufall suggests a justification of the term 'world' in that "A microworld, as the real world, embodies principles rather than didactically presenting them" (Pufall 1988, p.15). This observation clearly aligns the type of

learning expected within the microworld context with those 'natural', Piagetian, constructivist forms of learning favoured by Papert and his colleagues, perceived in many instances to occur more frequently outside the classroom than within it.

In the same volume, Forman and Pufall define microworlds (and macroworlds) as 'constructive' when they:

> *...provide new representational systems within which we can more effectively 'think about' what we know and what we have yet to know ... these representational systems are constructive in that they provide practical and explicit procedures for solving problems with the microworlds.*
>
> (Forman & Pufall 1988, p. 249)

In this sense one might relate the microworlds of Logo to the broader concept of problem 'spaces' as encountered particularly in mathematical thinking, an important aspect of these being the degree to which selection of the parameters of the space concerned structures both the process of problem solution and the terms within which the solution itself can be conceived.

From another point of view, Harel and Papert (1991, p. 72) relate microworlds to the concept of 'situated' learning, as expounded by Suchman (1987), suggesting that they encompass within their structures both a context and 'relationships' which contribute to the learning process within such environments. They go on to suggest, however, that their current chosen emphasis is more particularly on "...the situating of knowledge in internalized, mental environments" (Harel & Papert 1991, p. 72).

The assumption that microworlds provide more effective contexts than the real world for the construction of knowledge in specific domains depends for its utility on an understanding that such knowledge is readily transferable to an appropriate degree and in a functional form to other contexts. This in turn presumes the ability of the learner to distinguish between those aspects of the microworld metaphor which apply in the world outside, and those which do not. As Forman and Pufall put it, "Constructive microworlds are manifestations of conceptual realities and not simulations of specific realities. As a consequence, general adaptations are achieved when children understand that microworlds are not to be treated as practical realities" (Forman & Pufall 1988, p. 249). In the case of Logo, for instance, we expect the learner to have acquired some useable understanding of geometric principles, rather than to deny the relationship of force and energy to movement in general. In fact it might be argued that an element of recognised 'unreality' within a microworld could be of benefit, in that the learner could build on intuitive knowledge gained from real-world experience while not being fully bound by it, thus potentially facilitating more flexible modes of thinking.

While writers such as Forman suggest that too great an emphasis on computer based 'worlds' in education might "truncate the social construction of knowledge" (Forman 1988, p. 246), it may be argued that, on the contrary, the role of the computer as both a symbol and an embodiment of many of the attitudes and values prevalent in contemporary Western culture, suggests that computer mediated activities and interactions are very much a part of our 'dialogue' with society. In this sense, computing environments could be seen as particularly legitimate sites for learning.

The spatial dimension implied by the term 'microworld' suggests important links with developing concepts of cyberspace and virtual reality environments, detailed discussion of which is beyond the scope of this paper. The word 'cyberspace', originally coined by the novelist William Gibson (1986), has become accepted terminology for the information 'spaces' created and mediated by computing technology, including 'virtual reality' applications. While it can be argued that cyberspace already exists in the form of global networks within which computer users can 'move' and interact at will, these are generally felt to be only the precursors of computer generated contexts in which representation will occur in a greatly increased range of forms, in some respects closely related to, but in others widely divergent from, 'reality' as we normally understand it. Within such environments as conceived both in the imagination of theorists and the actual achievements of researchers, the emphasis is on constructing a simulation of multi-dimensional space within which information of all sorts takes on quasi-physical forms. While in the most obvious sense cyberspace is a purely abstract environment, paradoxically the physical, at least as analogy, takes on increased importance, and in fact is regarded as one of the most important practical advantages of these environments, in that abstract information may perhaps be more easily comprehended and manipulated in some 'physical' form. From a practical point of view, such environments extend the advantages already inherent in simpler forms of simulation and modelling.

Discussion of virtual reality environments generally places as much emphasis on 'interpersonal' interactions as on interactions with abstract concepts or data, in whatever form. Traditional microworlds have been criticised as

learning environments for their deficiencies in this regard (Pufall 1988). Certainly the 'social' dimensions of the learning experience are not well represented in traditional microworld contexts. They are a poor substitute for the 'real' world in this regard, which is one reason why educational programs which stimulate interaction between students outside the confines of the program itself have always been popular.

In conclusion, conceptions of abstract spaces as sites for intellectual activity are certainly not new, and the idea that computing environments, even of traditional types, constitute 'worlds' within which knowledge can be represented, explored and constructed is clearly of potential interest to educators at all levels. The extent to which it is helpful to conceive of these environments as 'worlds' is less evident. While a number of features of the 'real' world which are thought to be relevant to the learning process can be replicated to a certain extent by computer programs, others cannot, and indeed it may well be that maintaining a distinction between the 'real' and the 'virtual' is an important aspect of the transfer of learning from computer based environments to the wider world. Such considerations are of increasing importance in the light of the proliferation of increasingly sophisticated computer based environments being designed for purposes of education and training.

References

Charischak, I. (1985), 'Exploring math worlds with Logo', in Palmgren, M. (ed.) *Pre-Proceedings Logo '85*, MIT, Cambridge, Massachusetts, pp. 67 - 68.

di Sessa, A. (1988), 'Knowledge in pieces', in Forman, G. & Pufall P.B. (eds) (1988),*Constructivism in the Computer Age*, Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Forman, G. & Pufall, P.B. (eds) (1988), 'Constructivism in the computer age: a reconstructive epilogue', in Forman, G. & Pufall, P.B. (eds) (1988)*Constructivism in the Computer Age*, Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Gibson, W. (1984; this edition 1986), *Neuromancer*, London: Grafton.

Harel, I. & Papert, S. (1991b), 'Software design as a learning environment', in Harel, I. & Papert, S. (eds) (1991), *Constructionism: Research Reports and Essays, 1985 - 1990*, Norwood, New Jersey: Ablex.

Papert, S. (1980b), 'Teaching children thinking', in Taylor, R. (ed.) (1980), *The Computer in the School: Tutor, Tool, Tutee*, New York: Teachers College Press.

Pufall, P. (1988), 'Function in Piaget's system: some notes for constructors of microworlds', in Forman, G. & Pufall P.B. (eds) (1988), *Constructivism in the Computer Age*, Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Setzer, V. (1989), *Computers in Education,* Edinburgh: Floris Books.

Sharp, P. (1984), *Turtle Steps: An Introduction to Apple Logo and Terrapin Logo*, Maryland: Brady Communications Company Inc.

Suchman, L. (1987), *Plans and Situated Actions: the problem of human machine communication*, Cambridge: Cambridge University Press.

Is There More to Logo than Piaget and Powerful Ideas?

John Oakley

Faculty of Education, Charles Sturt University

Bathurst, N.S.W. 2795, Australia

Introduction

> *"Logo ... is first and foremost a philosophy of education"*
>
> (Higginson, 1984:33)

A criticism that has recently been levelled at Logo users is that, while there has been much written about Logo in the past five years, there has been little fundamental thinking done since the initial work by the M.I.T. group up to 1980 (Leron, 1985). Additionally, Logo users have been charged with failing to apply the process of "debugging" to the Logo philosophy (Leron, 1985). Logo itself has also been accused by some researchers of failing to live up to its promise. Papert has defended Logo and its so called failure, on the grounds that many inapplicable questions about Logo are being researched in inappropriate ways (Papert, 1985). This defence would seem to be substantially justified on the grounds that Logo is an approach to education which for the most part has been inadequately implemented. Unless research on cognitive outcomes, for example, is based on the right types of questions relating to children

taught by a Papertian implementation of Logo over a sufficiently long period of time, the findings obtained must be regarded as peripheral to the key concepts embodied in Logo.

This paper carries out some "debugging" by clarifying a relatively neglected aspect of the Logo approach to education, and by putting forward an additional perspective that may assist in the deeper understanding and more successful implementation of Logo.

The Problem

Some users of Logo such as Uri Leron (1985) or Jim Howe at the University of Edinburgh would argue that the Logo package as presented by Papert ought to be separated into its component parts which can then be examined or used separately. While this approach simplifies the teaching of Logo for teachers trained and practised in traditional methods of teaching, and indeed may be more comfortable for some children who have previously been taught by such methods, it does tend to reduce Logo to the status of a computer language or yet another teaching aid, rather than a philosophy of education.

The process of learning with Logo is described by Papert as essentially Piagetian learning, that is, *"learning without being taught"* (Papert, 1980), although some writers would argue that Papert *"seems to have forgotten some of the major premises of Piaget's theories about the development of learning"* (Rousseau and Smith, 1981).

Papert goes on to assert that children are innately gifted learners who build their own intellectual structures, not by being taught, but by interaction with the environment, particularly in their pre-school years. This interaction with the environment occurs *"in a carefree spirit without explicit external rewards and punishments ... (and) with the proper emotional and intellectual support"* (Papert, 1980:42).

I want to question Papert's use and implied definition of "*learning without being taught*". What Papert might mean relates to the traditional view that society has of the dominant teaching style found in most of our primary and secondary schools. The reference to pre-school years possibly encompasses the more child centred, less directed, informal style of teaching commonly used (almost unwittingly) by parents and others with very young children, and with other children in many non-school situations. This interpretation is supported by Piaget's statement that "*students should be allowed a maximum of activity on their own ...*" (Biehler, 1974), a statement that in no way excludes the teacher. It is unfortunate that Papert has used the phrase "learning without being taught", as this might lead some readers of *Mindstorms* to assume falsely that his view of Piagetian learning equates with a laissez-faire approach to teaching, when in fact he indicates that he does not advocate "*spontaneous, free-form classrooms or simply 'leaving the children alone'*" but rather "*supporting children as they build their own intellectual structures*" (Papert, 1980); in short, a democratic child centred approach to learning. This is the style of teaching that would seem to be more appropriate in the Logo environment. And yet, in a formal school setting, it is a teaching style that is extremely difficult to implement. Herein lies a fundamental problem for successful implementation of Logo.

Throughout his writings Papert seems generally to emphasise physical and cognitive elements of the learning environment, and to make less reference to the social element that increasingly is being recognised as perhaps the most significant element of the environment.

Leron (1985a) has argued that the philosophy (Papert, 1980), the practice (Watt, 1983), and the language (Abelson, 1981) of Logo have been well covered by these and subsequent writers, but that one of the neglected areas has been the method of integrating these facets of Logo into successful implementation. However the assertion that the practice of Logo has been adequately covered in the literature is open to question. Leron's thesis is that *"the ideal of simultaneously attaining these two goals* (of Piagetian Learning and powerful ideas) *creates a fundamental tension in the practice of Logo"*, in that these goals may be to some extent mutually exclusive, since children may not be as successful as desired in their acquisition of powerful ideas if a totally spontaneous, non-directed, Piagetian approach is used (Leron, 1985). Here there is an implied (and false) assumption that non direction necessarily implies lack of teacher influence!

However, Leron does correctly identify the major educational challenge in the implementation of Logo as *"finding more active ways of helping children without sacrificing the spirit of meaningful and exploratory learning"* (Leron, 1985:28). He (and perhaps many others) is hindered to some degree by his concept of the teacher as being external to the classroom group, rather than a member of the group as envisaged by Papert. Nevertheless he does make several other important statements: "*... it is possible that the more prevalent problem with Logo will not be too little learning ... but rather too much teaching - and of the wrong sort"* (Leron, 1985:29); *"the problems ... are not unique to Logo learning. They are likely to appear whenever we make a serious attempt to satisfy the demands of both the learner and the subject matter"* (Leron, 1985:32).

These comments, together with phrases from Papert such as "(the teacher) *does not provide answers ... but rather introduces the child to a method for solving ... (problems)*" (Papert, 1980:58) and *"teacher and learner can be engaged in ... collaboration"* (Papert, 1980:115) indirectly focus on another major element which is frequently mentioned (often in passing), but never really developed by Papert or most other writers. This is the psychosocial aspect of the Logo approach to education. The neglect of this element is illustrated by McDougall's identification of three major approaches to the teaching of Logo programming (McDougall, 1985), all of which seem to concentrate on cognitive aspects of Logo. A fourth method, briefly outlined and described more fully elsewhere (Richardson, 1985), is seemingly the only method described that attempts seriously to incorporate the psychosocial aspects of Logo. Yet this element is fundamental to the successful implementation of the Logo philosophy into workable educational practice. In fact there is convincing and consistent evidence that the psychosocial environment of the classroom accounts for appreciable amounts of variance in learning outcomes (Fraser, 1981). Papert hints at the critical importance of the psychosocial element and its effect on the educational process, by the use of phrases and sentences such as

> *"process of learning becomes more active and self-directed".*
>
> (Papert, 1980:21)
>
> *"to help learners learn how to make the choice themselves".*
>
> (Papert, 1980:98)

The importance of the influence of psychosocial aspects on the child and his or her interaction within a group situation as part of the Logo process has been explicitly recognised by Watt (Tovar et al, 1985,) as *"the critical and positive element of all classroom based Logo"* and by Tovar et al (1985,) as *"an area ... that has been neglected in Logo research"*.

Increasingly other writers are focusing on the psychosocial element of Logo. Louie (1985) and Johnson and Johnson (1985) describe aspects of the psychosocial element that would seem likely to be of substantial significance in the "debugging" of the profession's approach to implementation of the Logo philosophy.

Towards a Solution

> *"Knowledge is not something one has, it is something one is"*
>
> (Northup Frye, 1984:35)

Logo is more than a computer language or a way of thinking about mathematics. It provides a way of thinking about methods of learning and teaching. It is concerned with social interaction; working as a cooperative member of a group. Logo involves giving the learner a measure of control of a social environment, in this case the classroom. Of importance, within this environment, are relationships among peers and between the teacher and individual pupils, satisfaction of basic motivational needs, self actualisation, and communication of ideas in a supportive, stimulating, and non threatening environment. Fundamentally Logo is concerned with the creation of a group situation in which the potential of individuals is more effectively tapped and more fully utilised. Success in the cognitive domain and the nature of attitudinal or affective domains is highly dependent upon the nature of pupil motivation, stimulation and

communication, which are largely a function of the organisational or group climate in which the child operates. Children need not only to be placed in an appropriate group situation with an appropriate leadership style(s), but also to be taught how to be effective group members. These are the first and some of the more difficult questions confronting the teacher who wishes to implement the Logo philosophy successfully. The desirable outcomes envisaged by Papert arise essentially from an appropriate learning climate, and hence classroom social climate, which may not fit the traditional picture of an orderly, well controlled classroom. As Schmuck and Schmuck (1979) point out, learning is a transactional process, involving the exchange of curriculum between teachers and students, and among students. They further emphasise the importance of *"building trust ... opening up channels of communication, providing opportunities for students, developing students' self-esteem, and providing opportunities for students to learn from one another ..."* (Schmuck and Schmuck, 1979:xvi). For the teacher these are essential tasks that require planning, energy, reflection, and persistence over a considerable period of time. Success is unlikely to come to the teacher who tried Logo last week and found that it didn't work!

The group management techniques referred to by Papert in a rather intuitive and anecdotal manner would seem to be the most significant stumbling blocks for Logo teachers. If there is to be success it has its beginnings in at least an exploratory study of group dynamics, a sub discipline of social psychology. Research in this area has added a solid scientific base to the philosophical contributions of Dewey and many others, and provides many answers to questions relating to pupil interest, motivation, initiative, creativity, and achievement. The learning process under Logo is very

much an application of the principles and procedures advocated by students of group dynamics.

What a child is, is very much a function of how his or her attitudes to learning are shaped by interpersonal interaction in the group situation.

> *"It seems that the factors which most influence the benefits that children get from working with Logo are largely related to the teacher. Especially important are her educational philosophy, pedagogic style and level of understanding of Logo. In those cases where the teacher's views about the purpose and preferred procedures of education are congruent with those of Papert, Logo has proven to be a very powerful tool for the development of significant learning .... there appear to be two different types of learning occurring ... social ... academic ..."*
>
> (Higginson, 1984:35)

Despite Papert's assertion that computers will restructure learning and thinking, it is likely that this will happen only indirectly. Computers, the Logo language, and its philosophy, will hopefully provide the stimulus for teachers to create the appropriate social and learning climate. It is the process, not the trappings, that will transform education.

The basic thesis of this paper is that, while there are social and academic learnings occurring, the social learning and the context in which it occurs have a profound influence on academic learning, particularly in relation to the nature of pupil communication and to the quantitative and qualitative aspects of productivity and creativity, all of which have been identified by Papert as significant aspects of the Logo

environment. It is further proposed that the teacher is the decisive element in the classroom and the most important single factor in fostering a group or classroom climate in which the desirable aspects of Logo can develop.

How does the teacher influence classroom climate? What effect does this have on children vis-a-vis the suggested educational benefits of Logo?

To answer these questions, a few of the more important findings from the field of group dynamics or organisational theory will be outlined, and their relationship to Logo shown.

Perhaps the best starting point is to examine a question that most teachers at one time or another have asked themselves: "Why do most pupils seem to lack motivation for school work while relatively few seem to be highly motivated and consistently perform at a high level?". Papert asserts that most teachers do not expect high performance from most students. Is this a well reasoned attitude on their part? The most simplistic reasons given for differing levels of performance include varying abilities or skills, different amounts or kinds of experiences, socio-economic background, and so on.

The question of what motivates children to perform is not so easily answered, and the "answers" given above really are secondary to a more important issue which is an integral part of making Logo succeed. Without going into detail, it is sufficient to say that a substantial body of research into how individuals function in groups has identified a fundamental relationship between motivation and human behaviour.

Motivation is not behaviour: it is a complex internal state made up of a complexity of wishes, desires, drives, anxieties,

frustrations, etc. that activate individuals. It is the intervening variable between human needs and behaviour. Behaviour is the individual's attempt to satisfy the needs that motivate him or her, and as such is fundamental to understanding and changing a teaching - learning environment and to the achievement of educational goals envisaged by Papert.

Several important and complementary needs - related theories of motivation have been developed; two of the best known being those of Maslow (1954) and Herzberg (1966). Maslow's theory (and its modification by Porter (1961)) is hierarchical, and suggests that when the need at each level of the hierarchy is satisfied, the next highest need will then cause an individual to strive towards its satisfaction. The following diagrams illustrate this hierarchy (Table 1).

Table 1

Self Actualization	Self Actualization
Esteem	Autonomy
Social Affiliation	Self-Esteem
Security and Safety	Affiliation
Basic Physiological Needs	Security
Hierarchy of Needs (Maslow)	**Hierarchy of Work Motivation** (Porter)

Of particular importance to the Logo teacher are the higher level needs, with self-esteem, autonomy, and self actualisation being the motivating needs envisaged by Papert for Logo using children. Children seeking satisfaction of the need for esteem would be characterised by needs to be recognised, to be respected, to have status and prestige,

with a dynamic interplay between a sense of satisfaction and self confidence and feedback from others by such forms as being asked for advice, being included in the group's activities, etc. Autonomy refers to the individual's need to participate in making decisions that affect him or her, to exert influence in controlling the work situation, to have a voice in setting job related goals, and have authority to make decisions and latitude to work independently (Owens, 1981). Self actualisation is characterised by strong inner direction, seeking self growth, being highly "motivated".

Herzberg's theory (Herzberg, 1966) complements that of Maslow in that he postulates a two factor theory of motivation that identifies motivational and maintenance factors (satisfiers and dissatisfiers). The motivational factors are similar to the higher order needs identified by Maslow and comprise factors such as achievement, challenge of the work itself, personal growth, responsibility, and recognition. Maintenance factors, roughly corresponding to Maslow's lower order needs, encompass potential sources of dissatisfaction, which can in general terms, be referred to as working conditions. An example would be the provision of computers, access time, etc. Papert's comment on the need for "more and freer access to the computer" fits into the category of maintenance factors which do not of themselves increase motivation, but can decrease it if they are perceived to be inadequate. Subsequent research strongly supports both theories of motivation outlined above. Needs based theories, while not explaining all aspects of motivation, do provide a major psychological base from which to begin to understand why Logo needs to be implemented in a certain manner.

However, there are other factors which relate to individual differences among pupils. The nature of the pupil's orientation towards the group and the task is of importance in understanding individual pupil reactions in the group situation. Gouldner's "cosmopolitans" and "locals" (Gouldner, 1958); Presthus' "upward mobiles", "indifferents", and "ambivalents" (Presthus, 1962); and Tannenbaum and Allport's work on personality types (Tannenbaum and Allport, 1956) are examples of research that provide yet another dimension that is necessary for gaining an understanding of the behaviour of children in the structured roles of the classroom, and give some insights into reasons why a minority of children will be more comfortable in a traditional classroom setting than in the democratic setting proposed by Papert.

The problem for the Logo teacher is how to apply knowledge of theories of motivation and the Logo philosophy in the classroom.

What other aspects of group dynamics can guide the teacher in this difficult task?

One of the earliest and most influential research projects on the teacher's role as leader in the classroom was carried out by Kurt Lewin in the 1930's. Lewin set up three groups of infants grade children and exposed each group to a different teaching (leadership) style on the part of the teacher as she moved from group to group. The three leadership styles used were unambiguously "authoritarian", "democratic", and "laissez-faire", which equate approximately with the "traditional" teaching style unfortunately adopted by many teachers of Logo; Papert's suggested style; and the "no teaching" style that some Logo teachers interpret as being advocated by Papert.

The findings of this experiment and others are dramatic, and provide important insights into how the leadership style adopted by the teacher influences the manner in which children are motivated to satisfy their basic psychological needs in a learning situation. The following summary of pupil reactions (Table 2) is of particular importance in clarifying the potential outcomes arising from the different methods of teaching Logo, and provides a means of comparing very well substantiated research findings with the philosophical stance of Papert.

Table 2

Leadership Style	**Pupil Reactions / Outcomes**
Authoritarian	increased aggression - grumbling, "losing" things, arriving late, overt jealousy, fault finding.
	apathy - passive resistance, dull and submissive, relative lack of light-hearted behaviour, indolence, disinterest.
	productivity - quantity high when directly supervised, low otherwise; quality generally low to fair. Much time wasting.
	creativity - low.
	pupil involvement of self - low, much unhappiness.
Democratic	high level of cooperation - helpfulness towards each other, little or no evidence of aggression in its various forms.

	presence of light-hearted humour.
	productivity - quantity high regardless of presence of teacher; quality high.
	creativity - high.
	pupils highly ego involved and happy.
Laissez-faire	lack of cooperation,
	increased aggression but lacking the tension of the autocratically led group.
	productivity - quantity low under all circumstances; quality low. creativity - low.
	pupil involvement of self - low.
	boredom.

The democratic teacher leadership style which satisfies Papert's requirement for a learning environment with the proper emotional and intellectual support and which endeavours to *"help learners learn how to make a choice for themselves"* (Papert, 1980:98 quoting Galway's strategy) results in learning and social outcomes that are substantially in accord with those proposed by Papert.

Arising out of many such research findings has been the identification of two important dimensions of teacher leadership that are once again in accord with Papert's philosophy. These are the importance of the teacher's concern for academic learning achieved by group members and, simultaneously, a concern for the social development

of the group in the form of friendship, mutual trust and respect, and warmth in the relationship between the teacher and pupils. Research by Halpin (1966), Blake and Mouton (1964), Reddin (1970), and others has demonstrated that the most effective group leader scores highly in both dimensions and uses a democratic leadership style.

Not surprisingly, a study of group dynamics encompasses many more factors than those implied by Papert, and it is not the intention of this paper to canvass those. However, one important aspect of group life that is not touched upon in *Mindstorms* is the range of roles adopted by group members in order to achieve the goals of the group and of its members. These fall into three major categories related to task, social aspects, and individual roles. Of particular interest in the Logo context are the task-related roles which have been identified by Benne and Sheats (1948) as the initiator-contributor, the information seeker, the information giver, the opinion seeker, the opinion giver, the elaborator, the coordinator, the orientor, the evaluator - critic, the energiser, the procedural technician, and recorder. Individual group members will assume one or more of these roles, and roles will often be shared. For the Logo teacher the importance of roles lies in the observation, perhaps even accusation, that some pupils appear to become more fully involved than others where involvement is measured in terms of visibility, verbal contribution, etc. In a cohesive group it is naive to assume that all pupils should play dominant roles. We all know of pupils whose role seems to be as keyboard operator, and others who are content to offer advice. ask questions or, seemingly to watch. The danger is to assume that those who tend to watch are not fully involved or contributing in a meaningful way to group life. It should be remembered that it is the sum total of group roles

that facilitates group and hence individual achievement. Although individuals can function effectively as individuals, the psychological and academic stimulus of a democratic group enhances individual performance. In contrast, authoritarian and laissez-faire leadership styles can have detrimental effects on individual performance.

Successful implementation of Logo requires in the first instance the creation of a democratic learning environment crucial to the development of favourable attitudes towards learning, to the fostering of "*richer and deeper interactions*", to "*more articulate, effective, and honest teaching relationships*" (Papert, 1980). It is in this type of environment that the cognitive aspects of Logo can be better nurtured and developed.

It is the creation of an appropriate social environment that is fraught with difficulties for most teachers, since it frequently requires a significant change in teaching style. Are all teachers capable of making that change? Research on personality structures would suggest not.

Yes, there is more to Logo than computers, Piagetian learning, and powerful ideas. Papert's vision of how Logo should be taught encompasses many of the best ideas we have yet on how children learn. This paper has attempted to focus more closely on the question of why children learn or want to learn.

References

Abelson, H. (1981). "Logo for the Apple 11". *BYTE*. McGraw Hill.

Blake, R.R. and Mouton, J.S. (1964). *The Managerial Grid*. Houston: Texan Gulf Publishing .

Benne, K.D. and Sheats (1948). "Functional Roles of Group Members". *Journal of Social Issues*. 4, 2.

Fraser, B.J. (1981). "Australian Research on Classroom Environment: State of the Art. *The Australian Journal of Education* 25, 3.

Frye, N. (1967). "The Instruments of Mental Production". In Booth, W.C. (Ed.). (1967). *The Knowledge Most Worth Having*. Chicago: University of Chicago. cited by Higginson, W. (1984). "About That Rose Garden: Remarks on Logo, Learning, Children and Schools. In Sorkin, R.J. (Ed.). (1984). *Logo 84 Pre-Proceedings*. Massachusetts: M.I.T.

Halpin, A.W. (1966). *Theory and Research in Administration*. New York: Macmillan.

Higginson, W. (1984). "About That Rose Garden: Remarks on Logo, Learning, Children and Schools". In Sorkin, R.J. (Ed.). (1984). *Logo 84 Pre-Proceedings*. Massachusetts: M.I.T.

Johnson, D.W. and Johnson, R.T. (1985). "Co-operative Learning: One Key to Computer Assisted Learning". *The Computing Teacher*. (Oct 1985).

Leron, U. (1985a). "Logo Today: Vision and Reality". *The Computing Teacher*. (February 1985).

Leron, U. (1985b). "Some Thoughts on Logo 85". In Palmgren, M. (Ed.). (1985), *Logo 85 Theoretical Papers*. Massachusetts: M.I.T.

Louie, S. (1985). "Locus of Control Among Computer-Using School Children". In Palmgren, M. (Ed.).

Logo85 Pre-Proceedings, Massachusetts: M.I.T.

McDougall, A. (1985). "Approaches to Teaching Logo Programming", in Duncan, K. and Harris, D. (Eds), (1985). *Computers in Education*. North Holland: Elsevier Science.

Owens, R.G. (1981). *Organizational Behavior in Schools*. Englewood Cliffs: Prentice-Hall.

Papert, S. (1985), "Computer Criticism Vs Technocentric Thinking". In Palmgren, M. (Ed.), *Logo 85 Theoretical Papers*. Massachusetts: M.I.T.

Papert, S. (1980). *Mindstorms*. Sussex: Harvester Press.

Piaget, (1970) quoted by Biehler, R.F. (1974). *Psychology Applied to Teaching*. Boston: Houghton Mifflin.

Reddin, W.J. (1970). *Managerial Effectiveness*. New York: McGraw Hill.

Richardson, J. (1985). *Introducing Logo to a Year One Curriculum: An Investigation*. M.Ed.Studies project. Melbourne: Monash University.

Rousseau, J.F. and Smith, S.M. (1981). "Whither Goes the Turtle?". *Microcomputing*, (September 1981).

Schmuck, R.A. and Schmuck, A. (1979). *Group Processes in the Classroom*, Dubuque: Wm. C. Brown.

Tovar, M. and Vasquez-Abad, J. (1985). "Some Aspects of Children's Social Interaction in Small Group Problem-Solving With Logo". In Palmgren, M. (Ed.). (1985). *Logo85 Pre-Proceedings*. Massachusetts: M.I.T.

Watt, D. cited by Tovar et al (1985). "Some Aspects of Children's Social Interaction in Small Group Problem-Solving With Logo". In Palmgren, M. (Ed.). (1985). *Logo85 Pre-Proceedings*. Massachusetts: M.I.T.

Teachers: Jewels in the Crown of Learning with Logo

Tony Jones

La Trobe University,
Melbourne, Australia.

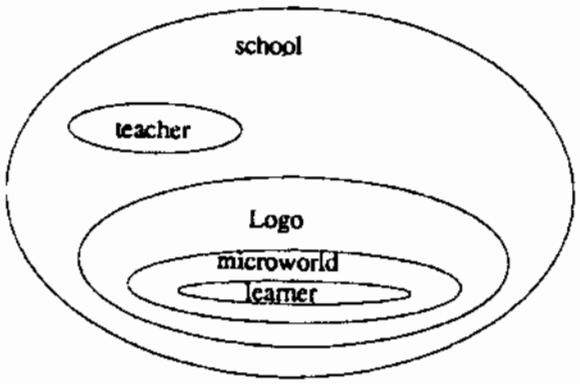
A perusal of the literature suggests that ever since it first emerged, Logo has been primarily concerned with learners rather than teachers. That is not to imply that teachers and teaching have never appeared. They often make appearances, but usually as supporting cast to the stars of Logo - the children, and occasionally adults, who are constructing mathematical and other knowledge. However the thesis for this paper is that the teacher is necessarily the centrepiece of effective learning.

Co-incidentally with the publication of *Mindstorms* (Papert, 1980), affordable microcomputers appeared in the market place and Logo became widely available. Implementing student use of computers in the school curriculum was new to teachers and there was no extensively documented and widely known research literature available. In the early 1980's, *Mindstorms* was the only Logo publication readily available to teachers, and it was read and re-read, interpreted and misinterpreted throughout the Western world. From their reading of *Mindstorms,* many teachers instituted a model that centred on the child as a learner and included

the teacher as a subsidiary part of the learning process. Figure 1. suggests a relationship many educators believe should exist between children, Logo and the teacher.

Nearly a decade prior to 'Mindstorms' Papert discussed problems with school mathematics and proposed a range of turtle geometry activities as examples to suggest meanings for 'doing' mathematics, 'knowing' mathematics, and 'understanding' mathematics. Papert (1971) argued that it is possible for children to do creative mathematics (that is to say: to do mathematics) at all stages of their scholastic lives (p 2) providing the traditional view of what constitutes mathematics is reconsidered.

Figure 1.



Although not stated explicitly, Papert certainly implied that the traditional style of school mathematics teaching must also change. Several times in the essay he links together 'creativity', 'creative mathematics' and the 'discovery method'. He poses the question:

> *In becoming a mathematician does one learn something other and more general than the specific content of particular mathematical topics? Is there such a thing as a Mathematical Way of*

> *Thinking? Can this be learned and taught? Once one has acquired it, does it then become quite easy to learn particular topics - like the ones that obsess our elitist and practical critics?*
>
> (p 3)

Activities suitable for school mathematics occur in the form of projects that characteristically involve the child over a long period of time, which results in it having a structure consisting of recognisable phases, and with these two in place the child can come to articulate aims, successes and problems.

> *Our image of teaching mathematics concentrates on teaching concepts and terminology to enable children to be articulate about the process of developing mathematical analysis.*
>
> (p 25)

Papert comments that the term 'education' evokes the perception of 'teaching' in most teachers. The methods and philosophies of Logo that were developed at the Artificial Intelligence Laboratory of M.I.T. and then expounded in *Mindstorms* (Papert, 1980) consciously concentrated on children learning rather than on children being taught. Papert (1980, p 8) notes that 'the metaphor of imitating the way the child learns to talk' was always before the M.I.T. Logo researchers. It is claimed that children learn to talk without formal instruction, and indeed at an age where they have no knowledge of formal learning or teaching.

> *We must ask why some learning takes place so early and spontaneously while some is delayed many years or does not happen at all without deliberately imposed formal instruction.*
>
> (p 7)

Because of this emphasis throughout the Logo literature on how and what children learn, it is necessary to deduce the role of the teacher. General descriptions such as 'facilitator' are often given, but usually with little accompanying substance or detail as to how the teacher can facilitate children's learning. Teaching Logo involves the teacher playing a range of different roles in the classroom. Consider for example the descriptions put forward by Watt (1986) in a chapter titled "The teacher's role".

At various times the teacher needs to be:

appreciator / acknowledger
model learner
observer
documenter
collaborator
question poser

(adapted from Watt, 1986)

The major problem with such vague and all encompassing job descriptions for teachers is that it is so easy to argue that such roles are applicable to all aspects of classroom teaching. This implies that teaching and learning Logo is no different from teaching and learning any other subject in the school curriculum. While it would be neither educationally sound nor desirable to provide teachers with a minutely detailed prescription for teaching Logo, some specific information is essential for most classroom teachers. Providing detailed guidelines and suggestions might prevent the wasteful 're-inventing the wheel' syndrome that has been characteristic of much school use of Logo. While there has been considerable communication between researchers, (for example through conferences such as Logo 84/85,

Logo and Mathematics Education and journals like *Logo Exchange),* little has permeated through to the majority of classroom teachers. Because most teachers discuss classroom Logo with few of their peers and characteristically rely on one or two Logo books for ideas and student activities, it is impossible for them not to have to repeat the learning processes and mistakes of previous generations of Logo using teachers.

Papert (1980) ascribes several roles to teachers, but makes it clear that the roles of learner and teacher are interchangeable and that there is no clear distinction between the two. He makes special reference to the teacher (or educator) as anthropologist (pp 32,181). In this role the teacher must be aware of the culture surrounding the children, and select relevant trends and materials from the culture to assist in the learning process. Papert argues that mathematics is an everyday activity and when learned in a Logo environment can be part of the culture of the child. In a Logo environment teacher intervention is restrained and focused.

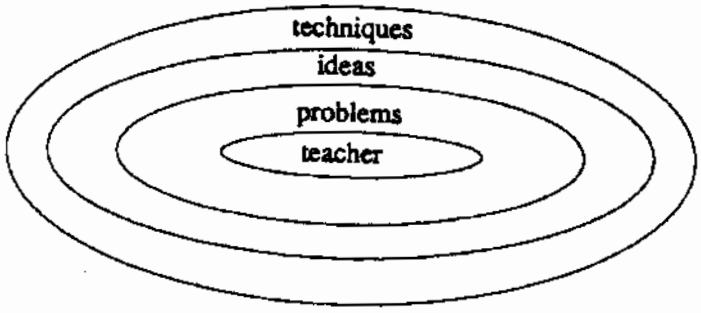
> *The Logo teacher will answer questions, provide help if asked, and sometimes sit down next to a student and say: "Let me show you something. What is shown is not dictated by a set syllabus.*
>
> (Papert, 1980, p 179).

The previous paragraphs reveal that research emanating from the Artificial Intelligence Laboratory at M.I.T. did not regard the teacher as peripheral to children's learning in a Logo environment. However the issue was not taken up in any real detail. A more traditional model, and one that has been almost universal to learning at various times in our history is shown in figure 2.

In this model it is both the responsibility and prerogative of the teacher to specify the curriculum through problems presented to learners. Over time, as students work at solving problems, they will be led into contact with certain ideas about mathematics. In turn these ideas give rise to mathematical techniques. These are more generalisable and can be used to help solve other problems.

Within this model there is coherence, with the ideas students come across creating cognitive and practical bridges between the problems provided by the teacher and the techniques acquired. There is also ample opportunity for both discovery by learners and direction by the teacher. Quasi-Piagetian learning (Leron 1985) was proposed as an approach that would involve teachers more actively, particularly in the preparation of learning materials, and yet would still enable students scope for exploration.

Figure 2. Elements of a curriculum



(adapted from Taylor 1980)

Why have computers made this model more necessary and relevant now than ever before? First there are the social and cultural changes that the increased use of technology in the workplace are bringing about. Secondary school students are now informed that it is most likely that they will have five or more jobs and they will have to retrain several times

during their working life. No longer can school leavers consider that their formal education has finished once they find employment. One implication of this for mathematics teachers is that the laborious committing to memory of facts and formulas that periodically characterises school mathematics is out of phase with vocational expectations of students. Employers suggest that school leavers need to be cognitively flexible as well as being experienced problem solvers.

Within the mathematics classroom the introduction of computers is having a more functional effect on teachers, because many problems are solved on a computer using a completely different method to that used with pencil and paper.

As examples consider the following different mathematical problems. First an activity relating to some fundamental geometric properties of rectangles.

A(2,2), B(2,5), C(14,7) and D(14,2) are the vertices of a rectangle ABCD. Draw the rectangle ABCD on your own set of axes. a) Write down the sides which are equal in length.
b) Write down the pairs of side which are equal in length. c) What is the size of each angle of the rectangle?

(Evans, 1987, p 237)

Using pencil and paper this is a very teacher (text book) directed task that involves only low level cognitive skills. The student is not asked to generalise this result to all rectangles (it appears to be assumed that there is no need to question whether the measurements made apply more generally), no attempt is made to compare properties of a rectangle with another quadrilateral or shape, and the student is not asked to formulate a definition for a rectangle.

Compare the paucity of mathematics performed in this task with the mathematics inherent in the Logo task of develop-

ing a procedure to draw a rectangle. Leron (1987) discusses in detail how the Logo approach enables the learner to apply and control concepts about variables and constants and to eventually formalise mathematical intuitions by creating a procedural definition of rectangles. What is the Logo using teacher's equivalent to the text book question quoted earlier? It could be in many different forms, such as make a procedure that draws a rectangle, which involves the learner in Logo programming, or the emphasis could be on the learner making use of a given procedure as in:

```
Explore the procedure SHAPE1
TO SHAPE1 :NUMBER1 :NUMBER2
REPEAT 2 [ FD :NUMBER1 LT 90 FD :NUMBER2 LT
90]
END
List some mathematical properties that you note.
```

The second example is a more complex mathematical problem and will be considered in the light of solutions using pencil and paper, a spreadsheet, and Logo.

Mixture problem
Find the mixture of two kinds of coffee, one costing $2.00 a (kg) and the other $3.40 a (kg), which gives a blend costing $2.60 a (kg).

Adapted from Arganbright 1984, p 187

An algebraic approach to solving this problem is to form and then solve a pair of simultaneous linear equations in two unknowns. This is a standard mathematical technique, and similar problems for using this technique abound in Australian year 10 mathematics text books.

Let the amount of the first coffee be A kg, and the amount of the second coffee be B kg. Then A + B = 1
the total amount of coffee is 1 kg,
and 2A + 3.4B = 2.6
The cost of 1 kg of the blend is $2.60.
Solving gives A = .57 and B = .43,
so to produce 1 kg of the blend costing $2.60 requires .57 kg of the first coffee and .43 kg of the second.

Arganbright (1984) provides the following solution for use with a spreadsheet:

Mixture Problem

Spreadsheet formulae

	A	B	C	D
1		COFFEE1	COFFEE2	BLEND
2	kg	.2	=1-B2	1
3	$/kg	2.0	3.4	
4	COS T	=B2*B3	=C2*C3	=B4+C4

Screen Display

	A	B	C	D
1		COFFEE1	COFFEE2	BLEND
2	kg	.2	.8	1
3	$/kg	2.00	3.40	
4	COS T	.40	2.72	3.12

(adapted from Arganbright 1984, p 187)

Either of these approaches to solving the problem could be applied in a Logo environment. Suppose the teacher develops a procedure (microworld?) such that the learner inputs the amount of each coffee (in kg) and then the procedure

outputs the weight of coffee for the calculation and the cost. For example, if the learner inputs 1 and 1 the procedure will output '2 kg costing $5.40'.

Using either Logo or a spreadsheet to solve this problem requires a different problem solving strategy from the learner and different pedagogy from the teacher compared with what learner and teacher would do in a pencil and paper environment. If the learner is not required to develop the spreadsheet or the Logo procedure, then the strategy required is that of trial and error. In either medium the learner is simply required to experiment with various inputs until the desired output is obtained. The problem solving process becomes much more complex for learners when they are also required to create the spreadsheet or the Logo procedure. No longer will trial and error suffice as a strategy.

In the computer environments the teacher's role is little more than that of encouraging the learner to experiment and to observe patterns such as which coffee to decrease proportionally in order to decrease the cost of the blend. Apart from holding discussions with individuals, it can be difficult for the teacher to determine how the learner is progressing with a problem. However if the learner is solving the problem through simultaneous equations the teacher can intervene if algebraic or logic errors occur, and can be aware of the progress of the learner by observing what has been written.

Much of the Logo literature argues for the process of learning being of more importance than the product. (For example Papert, 1980.) However some teachers feel left out of the learning process because they set the problem and then receive little feedback from their students until the problem has been solved or abandoned. The curriculum model sug-

gested earlier in this paper has deliberately set the teacher as the focal point, and this certainly implies that the teacher is not peripheral to the learning process. The teacher becomes involved not only by observing and when appropriate intervening in the work of individuals or groups, but also by ensuring that there is interaction between students on a class level, perhaps at the start and conclusion of each lesson. Some teachers make use of diaries in which students record what they planned to do, what they actually did, and the strategies and techniques they used to solve set problems. Effective learning in a computer environment seems more dependent on the skills of the teacher than ever before.

References

Arganbright, D., (1984). "Mathematical applications of an electronic spreadsheet" in V. Hansen (ed) *Computers in Mathematics Education: 1984 yearbook.* Reston, Virginia: NCTM.

Evans, M. et al, 1987. *Mathematics for Seven.* A E Press: Melbourne.

Leron, U., (1985). "Logo today: vision and reality" in *The Computing Teacher,* 12(5), 26-32. Oregon: ICCE

Leron, U., (1987). "On the mathematical nature of turtle programming" in *Logo Exchange, 5(9),* 13-15. Oregon: ICCE

Papert, S., (1971). "Teaching children to be mathematicians vs teaching about mathematics" in AI *Laboratory Logo memo No.4,* Cambridge, Mass: MIT.

Papert, S., (1980). *Mindstorms: children, computers and powerful ideas* Brighton: Harvester Press.

Taylor, P., (1980). "On Virgil: my opening lecture to Mathematics 120" in *For the Learning of Mathematics,* 1(1), *49-52.*

Logo - A Learning Environment

Terry Malone

Wesley College

Introduction

The micro-computer is now being recognised as a valuable educational tool in the classroom. A number of primary schools are experimenting with its use at various year levels and finding that it is potentially a significant force in educational practice. More software is becoming available which allows teachers to incorporate computer activities in planning for their classes. The community is also becoming aware of the importance of young children being exposed to computer activities and of being encouraged to develop a positive attitude to computer technology. The result of such experience at an early age should assist in giving these children the confidence to work creatively within the computing environment of the future.

However, the integration of computers within the school curriculum typically depends on one or two teachers in the school who become enthusiastic. The children in their classes often make giant strides during the year, only to be left with a feeling of frustration in subsequent years through lack of continuity.

The aim of this paper is to recount the experience of implementing Logo in a sequential way throughout a school, and

in doing so to offer guidelines in using the microcomputer as a teaching tool.

What Is Logo?

Logo is a computer language developed by Seymour Papert and associates for use with children of primary school age and is described in Papert (1980). Papert had worked with Piaget in Geneva before commencing the development of Logo, and this background has resulted in the language being in sympathy with the Piagetian model of learning. In fact, it is the concrete nature of this language which is appealing for use with primary age children.

Logo is a highly structured language, requiring a powerful computer. However, it is very easy to use as the commands are based on the normal language used by children. For example, if you require the computer to draw you type in DRAW. The computer will then get ready to go forward. So you type in FORWARD 30. The turtle then trundles off in a forward direction for 30 steps. Now you might want the turtle to turn a corner and head off in a different direction. so you type in RIGHT 90. The turtle will then turn right on the spot through 90 degrees. Now you could ask the turtle to go forward again. So you type in FORWARD 30. Your instructions so far would look like this:

```
DRAW
FORWARD 30
RIGHT 90
FORWARD 30
```

This is the simplest mode of operation involving a turtle. It is possible to do equally simple things using words and sentences. The basic commands to operate the Turtle on the screen, or the Robot Turtle on the floor are listed below:

```
DRAW
CLEARSCREEN
FORWARD
BACK
RIGHT
LEFT
PENUP
PENDOWN
TOOT
```

With these commands the operator can perform an interesting variety of drawings. As the teacher and the children become confident other commands can be introduced. In addition, the operator can make up and define his or her own commands.

Mathematical Concepts

An interesting feature of Logo is the learning environment it helps to create in which children can explore in a non-threatening way, an extensive range of concepts. As Logo requires a minimal amount of computer knowledge initially, teachers and children very quickly become confident in using a computer in a way where they remain "in charge". Purposefully, our approach does not emphasise programming. The focus instead is on exploring a range of concepts which form part of the normal maths curriculum.

Learning as a Process

Through Logo, the user is exposed to traditional maths concepts from a different point of view. By focusing on the process of solving a problem, whether it is simply parking the Turtle in a garage or designing a picture, the child brings to the situation a variety of concepts in estimating, trialling and modifying his ideas in the process of problem solving. There are no wrong paths and many correct ones.

No two children need follow the same steps. However, because Logo is a highly structured and logical medium, the child gradually becomes more analytical in the way he approaches a task.

Logo and the Guidelines for Primary Maths

In an article in this publication, McDougall and Adams (1982) discuss in some detail the inter-relation of Logo and the Victorian Mathematics Curriculum published by the Curriculum Services Unit (1981). Briefly, estimation, space and problem solving are some of the concepts emphasised in these recently published guidelines, which are also highlighted in the Logo experience.

Space. Many tasks in Logo involve manoeuvring the turtle. In these activities, those concepts embodied in Location need to be utilised. The child is encouraged to perceive the situation from the Turtle's point of view. If the floor Turtle is facing the child then the Turtle's left is not the child's left, for example. In drawing a shape the child's attention is focused on the components that make up that shape. For example, to draw a square the length of the sides and the turning angle assume added importance. These components can be changed in some way resulting in variations in shape.

Estimating. Through the use of roadways and mazes the ideas of estimation and planning are utilised. To get the Turtle to reach a spot on the floor or on the screen requires estimation skills of increasing sophistication. An early activity involves getting the Turtle back to where it started after a short trip. This game requires the children to guess the number of steps the Turtle must take and graphically displays the accuracy of their guess.

Problem-solving. The Logo language is designed to put the child in charge of setting-up a problem, making choices, experimenting, trying out solutions and building on earlier experience. The logical nature of the language encourages the child to work out an appropriate strategy. Making choices in setting about solving a problem is an important part of the learning.

Getting Started

Three aspects needed to be considered in starting a programme in school:

(1) Drawing up a curriculum

(2) In-Servicing the staff

(3) Co-ordinating the use of the computers.

Drawing up a Curriculum

It was decided to provide a set of activities to be attempted at each level. As far as possible the activities were open-ended to allow for both teachers and children to explore and experiment as their levels of expertise increased. Logo naturally allows a broad scope of activity at every level and the more it is used, the more ideas for activities are discovered by the user.

As a starting point the activities were related to the concepts suggested in the Mathematics Curriculum Guides. Often one Logo activity involved experience in more than one Maths concept. The way in which the activities at each level were completed was left to the teacher. However, lateral extension was encouraged rather than vertical.

Some activities involved the whole class working together and the teacher operating the computer. Others involved groups of children working at a task and some activities

allowed for two children to pursue a task. Generally all new computer activities were introduced to the whole class and then groups used the computer throughout the day. Classes at Preparatory level used the Floor Turtle extensively and activities tended to be more teacher-centred.

Not all activities required hands on experience. Lead-up and follow-up activities were part of the programme. Where appropriate children kept a record of their work and displays of Turtle drawings were prepared by the teachers.

In-Servicing the Staff

We were fortunate in having at least one of the two teachers at each level reasonably conversant with Logo arising from a series of four workshops in Logo run for all teachers in Term One. Although the Logo conversant teachers became the resource persons for their level, each class teacher was responsible for the running of the activities. In addition one teacher was given the responsibility for demonstrating and/or initiating some of the activities where the class teacher thought it useful. This person was also the "problem-solving" for technical and co-ordination problems. The teachers were encouraged to take a computer home over the weekend and all teachers found this an invaluable means of familiarising themselves with the activities.

Finally, the fact that the teachers were using Logo in the classroom was an excellent learning experience for them. The more they did, the more confident they became.

Organising the Hardware

The Preparatory School has ten classes ranging from Preparatory to Year 4, with 3 Apple computers and 1 Floor Turtle allocated to it. To ensure use by all classes the computers were grouped in two components. Group 1 consisted

of 2 Apple II computers and Group 2 consisted of 1 Apple II to which the Turtle was attached. Each group was circulated on a rostered basis. Each Year level had access to the 2 Apples for one day a week and also to the Floor Turtle for one day a week.

These arrangements required the computers to be moved daily from class to class - a time consuming procedure. We were fortunate in having this duty allocated to maintenance staff. Furniture (low tables for young children, an area of lino flooring for the Floor Turtle, and the appropriate electrical connections) was set up in a shared area between each Year level so that both classes had access. In the morning the teacher set up the machines and "booted up" Logo and after school the computers were disconnected and moved to the next unit. Obviously this approach is a compromise. Ideally one computer per unit is needed so that it becomes a permanent part of the classroom.

Classroom Management

Optimum utilisation of the computer depends upon a group working at the computer/s while other children are doing different activities. Most primary classes allow for this flexibility which also encourages a degree of independence among the children.

Once an activity is introduced to the class, a roster is drawn up consisting of pairs or groups of children. The children then move to the computer area in turn and proceed with the task described in the initial introduction. A time is allotted (usually 15-30 minutes depending on the activity) and at the end of that time the children move from the computer area and call the next group.

It is important to keep in mind that working at the computers represents only part of the computing experience. Lead-

up and follow-up activities involving discussion and role-playing are also vital elements in the experience.

Summary

This approach to using Logo in the classroom represents one step in the process of exploring the computer's potential in the classroom. The overwhelming outcome of our experience is that the more the teacher and children use the computer, the greater the possibilities become. The initial step of getting the computer into the classroom was the most difficult.

References

Curriculum Services Unit. (1981). *Mathematics Curriculum Guide.* Victoria: Education Department of Victoria.

McDougall, A. & Adams, T. (1983). Logo and Primary Mathematics. In Blane, D. (Ed.) *The Essentials of Mathematics Education.* Parkville, Victoria: The Mathematical Association of Victoria.

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas.* Brighton, Sussex: Harvester Press.

Doodle Design Debug: Process vs Content Issues in Classroom Computing

Sandra Wills

A version of a paper originally presented to the Second Annual NSW Computers in Primary Schools Conference, Macquarie University, September 1983.

Unfortunately, current classroom computer usage tends to reflect and reinforce traditional teaching methods rather than harnessing the full power of the computer to enable students and teachers to learn in ways that were not previously possible. This paper examines three more recent examples of computer usage that go beyond drill and practice, showing a concern for the PROCESS of learning not just the CONTENT.

Student use of the computer as a tool for word processing, data base enquiry, and Logo projects is analysed with particular emphasis on the problem-solving processes of doodling, designing and especially debugging.

A quote from the British Microelectronics in Education Project: "HARDWARE without SOFTWARE is just JUNK but SOFTWARE without good TEACHING is just NOISE!"

The important ingredient for success in computer education is how we as TEACHERS integrate computing into the

curriculum. It is not our role to become programmers or hardware geniuses. Our role is to evaluate computing from an educational stance, not only with regard to the educational value of the CONTENT of the software but also with regard to the value of the PROCESS of teaching and learning the software employs.

The HOW of learning is as important as the WHAT. The NSW Education Department document on the "Aims of Primary Education" reinforced this process vs content approach when it put forward COMMUNICATING, EXPRESSING and INVESTIGATING as major aims. The introduction of computing in schools does not change these aims. It should complement them.

Table I provides one perspective on how computers can be used in schools. For the purposes of this paper I wish to compare and contrast "learning FROM computers" and "learning WITH computers".

In this introductory phase that most schools are going through, the two major uses of computers appear to be BASIC programming and simple CAI, and particularly CAI in the primary schools.

With CAI the machine is used to teach the student some topic such as mathematics, English, geography. I'm sure you have all seen examples of the CAI that represents the worst aspects of our teaching methods forcing the student to try guessing that one pre-ordained answer the teacher/machine is expecting rather than encouraging them to think, reason, and solve problems tor themselves.

CAI concerns itself with the CONTENT of a subject, transferring the teacher/author's knowledge to the student. Although CAI has many advantages such as immediate

feedback, individualised rates of learning, and perhaps automatic marking and record-keeping, the PROCESS of learning is really no different from what happens in normal classrooms. It automates what could be done with a one-to-one teacher-student ratio.

Since one-to-one is an impossible dream and since most teachers would be quickly bored if their talents were only employed in a drill approach to teaching, then CAI certainly has a place. However it is not the whole story. Just as drill under-utilises a teacher's talents, it also under-utilises a computer's power. And it under-estimates the capability of our students, relegating them to a role similar to rats in a Skinnerian maze.

Seymour Papert on CAI

CAI was perceived as modelling a good teacher (e.g. patience, feedback, drill) but then we always perceive new technology in terms of the old. For example automobiles were first known as horseless carriages.

Table 1

HOW CAN WE USE COMPUTERS IN SCHOOLS?

A. SUPPORT SCHOOL ADMINISTRATION

- library
- office
- inter-regional communication

B. SUPPORT STUDENT LEARNING

learning ABOUT computers
computer awareness
computer science
commerce and business studies

learning FROM computers
computer assisted instruction (CAI)
drill and practice
tutorial learning WITH computers
word processing
simulations
data base building and enquiry
computer aided drafting
music making
electronic blackboard
spreadsheet packages
survey analysis
logo as a tool

CAI is merely using "bright new gadgets to teach the same old stuff in thinly disguised versions of the same old way".

Jacques Hebenstreit on CAI

There is generally a lack of freedom for students in CAI programs. For example, unlike a text book, they cannot browse, skip sections at their own discretion, or correct previous answers at a later time should their understanding become clearer.

The programs are "cleverly" written by the author/teacher to give the computer a "personality", which thus disguises the important role of people in the teaching process and furthermore falsely poses the computer as all-knowing and intelligent, contrary to the goals of computer awareness.

Thirdly, if CAI is supposed to model the "good teacher" then it also REPLACES the teacher, surely not a politic use of computers in the current age of teacher surplus.

Isaac Asimov on CAI

"Any teacher who CAN be replaced by a computer SHOULD be!" If you're teaching like a computer then you deserve to be replaced by one. Teaching is more than drill and practice therefore educational computing should also be beyond drill and practice.

Doodle, Design, Debug

Beyond drill and practice is using the computer as a tool to help you do things that were not possible manually and to aid you intellectually. It is not so much the content of what you are learning but how you do it. Perhaps the computer could enable us to learn in ways that were not possible before. Perhaps it enables us to explore the processes of problem-solving, that is, those skills which may be transferable between content areas.

Using the PROCESS perspective, let's analyse three examples. I believe that each can be used to highlight to the learner three important stages of learning and problem-solving. I've chosen to call these three stages:

DOODLING DESIGNING DEBUGGING.

Word Processing

A word processor harnesses the computer's memory capacity to enable us to store the words we write and to easily modify them without rewriting all the words again. We are "released" from the mechanics and tedium of hand-writing and are free to experiment more readily than we might otherwise be inclined if faced with the boring prospect of rewriting huge chunks of otherwise unchanged text.

Experimenting is DOODLING. Free to doodle, without feeling that we are wasting precious time (and paper). Any of our "doodles" can be correctly recalled without rewrit-

ing if we decide that the doodle might perhaps form the basis of a large piece of writing.

DESIGNING. After doodling or instead of doodling, you might have a design or plan for what you want to write. Usually this takes the form of headings and sub-headings with which you intend to structure your piece of writing. With a word processor, you are free to type up this structure and then insert your writing under any of the headings in whichever order your train of thought takes you. Word processing has been described as "power typing" but it's only that to a typist; to a writer, it's "power writing". At any time in your designing, you can revert to doodling, to experiment with order of paragraphs or style, or calling up previous doodles to investigate how they fit in. Put it all down as it flows then go back to tidy it up later.

That is DEBUGGING. All DESIGNS need modification as they develop past the original specification. Experienced writers know that writing involves constant revision and improvement. This process is emphasised in Donald Grave's PROCESS WRITING MODEL, which is now feasible to put into practice with the availability of word processors. Good writing involves re-reading and thinking in order to iron out the "bugs". Word processing enables you to easily modify parts without altering the whole.

According to Graves this is the PROCESS of learning to write effectively, regardless of the CONTENT of the text. With word processing you are NOT learning writing FROM a computer, but you are aided in your writing skills development with a useful tool.

And it is the teacher's role to highlight and discuss these processes. Do not assume that giving a kid access to a word

processor will magically turn him/her into a best-selling author. The processes of writing need to be consciously understood.

Information Handling

Using the First Fleet Convict Data Base as an example of data base software that is readily available to schools, thirteen items of data about each of the 777 convicts who were on the First Fleet are stored on diskette and can be quickly accessed, collated, counted, and selected according to the request you type on the computer, if it is a suitably phrased request. For example:

SURNAME = JONES

will select in a matter of seconds all convicts with the surname of JONES. This is a lot quicker than looking it up in a book, unless of course you're lucky enough to have a book that already lists the convicts in alphabetic order of surname.

When first approaching the data base, users tend to DOODLE, that is experiment with isolated requests, just to see what will happen. As confidence and familiarity grows, a design for a project may grow from one of the doodles. For example:

CRIME = MURDER

often causes great surprise when the computer draws a blank and this may spark off a concentrated investigation of why there were no murderers and which were the worst crimes for transported convicts. The learners will progressively DEBUG the design, testing related requests of the data base until they find a satisfactory conclusion to fit their hypothesis.

The CONTENT of the data base can be anything from NSW bushrangers to scientific observations in Antarctica, or census records, but what is important are the methods the

students employ to solve problems and learn for themselves from the data that is available to them. Rather than passively reading someone else's opinions on convicts, for example, they are actively researching the topic for themselves. And the teacher's role is to guide their understanding of search strategies and problem solving.

Doodle, design, debug . . . or rephrasing Papert "teaching children to BE historians versus teaching children ABOUT history".

Logo

Logo as used in primary schools, focuses on the "turtle" and uses commands like:

```
FORWARD 60
LEFT 90
BACK 40
PEN UP
RIGHT 10
```

to make it draw on a screen (or if you're lucky enough to afford a robot-turtle, on paper).

This simple and concrete, yet powerful "microworld" enables the child to explore the content of such topics as:

- spatial concepts
- mathematics and geometry
- computer programming and computer awareness
- art and design

but if we follow the "doodle, design, debug" philosophy, it is also a vehicle for highlighting problem solving strategies.

The learner obviously begins by doodling, investigating the capabilities of the world, discovering properties of angle,

distance, and measurement.

The teacher can direct and extend this important doodling phase by setting such projects as mazes (Figure 1)

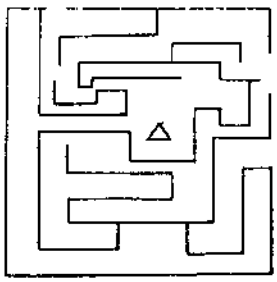


Figure 1.

or turtle golf (Figure 2)

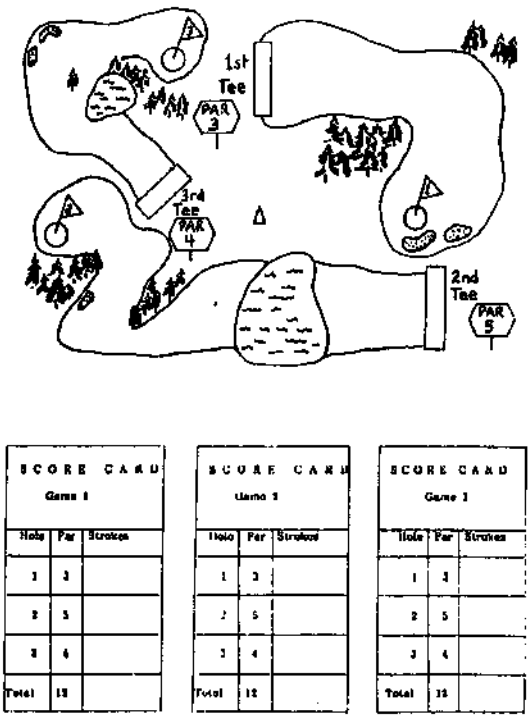


Figure 2, MECC 1983

presenting them on transparencies to fit on the screen if no robot-turtle is available.

Inevitably though the learner will want to make the turtle draw something. They will have a design in mind (Figure 3) and of course whenever there is a design there'll be a need for debugging (Figure 4).

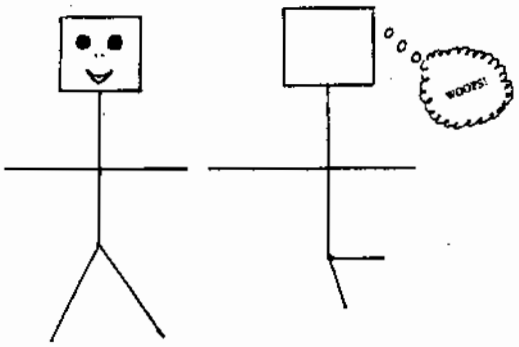


Figure 3.

Figure 4.

This will occur at every step along the way of learning with Logo for example:

DESIGN IN MIND:

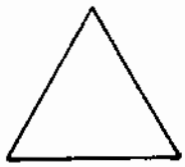


Figure 5.

PROGRESSIVE DEBUGGING:

```
REPEAT 3 [FORWARD 80 LEFT 60
REPEAT 3 [FORWARD 80 LEFT 90]
REPEAT 3 [FORWARD 80 LEFT 135]
REPEAT 3 [FORWARD 80 LEFT 120]
```

Then Doodle:
LEFT 60
REPEAT 3 [FORWARD 80 LEFT 120]
LEFT 60
REPEAT 3 [FORWARD 80 LEFT 120]
REPEAT 3 [LEFT 60 [REPEAT 3 [FORWARD 80 LEFT 120]]]

Figure 6.

A three-dimensional design!

Notice the order doodle, design, debug is not pre-ordained! Doodling can be an interesting strategy at any stage.

The Ever-Popular "House with a Bug" Example with a Difference.

If the Logo learner gets to the stage of storing procedures in the computer's memory and using these as building blocks in more complex designs, the need for designing and planning ahead carefully becomes even more crucial.

```
TO SQUARE
REPEAT 4 [FORWARD 60 LEFT 90]
END
```

```
TO TRIANGLE
REPEAT 3 [FORWARD 50 LEFT 120]
END

TO HOUSE
SQUARE
FORWARD 60
LEFT 30
TRIANGLE
END
```

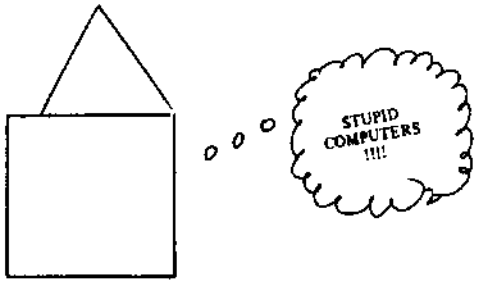


Figure 7.

But of course the best made plans . . . so into the debugging!

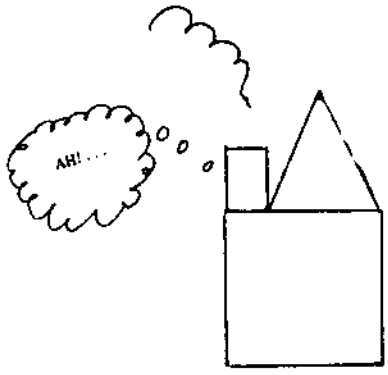


Figure 8.

Oh well, if the debugging is too hard, compromise, change the original design, DEPLAN rather than DEBUG! A cop-out? Or ingenuity?

And as for doodling with your building blocks, see Figure 9.

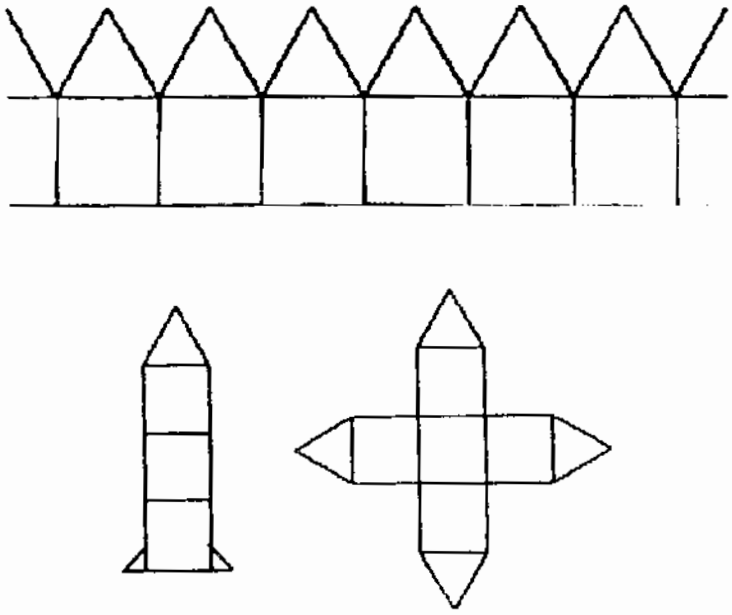


Figure 9

Logo is great for turtle geometry and drawing but it goes further than that. It is the role of the teacher to use those Logo projects not just to teach the content but to also highlight the problem solving strategies which may be generalised enough to transfer to other content areas

What If?

Of course we always have been able to DOODLE, DESIGN, DEBUG using pencil and eraser/ "liquid paper"! The best thinkers have always understood the debugging process as an aid to solving problems But the computer with its memory capacity now makes it even easier for us to think creatively and perhaps opens up new avenues for many learners who may previously have found thinking laborious instead of an exciting challenge.

It enables them to ask "what if . .?" and seek to find the answer for themselves. A computer environment is unique in that it provides the learner with the flexibility to experiment, build, modify, save, combine and generalise.

Cynthia Solomon on Learning

"A different way of looking at learning and teaching emerges, one based on the Piagetian idea that even YOUNG CHILDREN HAVE THEORIES. Thus teaching and learning are not a matter of being right or wrong but rather a process of DEBUGGING"

References

Education Department of NSW, (1977). *A Supplement to the Aims of Primary Education in NSW.*

Hebenstreit, J., (!983). "Computers in Education: the French Approach", unpublished paper presented to the WA Education Department.

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

Minnesota Educational Computing Consortium. (1983). *Apple Logo in the Classroom.*

Solomon, C. (1982). Introducing Logo to Children. *Byte.* August 1982, p208.

Teaching a Computer to Write Poetry

Anne McDougall

Faculty of Education
Monash University

Tony Adams

Department of Computing
Royal Melbourne Institute of Technology

This paper describes a computer-based environment in which children might explore and play with language. The activities outlined include sentence generation and the building of small poems by the computer, with dictionaries or word lists designed by the children. The computer programs used are included at the end of the paper, in a form ready to be typed into a computer.

The programming language used, Logo, was designed for education and is relatively easily used by people with little previous computing experience. Teachers wishing to extend the ideas presented in the paper should find the references listed at the end helpful, particularly the last chapter of the book *Learning Logo* (McDougall, Adams and Adams, 1982).

Teaching the Computer to Make Sentences

It should be said at the outset that the teaching in this activity is done by the children, with the computer as the learner. In order to plan how to teach the computer to make sentences, the children might be encouraged to make up some simple sentences themselves and consider how it is done. Will it be enough to teach the computer some words and then tell it to put them together in threes and fours followed by full stops?

The need to classify words according to their functions in a sentence should soon become clear. At this stage the children might set up for the computer lists of nouns, verbs and adjectives, for example. This is done by typing in commands such as:

```
MAKE "NOUN [DOG PLANE HOUSE CARS CHILDREN]
MAKE "VERB [RUN SMELL SING FLY SLEEP]
MAKE "ADJECTIVE [TINY FRIENDLY MISERABLE
BLUE QUICK]
MAKE "ARTICLE [THE A]
```

There are no computer restrictions on the number or types of words in the lists.

Once the lists are ready, the computer can select words from them to make sentences. For example, typing the command:

```
(PRINT FIRST :ARTICLE FIRST :ADJECTIVE
FIRST :NOUN FIRST :VERB)
```

would cause the computer to select the first word from each list and print these in the order indicated, to give the sentence:

```
THE TINY DOG RUN
```

Evidently teaching the computer to distinguish between nouns and verbs is not enough to have it produce grammatically acceptable sentences. The word lists need further refinement, and this provides a setting for the discussion of matters such as tense and number. Clearly the level and degree of formality of this discussion will depend on the level of the children and the preference of the individual teacher. The computer procedures are intended simply to provide an environment in which such discussion might take place.

The word lists might now be modified to contain only verbs in the present tense and singular nouns, for example:

```
MAKE "NOUN [DOG HOUSE PLANE CHILD CAR]
MAKE "VERB [RUNS SMELLS SINGS FLIES SLEEPS]
MAKE "ADJECTIVE [TINY FRIENDLY
MISERABLE BLUE QUICK]
MAKE "ARTICLE [THE A])
```

Now the command

```
(PRINT FIRST :ARTICLE FIRST :ADJECTIVE
FIRST :NOUN FIRST  :VERB)
```

gives

```
THE TINY DOG RUNS
```

To make more interesting sentences, computer procedures can be written (RAND, GETRANDOM, and LENGTH listed at the end of the article) to have the computer select a word at random from each list. Thus typing the command

```
(PRINT RAND :ARTICLE RAND :ADJECTIVE RAND
:NOUN RAND :VERB)
```

might produce

```
A MISERABLE CHILD SLEEPS
```

or

```
THE BLUE HOUSE SINGS
```

Both of these are grammatically acceptable sentences, but the computer is not always producing sentences which make sense. Some children might like to design lists of words so that the computer's sentences will not only be correct but also sensible; others will delight in the computer generated nonsense sentences, and design word lists to make them even more improbable.

The PRINT command above contains essentially a 'template' for a sentence. Changing this template alters the structure of the sentences generated by the computer. Thus

```
(PRINT RAND :ARTICLE RAND :ADJECTIVE
RAND :ADJECTIVE RAND :NOUN RAND :VERB)
```

commands the computer to put two adjectives in its sentence, producing perhaps

```
A TINY BLUE PLANE FLIES
```

Children might experiment with different templates to see which produce acceptable sentence structures.

Templates for Groups of Sentences

A large variety of sentence templates can be produced. The procedure WRITE (listed at the end of the article) allows templates for groups of sentences with full stops, commas, specific words not chosen by the computer, and formatting such as starting new lines. Let us have the computer make two sentences, combining some words it selects from the lists with others we specify exactly. The second sentence should start on a new line. The template for such a pair of sentences is typed into the computer with the WRITE command.

```
WRITE [ "THE ADJECTIVE NOUN VERB FS NL
"TODAY "WAS "A ADJECTIVE "DAY FS]
```

The FS stands for 'full stop' and NL for 'new line' (see the procedures at the end of the article). As a result, the computer might write.

```
THE TINY CHILD SLEEPS.
TODAY WAS A MISERABLE DAY.
```

Computer-Based Poetry Writing

The template idea might be extended to establish the structure for computer- generated poems. The aim of these activities clearly cannot be the production of beautiful poetry as such, but rather the development of a classroom environment in which discussion about poetry writing is stimulated.

The examples we will use here for templates are of Haiku-like poems; they are not strictly in Haiku form as the WRITE procedure as presented does not count syllables.

To make a template we begin with an actual poem, for example

```
LATE COOL SHOWERS FALL.
TINY BLOSSOMS OPEN AND
GREET THE NEW WARM SUN.
```

The words are now categorised into nouns, verbs and so on; any words we want retained in the poem are not categorised.

LATE	COOL	SHOWERS		FALL
adjective		noun		verb
TINY	BLOSSOMS		OPEN	AND
adjective	noun		verb	
GREET	THE	NEW	WARM	SUN
verb	article	adjective	adjective	noun

From this we can make a template to use with the WRITE procedure, so the computer can produce similar poems. Let us call the template HAIKU1.

```
MAKE "HAIKU1 ["LATE ADJECTIVE NOUN VERB FS
NL ADJECTIVE NOUN VERB "AND NL VERB ARTICLE
ADJECTIVE ADJECTIVE NOUN FS]
```

Then typing the command WRITE :HAIKU1

should result in a poem with the structure of the original one, but with words selected from the word lists. Using the word lists above, the computer might produce

```
LATE MISERABLE CHILD RUNS.
TINY DOG SLEEPS AND
RUNS A QUICK BLUE PLANE.
```

Whether the poems produced are interesting, evocative or nonsensical will depend again on the contents of the word lists from which the computer selects as it 'writes'. A number of activities can be developed using the template with different word lists. For example, some more meaning might be introduced into the computer's poetry if the types of adjectives used are restricted - say to descriptions of size.

```
MAKE "SIZEADJECTIVE [LARGE TINY ENOR-
MOUS MAN-SIZED MINUSCULE]
```

Then we replace some of the ADJECTIVEs in the template with SIZEADJECTIVEs.

```
MAKE "HAIKU2 ["LATE SIZEADJECTIVE NOUN
VERB FS NL SIZEADJECTIVE NOUN VERB
"AND NL VERB ARTICLE SIZEADJECTIVE
ADJECTIVE ADJECTIVE NOUN FS]
```

Typing

```
WRITE :HAIKU2
```

might produce

```
LATE ENORMOUS HOUSE SLEEPS.
MAN-SIZED DOG FLIES AND
SINGS A LARGE QUICK PLANE
```

Clearly, further restrictions must be made in the word lists if more sensible poems are to be written.

Children might investigate the introduction of rhyme into the computer's poems. For this lists of rhyming words must be set up.

```
MAKE "ANDVERB [PLANNED FANNED]
MAKE "ANDNOUN [HAND BAND LAND SAND]
```

Next the template must be set up with rhyming words at the ends of the lines.

```
MAKE "HAIKU3 ["LATE ADJECTIVE NOUN
ANDVERB FS NL ADJECTIVE NOUN VERB "AND
NL VERB ARTICLE ADJECTIVE ADJECTIVE
ANDNOUN FS]
```

then typing

```
WRITE :HAIKU3
```

might produce

```
LATE BLUE CAR PLANNED.
FRIENDLY DOG RUNS AND
SMELLS THE TINY BLUE SAND.
```

The problem of metre in poetry could be explored, and children might decide to use word lists containing only single-syllable or two-syllable words. The activities described could be extended using Haiku templates based on

another original poem, or templates for different kinds of poems altogether. There is no reason why the word lists must contain English words; similar activities could be carried out in foreign languages using the same computer procedures and changing only the word lists.

Exploring the relationships between different types of word lists and the computer- generated poems that result from them could provide a context for discussion of many facets of language and poetry. As was the case for sentence building, the depth and nature of such discussions will depend on the individuals involved.

The Computer Procedures

Logo is a computer language that has been designed for educational use. Papert (1980) describes the educational philosophy of Logo, while Abelson (1982) and McDougall, Adams and Adams (1982) both describe the language in detail.

The following procedures are written in Apple Logo, one of two full versions available for the Apple computer. No substantial modification will be necessary for these procedures to operate in the other Apple computer versions of Logo, or in full Logo on other microcomputers.

To set up the procedures no detailed knowledge of Logo is required. A working knowledge of the language, to the level of Chapters 1 and 2 of the McDougall, Adams and Adams book, will prove useful.

An important feature of Logo is that once procedures such as those shown below are set up, they become part of the Logo language, and may be used without understanding how they work. For example it is only necessary to know that the WRITE procedure makes use of a template and a

number of word lists to produce sentences or poems.

```
TO WRITE :TEMPLATE
IF :TEMPLATE = [ ] [PRINT [ ] STOP]
MAKE "WRD FIRST :TEMPLATE
IF FIRST :WRD = "" [(TYPE BUTFIRST :WRD "\ )]
[(TYPE RAND THING FIRST :TEMPLATE'\ )]
IF FIRST :TEMPLATE = "NL [PRINT [ ] ]
WRITE BUTFIRST :TEMPLATE
END
```

The WRITE procedure is used by the student or teacher to write sentences or poetry. WRITE makes use of a number of other procedures (shown below) to select words at random from word lists referred to in the template. These procedures also should be typed into the computer, and can be used without further understanding of their internal workings.

```
TO GETRANDOM :DICT :NUMB
IF :DICT = [ ] [OUTPUT [ ] ]
IF :NUMB = 1 [OUTPUT FIRST :DICT]
OUTPUT GETRANDOM BUTFIRST :DICT :NUMB-1
END

TO RAND :DICT
MAKE "NUMB 1 + RANDOM LENGTH :DICT 0
OUTPUT GETRANDOM :DICT :NUMB
END

TO LENGTH :DICT :COUNT
IF :DICT = [ ] [OUTPUT :COUNT ]
OUTPUT LENGTH BUTFIRST :DICT :COUNT+ 1
END

MAKE "NL []
MAKE "FS [.]
MAKE "CM [,]
```

The template can also include punctuation such as full stops (FS), commas (CM) and new lines (NL). To create new punctuation characters it is only necessary to set up the appropriate word list. For example

```
MAKE "SC [;]
```

would enable the characters SC to be used in a template to create a semicolon at that point in the poem or sentence.

References

Abelson, H. (1982) *Logo for the Apple II* . Peterborough: Byte/McGraw-Hill.

Lawler, R (1980) One Child's Learning: Introducing Writing with a Computer. *Logo Memo No. 56*, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

McDougall, A., Adams T. and Adams P., (1982) *Learning Logo on the Apple.* Melbourne: Prentice-Hall.

Papert, S. (1980) *Mindstorms; Children, Computers and Powerful Ideas* . Brighton: Harvester.

Sharples, M. (1980) A Computer Written Language Lab. *Research Paper No. 143,* Department of Artificial Intelligence, University of Edinburgh.

Sharples, M. (1981) A Computer-Based teaching Scheme for Creative Writing, in Lewis, R and Tagg, W. (eds.) (1981) *Computers in Education.* Amsterdam; North-Holland.

Editor's Note: To obtain "\" when using Apple Logo type "Ctrl Q" (on an Apple IIe).

Logo - A "Game" We Play

Jenny Betts

St Hilda's Junior School
Southport, Queensland

Introduction

Many people have written educational games in the hope that they will provide a fun way for students to learn particular concepts. Reading an article by Alan Angwin (1990), who had used Logo to write a game called "Where is Chocolate" to aid in teaching language with Preschoolers, triggered the thought that perhaps it was possible for 11 and 12 year old students to "officially" create their own games, after all, they were already doing this in their spare time. The students participating in the Queensland Sunrise Project had been using Logo for the past 12 months and had already developed quite a large bank of sophisticated programming tools which had helped in establishing individual learning environments by constructing animated databases. Creating games to convey knowledge was yet another path we could explore.

Background

Each child had access to one T1000SE Toshiba laptop computer which could be taken home each night and over the weekends. It was a common sight to see students using Logo and devoting the entire day to managing set tasks,

then taking home the computer to continue throughout the evening. Students had spent many hours using Logo during the day, which had made it possible for them to construct and reconstruct so many of the useful tools. Logo was not treated as a subject, it was simply a tool we manipulated to attain our goals. For example, if our chosen topic for discussion was Pollution, the students would use Logo to express their ideas on that subject. Presentation of ideas would be varied:

- The use of time lines to show major oil spills.
- The creation of databases about water pollution. These databases may contain animated events such as the water cycle.
- The design of a game in which the users may collect all the litter that is dropped.
- The creation of an animated story.
- The design of an animated "Choose your own Adventure Story".
- The simple task of typing the researched information into LogoWriter, printing it, then correlating the bits of paper to form a book.

Students were able to access a scanner, which aided the graphics, and a CD ROM from which they extracted relevant facts. PC Globe was also used to obtain maps of countries, flags and any other relevant facts which may have been useful to include in the work that was being completed.

Working Smart and Having Fun

Friends and colleagues are a great source for inspiration and advice. Simple advice or a simple passing remark such as

"work smarter not harder", can start the wheels of thought in motion. Upon hearing those words from a colleague, I was compelled to follow this philosophy as much as possible when working with the Sunrise Project. Even the Minister for Employment, Education and Training, Mr John Dawkins (1991), stated that the transformation of Australia to a clever country, would be a "country which works better and *smarter*". The Logo learning environment seemed apt as an aid to achieve this goal.

According to Weir's (1987) description of the Logo culture, a unique attitude towards learning to learn in a smart way is developed Being dependent on one another, a component of the Logo environment, fosters a "smart" learning environment. Working together to solve problems, working together to plan an outcome and working together to achieve a desired result are the main components of our classroom learning strategies. Students who use Logo become aware of their own strengths and weaknesses very quickly and learn to work cooperatively if they are to achieve the goals they set for themselves. Students develop a confidence to organise themselves when tackling independent tasks. A similar confidence is present when students seek help. They are not ashamed nor embarrassed to seek help from other members of the class who possess skills they lack.

Students are encouraged to be constructively critical of any ideas and tools they acquire, and if necessary, reconstruct those ideas and tools to meet the needs of their own current project. Adaptations to tools are freely shared with other members of the class. It was "everyone's responsibility to foster productivity" (Nevile 1991), and to build upon what we already knew. "Working smarter" has led to "working better" and therefore greater improvements were made.

Method

It does not matter whether the students create a database, game, report or any other genre to present work. What does matter, is the means they use to achieve the desired result. In order to provide students with an opportunity to choose a method of presentation, to provide the students with an opportunity to pursue an area of personal interest, and to provide the opportunity for me to focus on the development of skills rather than content, my learning/teaching strategy includes four phases.

1. Planning
2. Action
3. Assimilation
4. Outcome

1. Planning Phase

Formulating an open-ended task leaves the door wide open for students to negotiate ideas with me and other members of the class. Initially the whole class discusses the many ideas which could be explored on a particular subject; for example, "Resources". Ideas suggested may include mineral resources, natural resources, water resources, preserving our resources such as forests etc. Directing the students' thoughts towards particular subjects is one of the major tasks during this discussion as it enables me to elaborate on ideas that are proposed by students, and to expose them to new subject matter.

Using the discussion as a base, students research any ideas they find interesting and begin to organise the responsibilities within their group. This is known as developing "The Proposal". "The Proposal" is meant to be very flexible, and is never strictly adhered to. It is simply a way of students formally organising their ideas. It is altered each time a

new idea is implemented or an old idea rejected. The freedom given to the students does not mean that students are left unattended to discover all things that are meant to be learned as in the case of incidental learning. Students are guided carefully during all phases and it is common for me to intervene and suggest more ideas to individual groups and students.

2. Action Phase

All group members begin to carry out the designated tasks either individually or in small groups. Tasks include researching, writing the Logo code, attending skills lessons (e.g. learning about timelines, interpreting maps, note-taking, writing letters), scanning appropriate pictures to provide relevant graphics, building models etc.

Intervention is necessary during this phase especially when it is evident that a particular skill is needed to be taught. Such times may be;

- demonstrate a skill that is required in order to complete the task. For example, if a group needs to use maps, then it may be appropriate for the teacher to demonstrate how to use an atlas effectively,
- encourage the students to attack a task in a more interesting way. For example, if they are creating a game, I may encourage the group to include a time-line to depict certain concepts within the game,
- provide students with resources and skills which enable them to complete a task. Students may include a letter within the game and therefore may need to be shown how to set it out correctly,

- guide the students towards exploring particular topics. For example, time zones if they are exploring different countries,
- to set up a video, film or film strip which contains relevant information.

3. Assimilation Phase

All members of the group meet to discuss the progress made by each individual group member. At times, the teacher may be needed to help the students develop skills in correlating their work.

Both the Action and Assimilation phases are revisited often and changes to the original plan are recorded on "The Proposal".

4. Outcome

The final product is displayed and shared with all members of the class.

This kind of teaching is not new except for the fact that not all students will be exposed to the same skills. Whole class lessons do occur but they are rare. Individual students are usually invited to participate in any lessons which are organised by me. Lessons are organised when;

- an individual requests help on that idea,
- I have seen a group or person attempting to use a particular skill to complete a task,
- I have noticed that it may be useful to include a particular idea in a project.

All students are given the opportunity to participate in those invited lessons, regardless of whether it is relevant to their current task or not. If a child wants to participate, "just

because", then they are welcome to do so. There have been times when a student has chosen not to participate, but at the end of my instruction, they will approach a class member who did attend, and then obtain a personal lesson from that participant. "Smart groups" have used the idea of sending one group representative to attend the skills lesson. This representative then becomes responsible for reporting back to the whole group with the details of what occurred and teach the other group members the skill.

Games We Play

Methods chosen by the students to record the knowledge they find are numerous. One of the methods that had become popular was the design of an adventure game. Within these games, students would include many smaller ideas; time lines, surveys, quizzes, problems to solve, knowledge testers, maps, graphs, mathematical problems to solve, music etc. The list is endless and extends as far as one's imagination.

As we progressed, students developed a vast amount of code which they shared freely. Code was frequently refined, with the help of friends, so that it would work within their own project. They were seeing that trusting and relying on each other was an advantage if they were to achieve the desired end result. It was important to students that the end product was just the way they imagined it to be. Half a job was simply not good enough. They worked together to develop "smart" learning strategies which actually enabled them to produce twice the amount of work in half the time.

When students were creating games, they were still expected to meet specific criteria. For example, if the topic was resources, I insisted that they include as much factual

information as they could about the topic. They were encouraged to create a game which would allow the user to become knowledgeable about the topic when playing the game, just like the commercial software with which we are all familiar. This required the students to research relevant information so that it could be included within the game framework. Even though students were focused on pursuing information which was of interest to them, my focus still remained on the development of skills.

Bank of Tools

Over the past two years we collected and reconstructed many programming tools and we will continue to do so, as will many other Logo enriched classes. Below are only some of the tools the students found useful when constructing games.

1. Password Please?

Students found it exciting to block the path of the game player unless the secret password is known. This took place in many forms.

```
to select
 make "chances 1
 make "code which.one?
end
```

This subprocedure permits the user to have only one chance at inputting the correct password *(make "chances 1)* . The subprocedure WHICH.ONE? is run so that a password is randomly selected from a list. Of course, this does make the chances of getting the correct password very difficult and one would only get through if luck was on your side.

```
to which.one?
 op  pick [gold copper coal]
end
```

WHICH.ONE? invokes a procedure PICK which randomly selects a password from the list gold, copper and coal.

```
to pick :list
 op item 1 + (random count :list):list
end
```

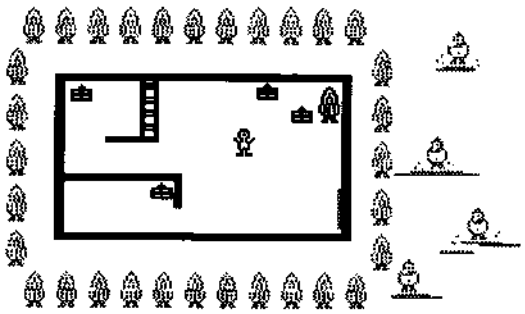
PICK randomly selects an item from a list.

```
to password
 name first rl "guess
 if :code = :guess [mineral]
 repeat 2 [pr [ ]]
 pr [INVALID CODE!]
 tone 3000 10
 if :chances = 2 [dynamite stop]
 make "chances :chances + 1
 password
end
```

PASSWORD is the superprocedure. MINERAL is a subprocedure which allows you to continue to the shop and buy items you may need throughout the game. DYNAMITE is a subprocedure which is invoked when the user types in an incorrect password.

2. Time is running out ...

Students often included a timing mechanism to challenge the user to navigate through a maze, puzzle or map as quickly as possible. Many creative timing mechanisms were designed (see Figure 1), one of which is below.



```
to timer
 cc
 make "time 200
  make "time :time - 1
 if :time = 0 [ran.out.of.time]
 (show "|Time left:-| :time)
 repeat 10 [wait 1 if key? [movement]]
 time
end
```

Figure 1. A game which is displaying the time elapsed.

Increasing the number of repeats slowed the timer. Decreasing the number of repeats quickened the timer. When a key is pressed a subprocedure MOVEMENT is invoked.

```
to movement
name readchar "??
 if :?? = "H [seth 0 fd 5]
 if :?? = "P [seth 180 fd 5]
 if :?? = "K [seth -90 fd 5]
 if :?? = "M [seth 90 fd 5]
 if colorunder = 1 [lose.points]
 if colorunder = 2 [add.points]
 if colorunder = 3 [stage2]
 time
end
```

```
to ran.out.of.time
 clear
 pr [Game Over.]
end
```

If the turtle passes over colour 1, the player loses points from the score. If the turtle passes over colour 2, the player will gain points. When the player reaches the correct destination, which is a turtle stamped in the colour 3, the "stage2" procedure will be invoked.

3. Keeping Score

Students also challenged the user by included a scoring mechanism. The procedure below enabled a program to keep score during a game.

```
to start
  make "points 0
end

to add.points
 make "points :points 0 + 1
 show ":points
end

to lose.points
 make "points :points 0 - 1
 show ":points
end
```

Below is an example of how the tool was used in a quiz.

```
to cancer
 Pr [TRUE OR FALSE. Cancer is caused
 by air pollution.]
 Pr [(T) or (F)]
 make "answer t
 make "first readlist "your.guess
```

```
 ifelse :answer = :your.guess
 [add.points carry.on] [lose.points
 block.path]
end
```

BLOCK.PATH and CARRY.ON are names of subprocedures invoked, depending on the response to the question.

4. How good were you?

At times, students who included a scoring mechanism, would also include a message that rated the user. (see Figure 2). The procedures below were created by one student who designed a game in which the user collected rubbish from polluted waters.

Setting it up

```
to setup.score
  make "litter 3
end
```

Recording the collection of the litter

```
to collect.litter
 if colorunder = 2 [ct pr [ |You have
 found the empty bottle someone threw
 in the water!!| ]
 tell 1   pu setpos [-18 -79]
 setsh 29   pe stamp tell 0
 make "litter :litter -  1]
end
```

Figure 2. The user will often receive a rating which indicates the level of performance.

How did you fare?

```
to final.score
 if :litter = char 27 [startup]
 if :litter = 3 [Pr [Sorry. You didn't
find any litter on your journey, but
at least you tried to save some of
your environment.] continue startup]
 if :litter = 2 [Pr [You only
 collected 1 piece of litter and did
 your little bit to save the
 environment in your area] continue
 startup]
 if :litter = 1 [Pr [You found 2
 pieces of rubbish and helped your
 area in collecting rubbish for a
 cleaner  and safer environment]
 continue  startup]
 if :litter = 0 [Pr [You collected
 every piece of litter on your journey
 that you could
 find!!Congratulations!!!] continue
 startup]
end
```

5. Which way do I go?

Students enjoyed designing games in which the user would use the arrow keys to direct a turtle around the screen. It was common for many students to display a map through which the user was to navigate (Figure 3).

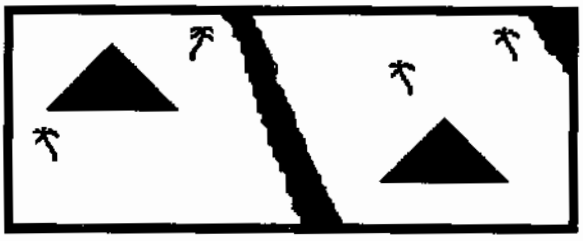


Figure 3. A map through which the user would navigate.

To enable the program to automatically move from one section of the map to another, students used the charunder primitive. A text file was created that indicated key positions on the map (Figure 4).

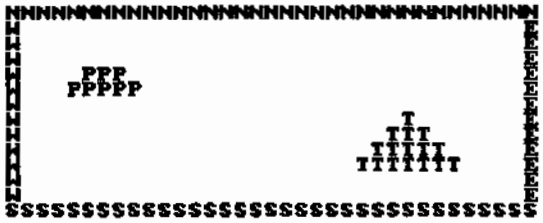


Figure 4. The file map1.txt

To ensure that the text was not seen by the user, it was loaded using the same colour as the background. The map was then loaded (Figure 5).

```
to map1
 Loadtext map1.txt
 top select bottom
 settc 0  loadpic "map1.pic
end
```

Figure 5. Combining the text and the graphic. Text will be hidden when it is loaded

The turtle is programmed to look for a character beneath it. If a character is found, a set of instructions is carried out.

```
to out1?
```

```
 tell 0
 if charunder = "n [bk 2 no.way]
 if charunder = "s[map5]
 if charunder = "e[map2]
 if charunder = "w [bk 2 no.way]
 if charunder = "p [pyramid]   if
charunder = "t [trap]
end
```

If the turtle is moved over the letter "P" of the hidden text, then a subprocedure called PYRAMID will be invoked.

If the turtle is moved over the letter "T" of the hidden text, then a subprocedure called TEMPLE will be invoked.

If the turtle is moved over the letter "W" of the hidden text, then a subprocedure called NO.WAY will be invoked. This was to prevent the user from moving off the map.

```
to no.way
  show "
  show "|There is no track|
end
```

Conclusion

Papert (1980) has stated that "the child is learning technical knowledge as a means to get a creative and personally defined end". We have used Logo as a means to an end. Students use Logo to express their thoughts. The key to its success is that the students are immersed in the Logo environment. They use Logo in all subject areas and they use Logo throughout much of the day. Of course, this can only be done if each child can access a computer for this amount of time.

It is exciting to see that the invention of laptop computers has caused the use of computers in education to advance forward in leaps and bounds. Combining the Logo environ-

ment with the portability of laptop computers has seen many changes in the way students attack their work. Students working together, willingly exchanging ideas and helping each other is one step closer to demonstrating how we can all work towards Dawkins' (1991) clever country.

References

Angwin, A (1990) Design and Development of an Adventure Game for Pre-Schoolers. in *Computers in Education*, eds McDougall A. & Dowling C., Elsevier Science Publishers B.V., North-Holland, Amsterdam, pp 623-628.

Dawkins, The Hon. John MP. Minister For Employment, Education and Training.(1991) *"A Clever Country? Australian Education and Training in Perspective"* First National Conference of the National Board of Employment, Education and Training Nov. 1990 pp. 7.

Neville, L.(1991) *When is a tool not a Tool?* (Unpublished) Australian Council for Educational Research.

Papert S. (1980) *Mindstorms - Children, Computer and Powerful Ideas* Brighton: Harvester Press).

Weir S. (1987) *Cultivating Minds - A Logo Casebook* Harper and Row: New York 1987 pp. 224.

A Tool to Think With: A New Look at Some Old Problems

Pauline N. Byrt

Kilvington. B.G.G.S.

Introduction

In this paper I will discuss some aspects of the educational potential of computers to provide a tool with which we can express and think about ideas (Abelson, 1985). In particular, I will give examples of Logo programs which can be used to explore some of the classical problems in science and mathematics. The facilities offered by Logo in this area have not been adequately appreciated by more than a handful of enthusiasts.

Logo is frequently perceived as a fine language for introducing primary students to computer management, solving problems and programming via turtle graphics — and indeed it is that. On the other hand, it is also well established that Logo can open the door to the intriguing new world of 'Turtle Geometry' (Abelson and di Sessa). However, in practice there are few teachers who feel they can afford the luxury of spending time exploring new territory when they are behind in their marking for their next class! Yet many teachers do want to do something with computers, knowing that, for their own professional development and in order to act responsibly towards their students, they

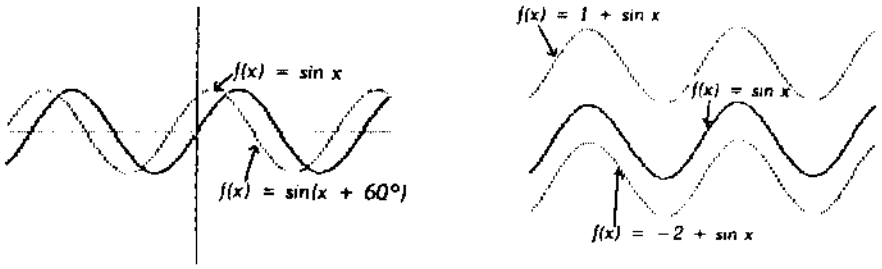
should incorporate into their teaching at least some facilities offered by computers.

Emphasis is shifting from "enthusiast-programmer" teachers and students writing their own software to a greater use of professionally written packages.

Unfortunately, apart from such open ended software as word processors, data bases, or spreadsheets, educational software is often disappointing in practice. It is just not flexible enough to meet the changing needs of different classroom situations. Computer Science could offer a service in this area, by providing facilities for students and/or staff to build up a library of appropriate Logo (or other suitable language) modules which could be used as a basis for investigations in other subject areas.

Take for example, plotting graphs. A maths teacher, having taught the essentials of graphing some straightforward functions of x, should be able to take a group of students to the computers and plot variations on the theme, all superimposed, in different colours, with the functions of x entered in their usual form:

```
y = x, then y = x + 1, y = x-2, y = x
+ 5, y = x + 10.459
y = x, y = 2x, y = 3x, y = 5x, y =
2.52x
y = x, y = 2x + 1, y = 2x- 5
y = x, y = x2, y = x3
```



Or again, the difficulty of trying to draw neat graphs of circular functions, demonstrating changes of amplitude, period and both vertical and horizontal translation are well known. The flexibility needed has been afforded to us to some extent by the use of transparencies on the overhead projector. However, most of us do not have the facilities to allow students to do this for themselves. Imagine the possibilities if students could graph for themselves on a screen, the following functions:

```
y = sin x
```

then, on the same set of axes, the graph of

```
y = sin x + 2, y = sin x - 2;
y = sin x, then y = 2 sin x,  y = -sin x, y = -
3 sin x;
y = sin x, y = sin 2x, y = sin x/2, y = sin x/3;
y = sin x, y = sin (x + 2), y = sin (2x + 2);
y = - 1 + 2 sin (x + 1).
```

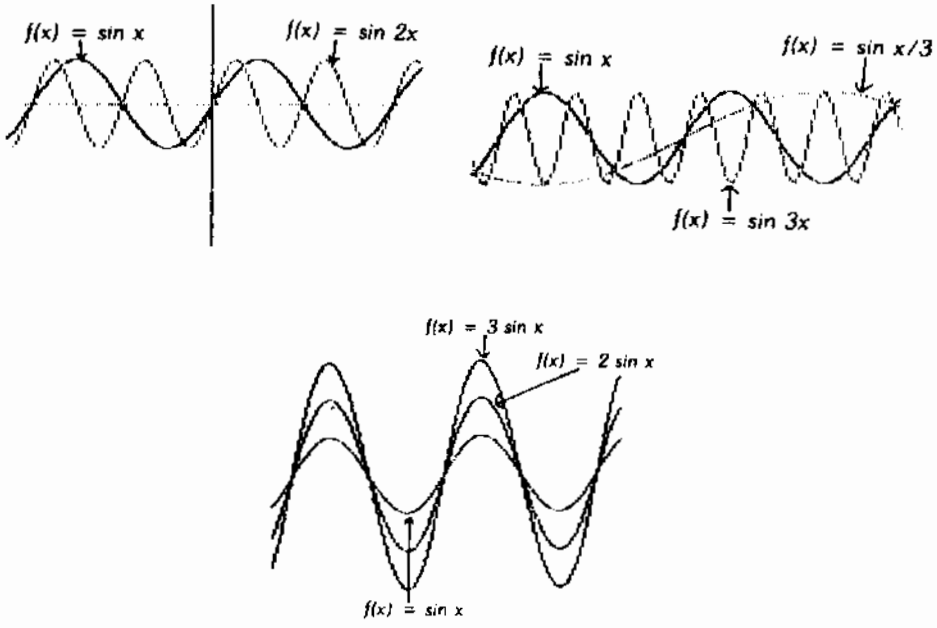
This can be easily achieved using Logo, as the program needed is simply constructed, and any variations of the functions are easily entered. For example, we begin our program with

```
to graph
type [F (x) = ]
```

When executed, the student is presented on screen with: F(x) = and may enter whatever function is needed. The rest of the program is discussed in more detail later, in the sections on coordinate geometry and circular functions.

It is a small step from here to investigate the properties of combined functions. What does y = sin x + sin x, or y = sin x + cos x look like? What about y = sin x + sin x/2, and y =

sin x + sin .9x? Properties of sound waves and 'beats' would flow naturally from such explorations.



Another difficult topic for students involves trying to visualise the relative motions of the planets in astronomy. Why not include a session using computers when the students can plot the motion of the planets around the sun (multiple sprites would be ideal, but even one turtle can do a lot)? They then could develop some feeling for relative motion by "sitting" on one moving planet and viewing the motion of the others from that perspective. The Logo procedures required for this are again brief and relatively simple, both to write and operate. The resulting motions are not easy to predict, and playing with different relative speeds and diameters helps one to understand the actual movements of those "wanderers" which inhabit our skies. Such activities certainly increase one's respect for the astronomers who formed the first coherent theories of planetary motion. One also starts to really appreciate the amount of calculation that needs to be done when planning for a space probe to make

a rendezvous with one of the planets. Creative students may even wish to use these activities as the basis for writing computer space games.

Biology includes some topics, for instance population studies, that are particularly amenable to investigation by computer. Programs that print out or plot, say, the number of rabbits on an island over a certain time period that multiply at a certain rate etc. (assuming that there were six pairs to start with), are particularly elegant using recursive Logo procedures. But the real strength of the Logo approach is most evident when you want to play the "What if we change this?" game, as it is so simple to alter the variables.

The Logo programs that follow are not intended to provide a comprehensive coverage of suitable topics, and indeed are just the tip of the iceberg. They were chosen simply because they were written in response to discussions with other colleagues and because they interested me at the time. I believe that the best way to introduce computers to either students or staff is to "*provide lots of interesting non trivial problems . . . where possible of general and social interest . . . alternating in appeal between humanities and science students*" (Bennett, 1976). Although this paper is too short to meet these goals it is presented as the basis for discussion. Most of the programs are appropriate for secondary students. The last two biology topics are not, and arose out of my curiosity to discover whether Logo would be of any help in post-graduate research.

Coordinate Geometry

Most books on Logo graphics concentrate on non-coordinate "Turtle" geometry, and if they discuss coordinate geometry at all, point out its disadvantages in the Logo context. For example, Abelson and di Sessa: "*In*

formulating turtle-geometric descriptions we have access to an entire range of procedural mechanisms (such as iteration) that are hard to capture in the traditional algebraic formalism. Moreover the procedural descriptions used in turtle geometry are readily modified in many ways" (1981:14). It is unfortunate however, that many have understood this to imply that Logo has little to offer the teacher of traditional maths. By contrast, I have found that teachers can quickly appreciate the power of Logo when they see how it can be used in topics with which they are very familiar.

Circular Functions

Investigation of factors that affect the characteristics of sine or cosine graphs can be tedious if the graphs need to be plotted by hand. By contrast, the speed with which the graphs can be drawn using the computer, combined with the ease of varying any number of factors, e.g., amplitude and period, allows the student to concentrate on the processes involved in exploration without being hampered by the mechanics of the drawing task.

It soon becomes obvious that some modification of the scales of both the x and y axes is required for circular function graphs. One way of achieving this is to modify the third module of the program in section 1 as follows:

```
to x
.setscrunch 30
output 3 * :x
end
```

The SETSCRUNCH command results in the maximum value of sin x (= 1) coming about one-third of the way up the screen. Converting x to 3 * x allows more than one complete cycle of the curve to appear on the screen.

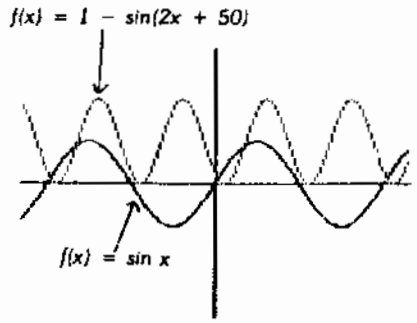
It does not require much imagination to see how these modules could be used when exploring the nature of a physical phenomenon that has some of the properties of sine waves, e.g., sound.

Summation of Sine Curves and Representation of Beats

Two sound waves with frequencies very close together produce the phenomenon known as "beats". These can be represented very simply by using an input such as fx = sin x + sin x * .8. (It is necessary to change the scale of the x-axis even more to get the full effect on screen (e.g., output 15 * x).)

Alternatively, the motion of the turtle can be made to represent the motion of a particle on a spring by fixing the x-coordinate to 0. In this case the SETPOS command in plot.point becomes:

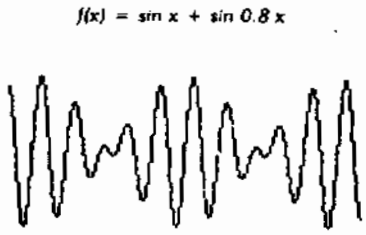
```
setpos se 0 run :fx
```



The following three brief modules, for example, allow the user to plot any function of x. (modified from Ludwig, 1985)

```
to graph
type [F (x) = ]
make "fx readlist
penup
```

```
plot.point-139 :fx
end
to plot.point :x :fx
if :x > 140 [stop]
setpos se :x run :fx
pendown
plot.point :x + 4 :fx
end
to x
output :x
end
```



When the first module (graph) is run, the user is invited to key in a function of x, e.g. 2*x, x*x + 6*x + 3, 1/x, etc. Note that there is no need to use the usual Logo form of a variable with the dots or colon as the third module (called x) automatically converts 'x' to ':x'. The MAKE statement stores the function of x as a list with the variable name fx.

The second module (plot.point) is then invoked Two variables are passed from "graph" to this module, namely x, which is set to -139 (the left hand edge of the screen), and the list fx.

Plot.point is a recursive module that plots the graph. On the first iteration SETPOS positions the turtle at the left of the screen. It is convenient to use an odd (e.g. 139) value of the x coordinate for the commencement of the graph to avoid 'division by zero' errors. The y coordinate needs to be

calculated for each value of x and is obtained from the RUN fx operation. The procedure is reiterated with the value of x increased to x + 4, and so on. The process is halted when the x value is greater than 140.

Extra modules to draw axes, indicate scale, etc. could easily be added.

The modules are potentially very useful for students familiar with the process of drawing graphs who wish to work with more complex functions of x, for example sin x, cos x, etc.

A related set of procedures can be used for polar coordinate graphs. For a much fuller treatment of this topic refer to Ludwig (1985).

Another Way of Looking at Sine Curves —Using the Gradient

Another way of plotting sine curves is based on the fact that the gradient of a sine curve at any point is cos x. In such a procedure the turtle moves forward in small steps along the gradient of the curve. (Ninety degrees is added to the heading of the turtle to make the curve run horizontally.)

```
to sin.curve :angle :distance :increment
setheading 90 + arctan cos :angle
fd :distance
sincurve :angle + :increment :distance
:increment
end
```

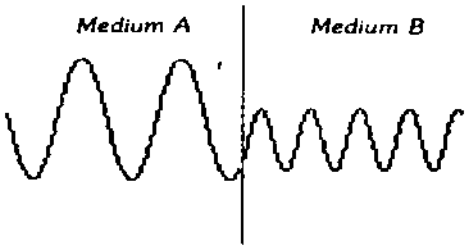
This method has some advantages. One can, for example, play with the forward distance that the turtle moves. Reducing the turtle step size by one percent each iteration produces a damped curve resembling the motion of a particle of a spring slowed down by friction:

```
to damped.curve :angle :distance
setheading 90 + arctan cos :angle
fd :distance
sincurve :angle + 10 :distance * .99
end
```

Alternatively, one could imagine that the turtle moves at different speeds in different areas of the screen and model, say, sound waves moving in different media.

sin.curve 0 1 10

damped.curve 0 1



Circles, Ellipses And Viewing Planets From The Earth

A simple procedure for drawing an ellipse with the focus at (0,0) is:

```
to ellipse :a :b :angle
if :angle > 360 [stop]
setpos se :a * sin :angle :b * cos
:angle
ellipse :a :b :angle + 4
end
```

"Ellipse 20 30 0" produces an ellipse with maximum radius 30, minimum radius 20, and "ellipse 40 40 0" produces a

circle.

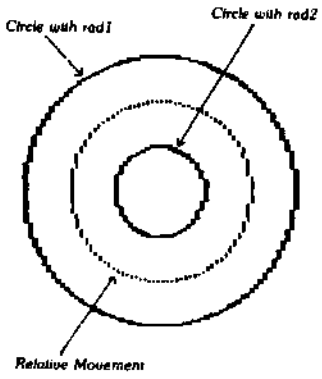
With multiple sprites we could very easily represent the motion of the planets in the solar system with the sun at the centre. We can even go back to a geocentric solar system and represent the motion of any planet as seen from the earth. For simplicity, circles rather than ellipses have been used for this exercise.

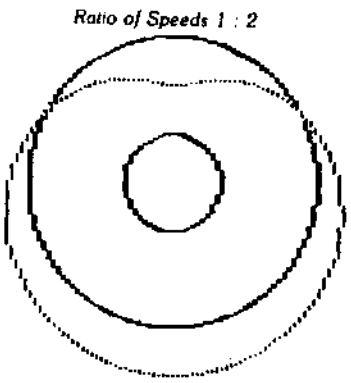
Firstly, imagine sitting on a body moving around a circle of radius 20 and viewing the motion of a body moving on a circle radius 60. The resulting position is obtained by subtracting our x— and y— coordinates from those of the other body:

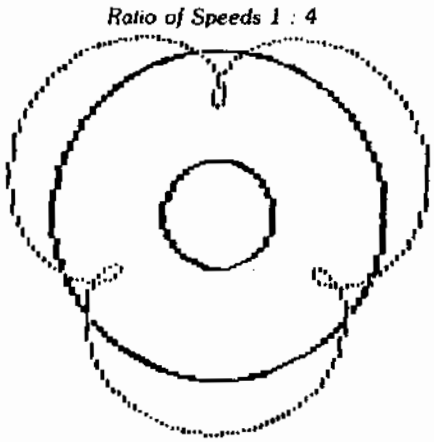
```
to relative. movement :rad 1 :rad2 :a
if :a > 360[stop]
setpos se (:rad1 * sin :a) - (:rad2 *
sin :a) (:rad1 * cos :a) (:rad2 * cos
:a)
relative.movement :rad1 :rad2 :a + 10
end
```

We are sitting on the circle with rad2. The movement of the body on the outer circle is obtained using:

```
relative.movement 20 60 0
```







It is possible to allow one body to be moving with a faster angular velocity than the other by introducing a "ratio-of-speeds" factor.

```
to rel.motion :rad1 :rad2 :ratio
penup
rel.motion1 :rad1 :rad2 :ratio 0
end
```

```
to rel.motion1 :rad1 :rad2 :ratio :a
if (and :a * :ratio > 360 :a > 360) [stop]
setpos se (:rad1 * sin :a * :ratio) - (:rad2
* sin :a) (:rad1 * cos :a * :ratio) - (:rad2
* cos :a)
pendown
rel.motion1 :rad1 :rad2 :ratio :a + 10
end
```

N.B. The ratio = the speed of the first (observed) body / the speed of the second (observer) body. (Or ratio = the time for one revolution of the second body / the time for one revolution of the first body.)

One quickly gains some interesting insights into the geometry of the sky after playing with different values of the three variables. For example,

- the diameter of the spiral motion of the observed body equals the diameter of the circle of the observer,
- the number of spirals is one less than the ratio of relative speeds.

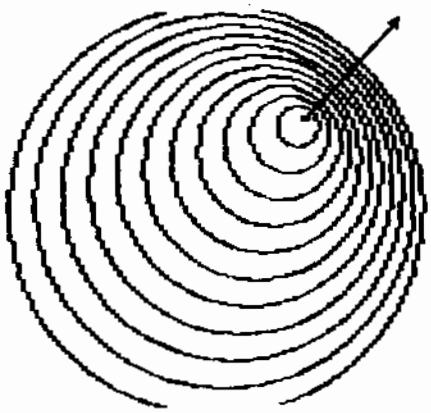
I have used these modules to plot the movement of the planets using the data in our Year X science text book with remarkably accurate results.

The Doppler Effect

Imagine a stationary source of sound. The sound waves emitted with time can be represented as circles of increasing diameter. If the body was moving, however, the sound waves as seen by an outside observer would be the result of the sum of the coordinates of the sound waves and the moving source. To achieve this effect we need two recursive modules, one to set the coordinates of the centre of the circle (the position of the sound source), the other to draw circles of increasing radii (the positions of the sound wave after fixed time intervals). Using the following modules the source of sound can move according to any function of x. The final effect is a picture of the waves at one instant in time. The program plots the largest circle (radius = 100) first as it was the one produced when the source was at the starting position.

```
to ripples :radius
type [f(x) =]
make "fx readlist
type [Distance moved by wave in unit
```

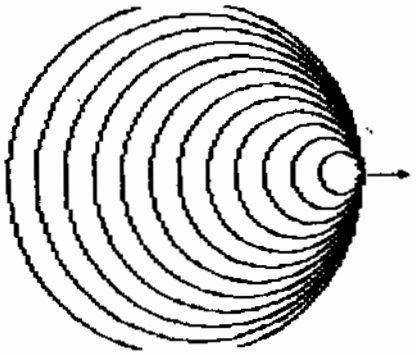
```
make "inc.rad rw
type [distance moved by source in unit time = ]
make "inc.x rw
move.source :radius 0 :fx
end
```



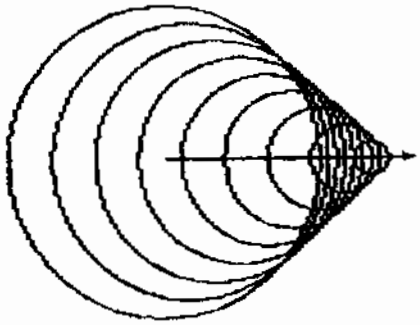
Speed of source < speed of waves
Source of waves moving along y = x

```
to move .source :radius. ms :x :fx
penup
if :radius.ms < 0 [stop]
plot.wave :radius :x :fx 0
move.source :radius.ms - :inc.rad :x +
:inc.x :fx
end
to plot.wave :radius.pw :x :fx :a
if :a > 360 [stop]
setpos se (:radius * sin :a) + :x(radius *
cos :a) + (run :fx)
pendown
plot.wave :radius.pw :x :fx :a + 10
end
```

The relative speeds of the source and the emitted waves can be varied very simply with fascinating results, and could form the basis of discussion on standing waves, what happens to the sound when one breaks the sound barrier, etc.



Speed of source = speed of waves
Source of waves moving along y = 0



Speed of source > speed of waves
Source of waves moving along y = 0

Finding Pythagorean Triads - A Non-Graphical Problem

Pythagorean triads satisfy the relation c2 = a2 + b2, a, b and c being integers.

```
to triad :a :b
if :a > 100 [make "b :b s 1 triad 1
:b]
local "c
make "c sqrt (:a * :a + :b * :b)
test (int :c) = :c
iftrue [(pr list :a :b :c)]
triad :a + 1 :b
end
```

It is an interesting exercise to let the computer run for some time. Far more tricks are generated than most students expect and can lead to some valuable discussion.

Triad 1 1 generates the following (incomplete) sets of numbers:

```
?triad 1 1      20      21      29.0
          4      3     5.0      28      21     35.0
          3      4     5.0      72      21     75.0
          8      6    10.0      72      42      5.0
         24      7    25.0      32      24     40.0
          6      8    10.0      70      24     74.0
         15      8    17.0      60      25     65.0
         40      9    41.0      36      27     45.0
         16     12    20.0      21      28     35.0
         35     12    37.0    9628  100.0
         84     13    85.0      16      30     34.0
         48     14    50.0      40      30     50.0
          8     15    17.0      72      30     78.0
         20     15    25.0      24      32     40.0
         36     15    39.0      60      32     68.0
         12     60    20.0      56      33     65.0
         30     16    34.0      12      35     37.0
         63     16    65.0      15      36     39.0
         80     18    82.0      27      36     45.0
         15     20    25.0STOPPED! IN triad
         21     20    29.0
       9920  101.0
```

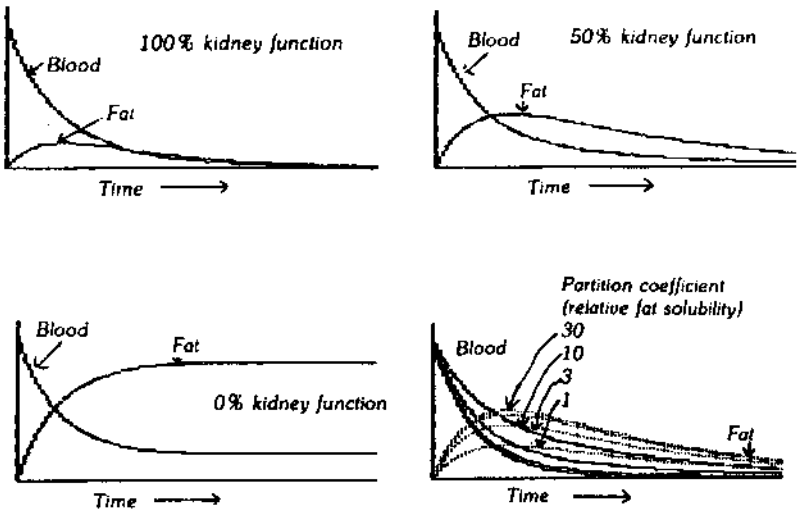
The program can be modified to find Pythagorean tetrads (the diagonal of a cube), and pentads (the diagonal of a four dimensional object?). It's a lot of fun to start up the program in the evening and return next day to see what has come up.

Anaesthetics - Modelling the Distribution of a Drug after Injection

One of the classical problems in medicine is to determine the fate of drug molecules after injection into the body. For instance, it is very important for an anaesthetist to under-

stand the path taken by an anaesthetic after injection into the bloodstream. It is known that anaesthetics, being non-polar molecules, are rapidly absorbed into the fatty tissues, and that organs such as the kidneys and liver metabolize the anaesthetic. Eventually, in one form or other, the molecules leave the body in the urine.

I have written a set of procedures that provide the basis for one simple model of this situation. The graphs of concentration of the anaesthetic in the bloodstream and in fatty tissues are quite typical of those found in experimental situations. The effects of factors related to the patient, such as obesity and kidney damage as well as drug-related factors such as the degree of fat-solubility, are very easy to investigate in relation to the persistence of the anaesthetic in the body. The procedures are available if anyone is interested.



The 'smelling Turtle'

Abelson and di Sessa (1981, chap. 2) have described in some detail procedures for modelling animal behaviour in

response to various stimuli. This is particularly interesting to me as I have spent many hours watching motile single-celled organisms, the swimming spores of fungal plant pathogens, which are attracted to plant roots. The literature contains many descriptions of the swimming patterns observed as the spores approach the source of an attractive chemical or a source of light, heat, or moisture and various terms have been coined to describe different mechanisms that are thought to be involved.

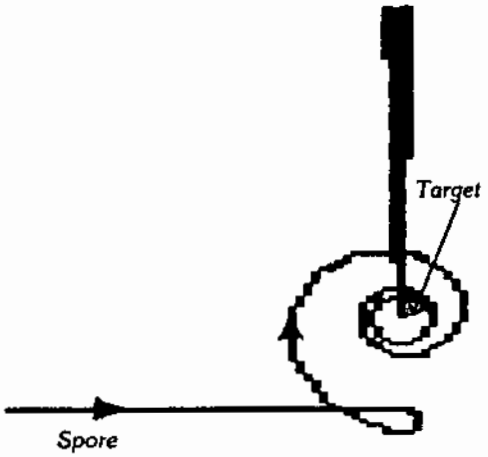
Briefly, the fungal spores swim for a certain distance on a helical path of fixed amplitude, frequency, and direction, then do a sort of 'tumble' and set off again in a new direction on their helical path. As they approach the source of an attractant the frequency of 'tumbles' increases and/or the distance between 'tumbles' decreases. Two mechanisms have been suggested:

a. Taxis, in which the spore can orient itself along a concentration gradient of the attractant;

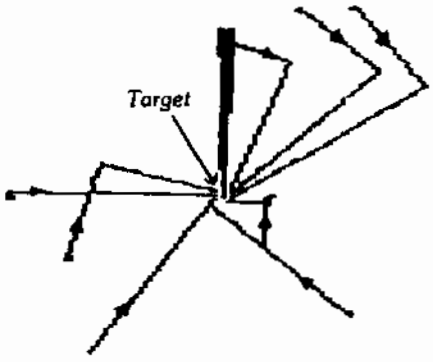
b. Kinesis, in which higher concentrations of the attractant cause some sort of change in either speed or frequency of turns, regardless of direction (Fraenkel and Gunn, 1961).

Most chemosensory responses of swimming bacteria are kineses, but with the larger single-celled organisms including fungi, the situation is less clear. Indeed, the evidence is quite contradictory (Hickman, 1970; Allen and Newhook, 1973). However, using even the comparatively simple models outlined by Abelson and di Sessa it is possible to see some possible sources of confusion. I will not go into the details. However, it quickly becomes clear that some of the suggested models are much more effective than others. In addition, it rapidly becomes obvious that by changing

only one variable, the angle of turn, in one of the models, causes the path of the "model spore" to change from an apparent taxis to an apparent kinesis. I suggest that the ease with which programs for such models can be written and tested makes Logo quite a remarkable tool for this type of research.



30° turns: "apparent kinesis"



90° turns: "apparent taxis"

Conclusion

The turtle can be programmed to exhibit a great variety of behaviours which may be tied to a greater or lesser extent to

the outside world i.e., it can exhibit local as well as global behaviour patterns. This makes Logo a most powerful tool, power being 'a way of measuring how much the language helps you to concentrate on the actual problem you wanted to solve in the first place, rather than having to worry about the constraints of the language' (Harvey, 1985). When using a computer to solve a problem or model a system, there is the possibility of continual interplay between the concrete and the abstract which makes the experience most valuable and exciting.

References

Abelson, H. (1985) Procedures as General Methods. In *LOGO in Australia: 10 Years On.*

Abelson, H. and di Sessa, A. (1981) *Turtle Geometry.* Massachusetts: MIT Press.

Allen, R. N. and Newhook, F. J. (1973) *Trans. Br. mycol. Soc., 61,* 287.

Bennett, W. R. (1974) *Scientific and Engineering Problem Solving with the Computer.* Prentice-Hall.

Fraenkel, G. and Gunn D. (1961) *The Orientation of Animals.* New York: Dover.

Harvey, B. (1985) *Computer Science Logo Style.* Massachusetts: MIT Press.

Hickman, C. (1970) *Phytopathology, 60,* 1128.

Ludwig, R. V. (1985) *Creative Computing, 11,* October, 98.

Stepping Out: Legged Robots in LEGO

Peter J. Carter

Wheels are easy, but legged locomotion opens up a whole new field of exploration. Machine walking is an object of research at a number of universities and other institutions worldwide, but the advent of LEGO/Logo makes it accessible to schools. Legged locomotion offers some advantages for machines, over rugged terrain for instance, where wheels would become bogged or trapped between rocks.

We tend to take walking for granted, it's simply something we do naturally. But watch a young child learning to walk and you realise that it is, in fact, a very complex skill, requiring balance and coordination. There are two types of balance involved, static and dynamic. Any animal or machine that moves without becoming unstable has static balance; an insect, for instance, that always has at least three legs on the ground is statically balanced. Walking humans are dynamically balanced; there are stages in our gait where we are, in effect, falling for a short period. Without accelerometers and rate gyros, any models we build will need to be statically balanced, although Pawson (1985) describes a 'walking man' model that relies on what he describes as 'quasi-static' balance. You can best describe

that to yourself by walking very slowly and noting how the motions differ from your normal gait. Jameson (1985) describes a gyroscopic biped, balanced and driven, through precession, by its gyroscope.

Not surprisingly, the walking of animals, including insects, has been studied for any information that may help the design of machines. Observing animals and then modelling their movements with LEGO has been done at the Hennigan School in Boston, and proved very fruitful. Alexander (1984) describes the gaits of quadrupeds with diagrams such as Figure 1:

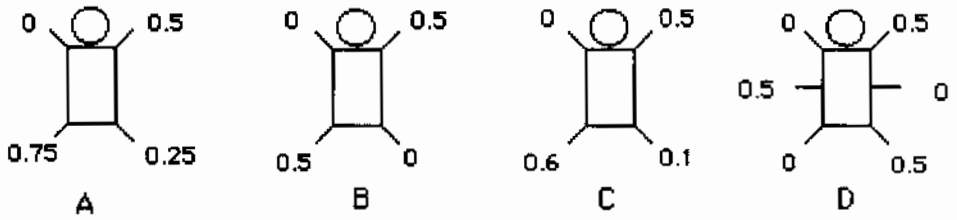


Figure 1

A Amble B Trot C An efficient Turtle gait D Alternating tripod gait for hexapods

The numbers by the feet show their relative phase, defined as the stage of the stride at which it is set down, expressed as a fraction of the duration of the stride following the setting down of the reference foot, the left front in our examples. The reference foot is assigned relative phase 0 and the others are in the range 0..1. A foot with relative phase of 0.5 will therefore touch the ground midway between two groundings of the reference foot. Among the animals studied by Alexander was the turtle (ie. tortoise), and he devotes more than two pages of text and diagrams to discussion of efficient turtle gaits.

There have been quadruped machines, with perhaps the most prominent being the General Electric Walking Truck, photographs of which appear in most robotics books, built during the 1960s. Since every movement was directly controlled by the driver without the aid of a computer it was a very awkward and tiring machine to operate. Later machines have been computer controlled, with varying degrees of success, with one of the first being the 'Phony Pony', whose legs had hip and knee joints and were controlled by electronic sequencers. A machine built by Petternella in Italy had telescopic legs and resembled a table, which is what it became at the end of the project. Hirose (1984) describes research with four legged machines capable of stair climbing and other feats.

Many recent machines have been hexapods, with Sutherland's machine designed to explore the control and use of hydraulics, and being the first to carry a driver, around a car park of Carnegie-Mellon University. Another man carrying hexapod, the Adaptive Suspension Vehicle, is being developed at the Ohio State University, its legs making use of sliding joints. Yet another, with electric drill motors, was in use at the same institution for exploring the use of sensors such as video cameras, for manoeuvring over obstacles, stair climbing and other problems. Perhaps the most spectacular machine has been one legged, the Carnegie-Mellon University hopper, a very interesting study in balance and control.

During the 1960s there was interest in legged machines for use on the Moon, although in the event, the only vehicles ever driven or dragged on the Moon (Lunokhod, MET and LRV) have been wheeled. We may see renewed interest in legged vehicles for Mars exploration, but here again the present favourite is a wheeled machine of sorts, the dumbbell shaped inflatable Mars Ball.

LEGO Machines

The simplest mechanism, with variations used in many models, uses two outer legs and a single central one, often the base of the machine. There are three gears on each side, with the two end ones connected to the 'feet'. The mechanism can be used horizontally or vertically and the machine is always statically stable. The parallel in nature is the pentapedal gait of kangaroos, and the idea is used in large excavating machines, where the base can rotate, either for swinging the jib or for changing direction as the machine moves. It would be easy to build that into a model with a second motor and suitable gearing. A not very different scheme is used in underwater robots built by Komatsu.

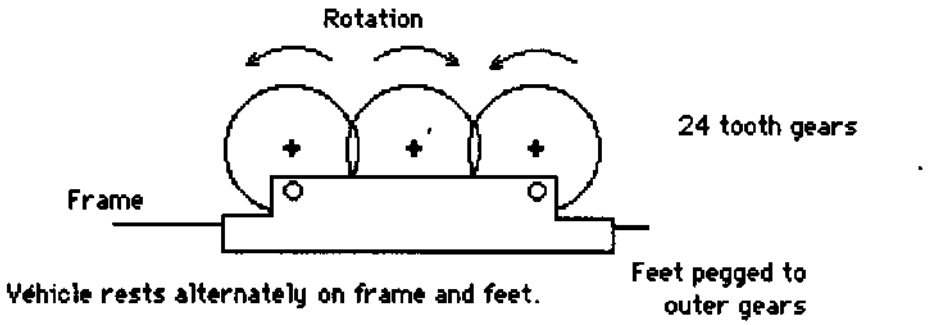


Figure 2 Basic walking mechanism

Hexapods can move their legs in threes as alternating tripods (gait D in Figure 1), and are therefore statically balanced, making them potentially the most suitable models. To quote Sutherland and Ullner (1984): 'We tried quite a few algorithms for controlling the sequence of leg recovery, but one of the simplest turned out to be the best. An alternating tripod gait seemed to work consistently well for all directions of travel.' Our first attempt is shown in Figure 3. Only one of the six legs is shown on the diagram, disconnected. Care must be taken when assembling the drive shaft that the legs are properly phased.

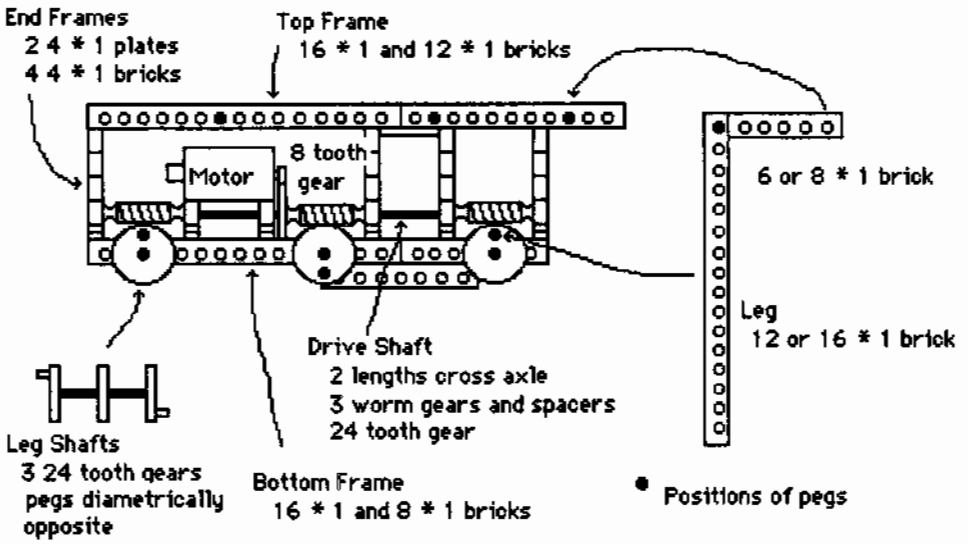


Figure 3 Hexapod 1

The machine is surprisingly stately in its movements, gently rising and falling as it progresses. That is a sign of wasted energy (the problem is known as 'geometric work'), for good leg mechanisms would keep the body height constant. That is an issue in real machines, and animals too, and it has been claimed that women, with shorter legs, and therefore less up and down movement, are more efficient walkers than men. Obvious experiments are with legs of different lengths, and perhaps with 40 tooth gears instead of 24.

Another leg mechanism experimented with briefly was, in effect, the previous one upside down, with a sliding pivot instead of the pivoting link (the 'knee'). It can be used with or without feet, but the steps are short and the system seems to have little promise. We also tried to make legs which moved rather like oars, but without success.

These machines can move only in straight lines. To be able to turn, either an extra degree of freedom is needed in each leg, in other words, another hinge and more mechanism, or

the machine can be built in three sections, with universals connecting their drive shafts, and steered by 'bending' with a separate motor and linkage.

Legs like these can be used for quadrupeds, and our first hexapod lost two legs and part of its structure to become a quadruped. We tried it with the trot gait, and also with the amble and Alexander's optimal turtle gaits (A, B, and C in Figure 1). The machine did work with all three, but looked decidedly precarious as it pitched and rolled, and was very slow. For a human rider, definitely most uncomfortable. It was improved slightly by fitting wide feet to reduce the rolling. It can obviously be used to explore different gaits provided you don't mind it frequently falling over. For success with a quadruped, each leg would have to be individually driven so that at least three legs are on the ground at all times. This would definitely be an interesting building and programming task.

A quadruped that keeps all four 'feet' on the surface, remaining statically stable, is a much more feasible proposition, although the resulting shuffling may not be considered real walking. Many years ago there was a toy tortoise, the 'Toytoise', which consisted of a pressed steel shell and two pairs of legs which rotated in opposite directions. The 'feet' were small rubber wheels with a simple free-wheel system, and the whole thing was operated by squeezing a handle, connected to the mechanism by cable. It would move forwards only, but could turn in either direction by biasing the leg rotations one way or the other.

The only difficulty with a LEGO version is the ratchet wheels, and this is solved with a paper clip, as shown in Figure 4. As the wheel rotates one way the clip rides over the tread in the tyre, the other, it drops in and locks the wheel.

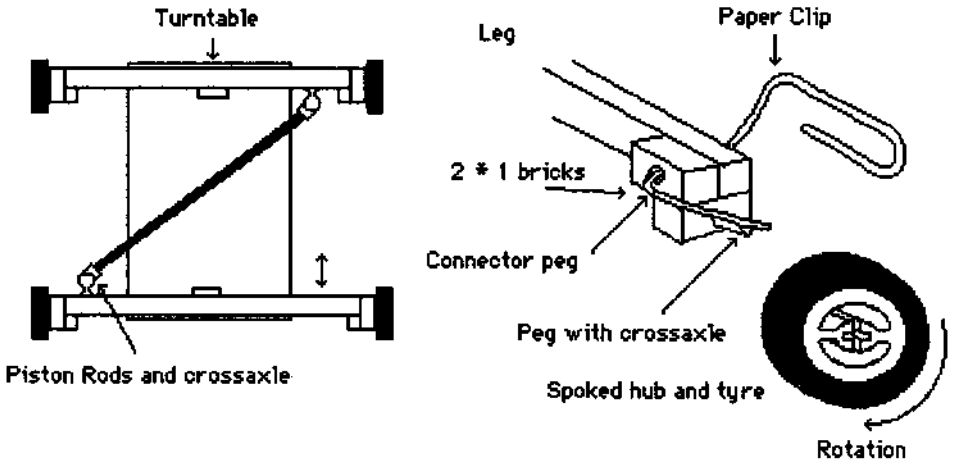


Figure 4. Shuffling vehicle with ratchet wheels

The legs are connected by a push-pull rod, and the whole mechanism driven through reduction gearing and another push-pull rod, not shown on the diagram. It is possible to make machines like this turn, either by having the motor operate such that the crank is biased towards one end of its travel or the other, or by varying the distance between the leg pivots. The first requires programming and position sensing, the second, an extra motor and gearing, and a sliding or pivoting leg mounting.

The same ratchet system can be used to make an 'inch worm' machine. The frame vaguely resembles a step ladder, with the motor opening and closing the two halves, and our version was made by rebuilding the toytoise vehicle. A different ratchet mechanism is used on the leg mechanism shown in Figure 5. The geometry is such that the wheel end of the leg moves in a straight line.

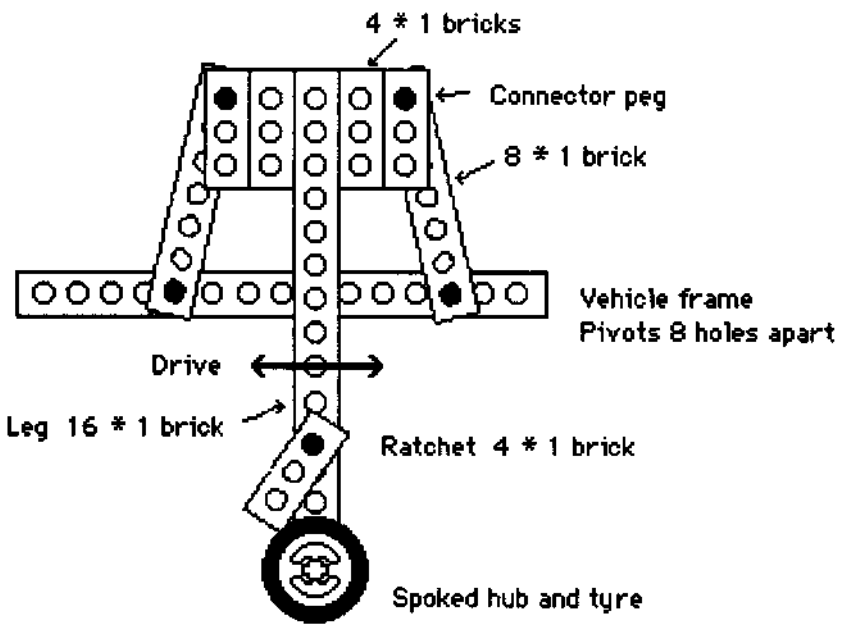


Figure 5 A 'constant length' leg

The prototype machine used the trot gait, although almost any gait would work. It should be possible to make such a machine turn by driving the left and right legs at different speeds with separate motors, but this has not been tried.

Little has been said about the programming aspects of these machines. In most cases very little programming is needed, unless sensors are used to monitor the positions of parts of the machines, during turning for instance. As the complexity of machines increases, with numbers of motors, so will the complexity of programming. Machines interacting with each other will also require programming.

As with all LEGO and Logo activities, students will devise their own theories: that is, after all, an essential part of the process of Piagetian learning. A group of Year 8 students working with the hexapod and toytoise machines decided

that they would both move at the same speed, because they used identical motors. They connected all the available cables, placed tables in line, and proceeded to race the machines. Sure enough, they did move at very nearly the same speed. It was the end of their time with the LEGO materials and they never did falsify their theory. But it was real science in action.

Conclusions

With LEGO and Logo we can now explore, in a small way, some of the issues in robotic locomotion, and our aim has been to develop simple, but workable, machines. The references listed below, particularly Todd's book and the special issue of *The International Journal of Robotics Research*, contain much for those with the interest to pursue the subject to some depth.

References

Alexander, R. (1984) 'The Gaits of Bipedal and Quadrupedal Animals', in *The International Journal of Robotics Research*, Vol 3 No 2, Summer .

Hirose, S.(1984) 'A Study of Design and Control of a Quadruped Walking Vehicle', in *The International Journal of Robotics Research*, Vol 3 No 2, Summer .

Jameson, J. (1985) 'The Walking Gyro', in *Robotics Age*, January .

Pawson, R. (1985) *The Robot Book*, Colporteur Press.

Pearson, K. and Franklin, R. (1984) 'Characteristics of Leg Movements and Patterns of Coordination in Locusts Walking on Rough Terrain', in *The International Journal of Robotics Research*, Vol 3 No 2, Summer .

Raibert, M. and Sutherland, I. (1983) 'Machines That Walk', in *Scientific American*, Vol 248 No 1, January.

Sutherland, I. (1984) 'Footprints in the Asphalt', in *The International Journal of Robotics Research*, Vol 3 No 2, Summer .

Time-Life editors (1986) *Robotics*, Time-Life Books .
Todd, D. (1985) 'A Pneumatic Walking Robot', in *Robotics Age*, January .
Todd, D. (1985) *Walking Machines: An Introduction to Legged Robots*, Kogan Page.
Waldron, K. (1984) *et al* 'Configuration Design of the Adaptive Suspension Vehicle', in *The International Journal of Robotics Research*, Vol 3 No 2, Summer

recursion *n.* (*see* recursion)

Peter J. Carter

The words of Friedman and Felleisen in their preface to *The Little Lisper* (1987), an amusing Lisp text with a clear emphasis on recursion. It is not necessarily my purpose in this paper to try to convince you that recursion is a 'powerful problem solving technique', but to give some familiar examples of recursion, and to give an explanation, by analogy, that may help to explain what happens during the execution of recursive Logo procedures.

Whether we realise it or not, recursion is common in our lives. News broadcasts and telecasts frequently include segments from reporters outside the studio, and in turn, outside reporters often present other speakers. We have no difficulty keeping track of the report (within the report) within the report. Modern telephone systems allow us to put a caller on hold while we make another call to find some information before returning to the original conversation. Language itself contains recursive elements, with clauses and phrases nested within sentences, and this is more apparent in some languages, German for instance, than others.

In the arts, music, with often subtle key changes, shows recursive structure, although unless the listener either has perfect pitch or can follow the score, the level of nesting

is easily lost. Shakespeare's 'Hamlet', with its play within the play, is but one example of recursive theatre. Most people are familiar with Russian dolls, nested one inside the other, and recursive pictures are quite common. It was just such a picture on a book cover that disturbed Sherry Turkle (1984):

> *'Whenever I looked at the photograph or the book I couldn't stop thinking about them, yet could find no way to capture for myself or for anyone else exactly what it was that was so upsetting and gripping for me.*
>
> *Other children meet this experience in the form of questions about where the stars end or whether there is ever a final image when mirrors reflect mirrors. In all of these cases, what disturbs is closely tied to what fascinates and what fascinates is deeply rooted in what disturbs.*
>
> *When I was in trouble with self-referential pictures I could get no help. The adults around me were no better able to handle the infinite series of ever smaller little girls than I was, except to assert their authority by telling me not to think about such things. Children's encounters with ideas like self-reference, infinity, and paradox are disturbing and exciting and are made all the more mysterious by the fact that appeals to parents about them are likely to provoke frustrating admonitions not to think about such slippery questions. Yet such questions become storm centres in the mind.'* (p 23)

Other workers have found, as did Turkle, that quite young children are able to recognise recursion and self-reference. Some children's literature, for instance *The Cat*

in the Hat Comes Back, contain recursive elements. But there are more subtle recursions in nature. Benoit Mandelbrot has drawn attention to the 'fractal' nature of much of the world (and fractals have become a popular means of demonstrating recursion). William Poundstone's thesis (1985) is that the universe can be described in a few simple, recursive rules, demonstrating his case with the game of 'Life' and other cellular automata. (Brian Silverman has, in effect, turned the book into software with *The Phantom Fishtank*.) Recursion in some form is inescapable.

Logo and recursion are almost synonymous, and there are occasional arguments about whether a programming language for students should use recursion so widely or be equipped with looping structures: REPEAT...UNTIL, FOR...DO, WHILE...DO, etc. (Logo's REPEAT is less powerful than those.)

More than once I have listened to discussions in the Pascal community about the value of both REPEAT...UNTIL and WHILE...DO, because students tend to be uncertain about which to use in which circumstances. Abelson, Sussman and Sussman (1985) are characteristically blunt in their comments about such choices, and their machine implementation: '...special iteration constructs are useful only as syntactic sugar.' (p 33) In other words, there need be no choice, iteration can be performed more effectively with tail recursion.

Many writers of Logo books, especially in the early years of Logo, offered explanations that were plainly wrong. That may have been due to their unfamiliarity with Logo compared with, say, BASIC, but the impression they leave is false. One example, by Gruber (1983):

> *'This brings us to what I would consider a more controversial aspect of the language: recursion, that peculiar programming construct in which a procedure calls itself. Here's a very simple example...* [With a line missing in the original]

```
TO PRINT.TO.10 :NUM
IF :NUM > 10 [STOP]
PRINT.TO.10 :NUM + 1
END
```

> *...The lines are executed in order, just as in BASIC...*
>
> *The key is the next line...*

```
PRINT.TO.10 :NUM + 1
```

> *This is the recursion, the procedure calling itself but with the argument :NUM increased by 1. Actually, in this simple example, the recursion is exactly equivalent to a GOTO the IF line.'* (p 14)

The fact is that the recursive call is *not* equivalent to a GOTO, and the procedure does not call 'itself'. Elsewhere in the article, Gruber discusses local variables, but nowhere does he offer a clear explanation of recursion. One is left with the impression of a loop. Recursion is *not* looping. Recursion is nesting, interruption and deferral; procedures calling new copies of themselves.

Other writers of the same period offer descriptions similar to Gruber's, or no explanation at all. Heller *et al*, for example, have six pages devoted to very cumbersome fractal procedures, but make no attempt to describe or explain the process, and the word 'recursion' does not appear in the index.

Fortunately, other writers, who seem to have come from a mathematics/computer science/LISP background have been more helpful. The most accessible writer is Brian Harvey, whose *Computer Science Logo Style* trilogy should be required reading for teachers, and course planners and administrators. In Volume 1 (1985) he presents four Chapters 5, each dealing with recursion from a different viewpoint: the combining method, the little people method, the tracing method, and the leap of faith method. I would suggest that you read these at your leisure.

Let me offer an explanation that is similar to Harvey's 'tracing method', but embellished by ideas adapted from Douglas Hofstadter, and taken from *Thinking Logo* (Carter 1987, pp 16..17). For a procedure, a small numerical example:

```
TO Counting2 :number
IF :number = 0 [STOP]
Counting2 :number - 1
PRINT :number
END
```

> *Turtles are known to occasionally mutter to themselves as they work. One was once overheard as it worked on this problem: (Seems it had access to a photocopier...)*
>
> *"Hmm, Counting2 3. :number's not 0, so what's next? Counting2 3 - 1." (Makes copy of Counting2, writes 2 on it, and puts Counting2 3 on the table.)*
>
> *"Hmm, Counting2 2. :number's not 0, so what's next? Counting2 2 - 1." (Makes copy of Counting2, writes 1 on it, and puts Counting2 2 on Counting2 3 on the table.)*

"Hmm, Counting2 1. :number's not 0, so what's next? Counting2 1 - 1." (Makes copy of Counting2, writes 0 on it, and puts Counting2 1 on the stack.)

"Hmm, Counting2 0. :number is 0, so that's the end of that one!" (Tosses it into the bin and picks up the top copy from the pile on the table.)

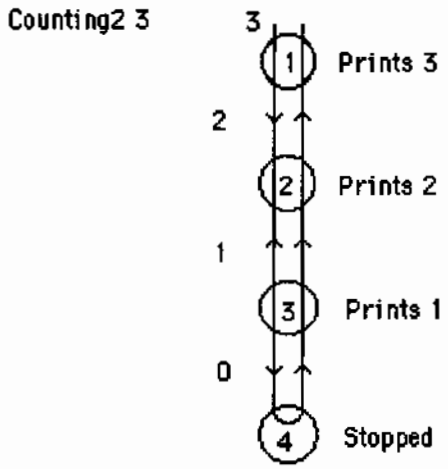
"Ah yes, print the value of :number." (He writes a 1 on the screen, drops Counting2 1 into the bin and picks up the top copy.)

"Ah yes, print the value of :number." (He writes a 2 on the screen, drops Counting2 2 into the bin and picks up the top copy.)

"Ah yes, print the value of :number." (He writes a 3 on the screen and drops Counting2 3 into the bin.)

"Done." he murmurs with a satisfied grin.

In many instances of recursion a process is replaced by a modified copy of itself; PolySpi is a classic example. In others, like Counting2, the process is deferred until the simplest case is finished and the recursion 'unwinds'. Below is the first of many diagrams showing how recursive processes work. The numbered circles represent calls to the procedure, 4 in this instance. The downward pointing arrows on the left show values being passed. If there were outputs they would be shown upwards on the right but Counting2 simply prints its values.



The choice of the word 'stack' is deliberate. The Logo interpreter uses a stack to store the variables for each instantiation of the procedure: at each call, the values are 'pushed' to the stack, as the recursion unwinds, they are 'popped' off. When a Logo system crashes with an OUT OF MEMORY error the problem is really a stack overflow, usually caused by a recursive procedure without an adequate stop rule. Mention of the bin in the story is an allusion to 'garbage collection', the method by which the interpreter reclaims unused memory.

Most modern Logo implementations have a TRACE facility, which allows inputs and outputs, and the level of recursion, to be monitored. A list processing example:

```
TO Remove :item :list
IF EMPTY? :list [OUTPUT []]
IF :item = FIRST :list [OUTPUT Remove
:item BUTFIRST :list]
OUTPUT SENTENCE FIRST :list
Remove :item BUTFIRST :list
END
```

```
SHOW Remove "potato [apple orange banana potato pineapple apricot]
 Remove potato [apple orange banana potato pineapple apricot]
  Remove potato [orange banana potato pineapple apricot]
   Remove potato [banana potato pineapple apricot]
    Remove potato [potato pineapple apricot]
     Remove potato [pineapple apricot]
      Remove potato [apricot]
       Remove potato []
       Remove Outputs []
      Remove Outputs [apricot]
     Remove Outputs [pineapple apricot]
    Remove Outputs [pineapple apricot]
   Remove Outputs [banana pineapple apricot]
  Remove Outputs [orange banana pineapple apricot]
 Remove Outputs [apple orange banana pineapple apricot]
[apple orange banana pineapple apricot]
```

It is not difficult to draw a diagram from such a trace, but the task becomes more interesting when the recursion branches...

```
TO CountWords :object
IF EMPTY? :object [OUTPUT 0]
IF WORD? FIRST :object [OUTPUT 1]
OUTPUT (CountWords FIRST :object) +
(CountWords BUTFIRST :object)
END
PR CountWords [This is [a list][of
lists]]
```

(see diagram on next page)

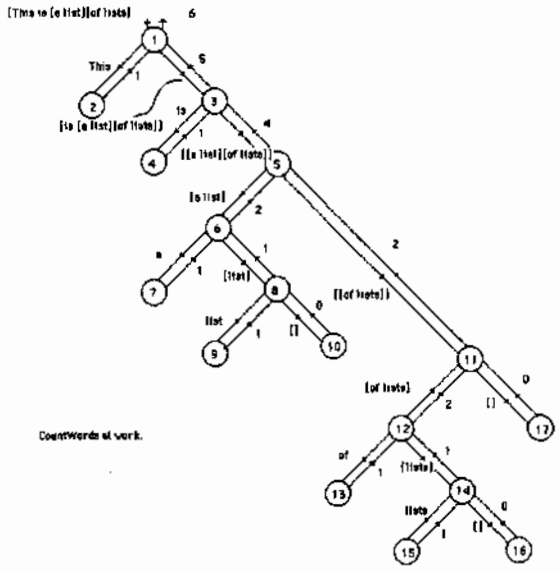
How should recursion be introduced? Most writers begin with tail recursion (PolySpi etc.) But there is a body of opinion that suggests that 'embedded' recursion be used first, with tail recursion coming later as a special case, this sequence reducing the likelihood of recursion being

seen as looping. I generally begin with tail recursion, but always with stop rules, and emphasise that the procedure is not calling itself, but a new copy of itself. A procedure like this can be used...

```
TO Steps :length
IF :length < 10 [STOP]
FORWARD 10 RIGHT 90 FORWARD 10 LEFT
90
Steps :length - 10
END
```

...and then changed to:

```
TO Steps :length
IF :length < 10 [STOP]
FORWARD 10 RIGHT 90 FORWARD 10 LEFT
90
Steps :length - 10
RIGHT 90 FORWARD 10 LEFT 90 BACK 10
END
```



How well do students cope? Turkle (1984) cites the case of 'Matthew' who was able to understand recursive procedures at age five. I have seen eleven and twelve year olds cope with tail recursion, but on the other hand I have seen Year 11 and 12 students in difficulties (and not only with recursion. I question the value of current Computing Studies courses for many students). John de Figueirdo (1989) found many of his students in difficulties. I have a strong suspicion that many of the problems can be overcome with adequate descriptions and explanations of recursive processes.

How does one design recursive procedures? Perhaps the advice of Hofstadter (1986) is the most concise:

> *'...To spell out the exact nature of this recursion-guiding pathway, you have to answer two Big Questions:*
>
> *(1) What is the embryonic case?*
>
> *(2) What is the relationship of a typical case to the next simpler case?*
>
> *Now actually, both of these Big Questions break up into two subquestions (as befits any self-respecting recursive question!), one concerning how you recognise where you are or how you are to move, the other concerning what the answer is at any given stage. This, spelled out more explicitly, our Big Questions are:*
>
> *(1a) How can you know when you've reached the embryonic case?*
>
> *(1b) What is the embryonic answer?*

(2a) From a typical case, how do you take exactly one step toward the embryonic case?

(2b) How do you build this case's answer out of the "magically given" answer to the simpler case?

Question (2a) concerns the nature of the descent towards the embryonic case, or bottom line. Question (2b) concerns the inverse aspect, namely, the ascent that carries you back up from the bottom to the top level.' (p 416)

Long ago (in computing terms) LISP was given the power of recursion for the symbol manipulation of AI research. Logo has that power without the confusing syntax and unnecessary choices of looping constructs of other languages. Logo is a language which allows students the freedom to explore and be creative.

One is mindful of Hofstadter's Law: 'It always takes longer than you expect, even when you take into account Hofstadter's Law.'

References:

Abelson, H. and Sussman, G. with Sussman, J. *Structure and Interpretation of Computer Programs* MIT Press, 1985.

Carter, P. J. *Thinking Logo* 1987.

de Figueirdo, J. 'Students don't think recursively' *POALL* Vol 4 No 4, pp7..15.

Friedman, G. and Felleisen, M. *The Little LISPer* MIT Press 1987.

Gruber, A. 'From LISP to Logo' in *Call A.P.P.L.E.* Vol 6 No 8 August 1983.

Harvey, B. *Computer Science Logo Style: Intermediate Programming* MIT Press 1986.

Heller, R., Martin, C. and Wright, J. *LOGOWORLDS* Computer Science Press 1985.

Hofstadter, D. *Gödel, Escher, Bach: An Eternal Golden Braid* Penguin 1980.

Hofstadter, D. *Metamagical Themas: Questing for the Essence of Mind and Pattern* Penguin 1986.

McDougall, A. 'Teaching about Recursion in Logo: a Review' in Dupé, T. *ACEC '89-Backup the Future* CEGACT 1989.

Poundstone, W. *The Recursive Universe* Wm Morrow & Co 1985.

Silverman, B. *The Phantom Fishtank* LCSI 1987.

Turkle S. *The Second Self: Computers and the Human Spirit* Granada 1984.

The Use of Logo in Mathematics

Belinda Robinson

Introduction

Since the Logo language has been made readily available to schools, there has been much written about its contribution to general problem solving skills, especially in the primary school.

Introductory books on Logo abound, providing ideas to learn procedures and draw attractive patterns. The power of recursion is demonstrated (often in such a way as to create false mental models) and the idea of top-down programming is presented - in many cases with Papert's famous house exercise (Papert, 1980). Some books give examples of using Logo in teaching Computer Science.

Material showing the use of Logo as a tool to teach secondary mathematics is becoming increasingly available but is spread mainly throughout journal articles, conference proceedings and books of a more general nature. An increasing body of literature suggests that many areas of the mathematics curriculum can be enhanced by the use of Logo and turtle geometry in particular though it must be recognised that other areas are, if anything, hindered.

The value of using Logo in mathematics is emphasised by enthusiasts who claim advantages in teaching about angles,

polygons coordinates, directed number, transformations, symmetry, infinity, variables and other areas of mathematics. While the majority of the work has used turtle graphics, there is some work being done with list processing, particularly in the area of algebra.

This document aims to draw together some of the ideas and research findings relating to using Logo in teaching secondary mathematics. In particular, some of the microworlds that have been outlined in various locations have been developed expanded and some others created.

Why Use Logo in Mathematics?

Logo was developed by Seymour Papert as a way of helping children to experience a *Mathland* in the same way that the best way to learn French is to go to France (Papert, 1980). The idea was to bring children into contact with powerful ideas connected with geometric thinking. He argued that children need to see a need for mathematics in what they do so that they learn from doing.

This same view was expressed by Brown in his report on experiments with 11-13 year old students who used Logo to investigate mathematical problems. He points out that Logo gives a different approach from the expository style used even in some software packages and found that the benefits of the computer was for "promoting better understanding of mathematical principles through allowing the students to operate within a mathematical environment where ideas can be tested." (Brown, 1986, p.40)

The processes used in developing appropriate Logo procedures are themselves processes promoted in the current mathematics syllabuses, viz design, code, test and debug. The exercise of programming itself can be argued to be a mathematical activity.

When the exercise of programming is done in the context of a mathematical problem with a language which does not interfere with the mathematics, there is a much stronger case for using Logo in mathematics. As an example, having the origin at the centre of the screen (as in Logo) encourages understanding of the Cartesian plane in contrast to AppleSoft BASIC where the origin is in the top left corner of the screen.

The additional benefit of using Logo is that in "teaching" the computer, the student gains a greater understanding of the topic. Not all Logo activities need, however, to involve programming from scratch. In some cases it is appropriate to provide tools that can be used as a black box to experiment with a mathematical idea. The procedures for such black boxes can be examined later for greater understanding. An example of this is the procedures for coordinate geometry (in part 2) or for circle properties (Computer Education Unit, 1986).

One particular advantage of using Logo is in the added motivational factor which is linked with the fact that the Logo environment can appear to be more concrete than formal mathematics. In the area of algebra, Logo can be used as one of a number of concrete aids. Logo microworlds can provide a new environment with which students can relate.

Value of procedural definitions?

In his recent study of how students develop algorithms, Ronnie Goldstein came to the conclusion that procedural definitions of algorithms are more concrete than algebraic definitions. As an example, he compares $a^2 + b^2 = c^2$ with the English statement of Pythagoras' theorem. Clearly the former is more abstract.

An algorithmic statement of Pythagoras' theorem could be:

To find the length of the hypotenuse of a right angled triangle:

1. Square the length of one of the sides.
2. Square the length of the third side.
3. Add the results of 1. and 2.
4. Take the square root of the result of 3.

By writing such an algorithm, students can gain a greater appreciation of what they are learning.

Writing a Logo procedure takes this idea a step further as students have to teach the computer how to do something. It gives them a reason to formulate algorithms and to write them without any ambiguity.

An example quoted by Goldstein is that of two girls trying to teach the computer to calculate percentages before they had any formal solution given to them. They knew that 5% meant 5p per pound and devised various strategies including:

```
TO PER :PERCENT :AMOUNT
MAKE "ONEPER :AMOUNT/100
PRINT :PERCENT/:ONEPER
END
```

which did not work when they tested it with some results they'd worked out themselves. Eventually they reached

```
TO PER :PERCENT :AMOUNT
PRINT :PERCENT * AMOUNT/100
END
```

giving them both a working solution and a feeling of satisfaction from solving a problem to get it. It also showed a positive response to errors as they worked through their bug to reach a correct solution.

Does Logo Work?

A much quoted study on the effects of Logo on learning was conducted by Pea in 1983. He found that there was no significant difference but even Pea recognised that he had asked the wrong questions (Pea and Kurland, 1984). As Papert said

> *In the presence of computers, cultures might change and with them people's ways of learning and thinking. But if you want to understand the change, you have to centre your attention on the culture - not on the computer.*
>
> (Papert, 1985, p.54)

Clements undertook a study on the effects on cognition of Logo activities. This was as a response to the studies that failed to show strong increases in mathematical ability. His results showed increases in development of the skills of: classification and seriation; on deciding on the nature of a problem and selecting a mental representation; comprehension; creativity and describing directions. They did not show significant gains in mathematical achievement. Like some other studies on Logo (eg Clements and Gullo 1984, Pea and Kurland 1984) the Logo activities were directed at late primary school and at general uses of Logo rather than at effects of using domain specific microworlds.

Studies of Logo have tried to measure changes on standardised tests as a result of using Logo but the very fact that what is tested is not all that is learned, makes the tests of limited value. Logo is more a philosophy of education and develops process rather than product. There is a diversity of ways in which it is used in classrooms. Research at one or other end of the continuum concentrating on the final

product of courses is therefore unlikely to provide any real indication of its value.

Emihovich and Miller (1988) argue that in order to understand Logo, we need to modify the way in which we study it. One difficulty in evaluating the effect of Logo is encountered when studying the product as this ignores how the learning process differed.

Formal studies of using Logo in teaching mathematics through specific microworlds are not common. There is, however, a growing interest in this use of Logo evidenced by the increasing number of publications with ideas for implementation and anecdotal accounts of the use of Logo. One extensive study was the Logo Maths Project, a three year longitudinal study conducted by Hoyles and Sutherland starting in September 1983. It specifically looked at using Logo in a mathematics classroom. It examined the role of collaboration and the role of the teacher and concluded that amongst other benefits, the use of Logo allows teachers to see how children are thinking and the strategies they use.

Styles of Teaching with Logo

Since Papert published his book *Mindstorms*, there has been a strong following of opinion that students need to be left to themselves to explore with Logo. Indeed, he stated that Piagetian learning is "learning without a curriculum" (Papert 1980). Those advocating this style have been described as MIT dreamers (Wills 1984 in Oakley, 1986) or as subscribing to "techno-romanticism" (Kurland 1983 in Oakley, 1986). Others have argued that it is necessary to teach the powerful ideas and some have taken the extreme of using Logo for direct instruction. The dilemma has been summed up:

> *On the one hand, if we adhere to the ideal of totally spontaneous, non-directed, Piagetian learning, we may find children not acquiring the powerful ideas as much as we hoped. On the other hand, if we push the powerful ideas too brusquely, we are back to classroom-type teaching with all the notorious side effects (notably alienation and a feeling of failure).*
>
> (Leron, 1985, p.27)

If one of the extremes is chosen, it tends to fail. In the Piagetian style, students use trial and error techniques without developing the powerful ideas they're meeting. Leron proposes a "Quasi-Piagetian Learning" model which gives more place to teachers and various supplementary learning materials but retains a degree of freedom for the learner. The various *microworlds* and tasks that have been suggested by a variety of writers tend to fit in with this description. It allows both a content centre and a learner centre in the classroom activities.

While it is essential that teaching within such microworlds does not become didactic, there is still a need for intervention by the teacher to ensure that students grasp the mathematics that they are meeting. An advantage of Logo is that it can help students to make sense of some events just beyond their level of understanding. While observers may be conscious of the mathematics within the turtle environment, the children must develop to the point of full understanding. Passing from the stages of informal thinking to the more formal thinking is important but needs structure to ensure that it happens.

An example of this growing understanding is in the use of variables. An introduction to variables could be as inputs to

procedures and then moving to direct assignment and passing of variables at a later time. At the introductory stage, students may have been using a concept still outside their full understanding but which will form a model for later study. Leron reports finding students with only a partial understanding of many ideas and it is important that this is seen as an intermediate step in learning and that appropriate structure is provided to ensure that students progress to a full understanding.

Features of Microworlds

Microworlds are sets of Logo procedures which can be used as primitives but can also be studied themselves. They allow students to work at top-level, executing commands or writing additional procedures as well as using the extension procedures.

Some microworlds may appear at first glance to achieve the same objectives as some specific mathematics software. Both the Logo microworld and the software package may have a place. A software package that allows students to graph functions will help them to get a feel for different functions and the effects of changing certain variables. Writing Logo procedures to graph functions will achieve this feel - though a lot more slowly - but also develop skills appropriate to sketching functions themselves such as looking for points of discontinuity (Logo will state that it can't divide by zero) and choosing appropriate scales. An advantage of programming is that the solutions become the students' own.

Where Does Logo Fit in Traditional Mathematics?

The debate here is that mentioned above - whether or not Logo should be linked to specific curriculums. Noss stresses that Papert's view of Logo as a "conceptual frame-

work" suggests that Logo programming is to allow students to participate in mathematical activity, ie to do mathematics rather than as a vehicle for learning mathematical content (Noss, 1987). This idea is that with this "mathematical way of thinking" students will find it easier to learn algebra or geometry.

In Noss's study with senior primary students, he noted that in letting students choose their problems they identified key concepts that would be needed and that these were viewed as a "vehicle for the content of the mathematics undertaken by the children". In his conclusion, he points out that the study did not use microworlds other than the standard turtle geometry environment but suggests that such microworlds would result in a more obvious connection between conventional mathematical content and the programming work. He felt that the use of microworlds could be a vital factor in promoting the integration of Logo into the secondary mathematics curriculum. Thus, even those recognising the need for freedom to explore in a mathematical environment see the place of microworlds with specific content to give specific direction to children's exploration.

Anyone with even a small familiarity with Logo, is aware that problems relating to geometry can be explored in microworlds - or in the unenhanced turtle microworld. Sadly, many still believe that Logo is merely an alternative for BASIC programming or that it is a tool for young children to learn left and right. In reality, the geometry that can be examined with Logo can include quite complex concepts such as limits, the idea of mathematical induction (using recursion) apart from topics such as fractal curves and uniform growth of biological curves. Common topics that can be addressed in Logo are:

> *constructions of triangles and quadrilaterals, inscribed angles and arcs of circles, elementary number theory, sequences and series, geometric transformations, vectors, similarity, tessellations, spirolaterals, Fibonacci numbers and the golden ratio, curve plotting, probability simulations, and all kinds of applications of recursive functions.*
>
> (Aieta, 1985, p 477)

Such a list can be extended to include limits, approximations to area, mathematical induction and complex numbers (depending on the implementation). Clearly, Logo is not just for the littlies!

How Does It Fit In with Normal Class Activities?

Different situations and beliefs have led to a variety of ways of introducing Logo into the mathematics curriculum in secondary schools. Some formal research has been conducted in very contrived situations. Brown's study in Staffordshire (Brown, 1986) drew pairs of students out of class for a day to work with the computer.

In the Logo Maths Project (Hoyles, Sutherland & Evans, 1985) a three year longitudinal study was undertaken with students aged 11 to 12 working in their "normal" mathematics lessons. The activities included exploratory activity at the initial stages where the goals were not clearly defined. This stage allowed students to become familiar with the learning environment. A range of projects was available for the students and occasional structured tasks were introduced by the researchers. Hoyles suggests, however, that there was scope for development of specific microworlds to link specific traditional mathematics to Logo projects.

In some cases in New South Wales, those experimenting to integrate Logo into the mathematics curriculum have implemented Logo in what they recognised as a less than satisfactory manner but in a way that was possible.

In 1983, Billabong High School in the Riverina implemented an experimental program of using Logo to teach a number of mathematical ideas (unpublished). Because it was part of a wider program, it was possible to have half classes with fifteen students for one or two lessons a week for half a year. In that time, students became familiar with Logo and used it to explore polygons, diagonals, circles etc. The difficulties faced were that in the initial stages there were groups of four to five per computer and, more importantly, that because of the fixed time when each class had access to the computer room, the lessons were divorced from their other class work. Those groups with only one lesson a week found major problems of lack of continuity. The experiment led to an appreciation of Logo and mathematics by some students and an interest in continuing with using the language by some of the teachers (including the present author).

During 1988, Bruce Horton at Mitchell CAE has continued working with a group of students he introduced to Logo in 1987 (Horton, 1988). The difficulty here was that the group had to cover the same work that was being covered by the rest of the class in a more traditional classroom. This has meant that in some cases, the problems solved with Logo were very contrived or at least not particularly different from the pen and paper presentation though in others such as Cartesian number plane problems (not included in the published paper), Logo was useful to add a new dimension to the task.

In other schools, attempts have been made to include a unit of work in which Logo was introduced and used for some specific explorations.

Ideally, computers would be available for small groups of two or three to use for mathematics tasks whenever Logo activities are appropriate. If Logo is to be seen as a tool to help in mathematics, it is preferable to have access whenever it is needed by having machines in the room. With limited resources, this is not generally feasible and is unlikely to mean more than one or two computers would be available. Given that there is not the level of facilities in the average school to allow this, one option is to timetable particular times for access to a computer room (without insisting that the explorations mirror other work but still drawing the links to other class work) while another is to have access to one or two computers for a large part of the lesson time and have groups cycle through using them. Where a discrete number of lessons are required for a particular unit of work using Logo, it is usually easier to get access to a computer room.

Reports from teachers at Billabong High School and Bruce Horton, suggest that it is important that lessons using Logo are not too spread out. One lesson a week does not allow a satisfactory level of retention from week to week to build sets of procedures. At least at the introductory phase, it is important to get students familiar with the operating system and elementary commands.

Time being at a premium in the secondary school, it may not be possible to allow the level of exploration that is wanted, especially if the whole Logo environment is new. As an increasing number of students have been introduced to Logo in the primary school, this will become less of a

problem and students will need less "black box" procedures to be provided. The microworlds designed by Niess for introducing informal geometry (Niess, 1988) take cognisance of the need to avoid allowing the requirements of the language masking the mathematics by using teacher prepared procedures so that students can explore at the top-level. In some of the examples in part 2, it may be necessary to provide building blocks where difficulties in recursion or difficulties with local and global variables are beyond the grasp of the students. In each case, however, there is room for students to explore to find solutions themselves.

Classroom Styles

The style of lesson that supports Logo microworlds is definitely not the expository style. Teachers have to be prepared to give students some control of their learning and move from a teacher centred approach to a student centred approach. In a Logo classroom, there will not be silence and that students will not be working on exactly the same exercise. Rather they will vary problems to explore even where the problems posed initially are the same.

Flexibility with goals is important. Where students have some degree of freedom to choose their own tasks, they tend to be more motivated. Generally, the freedom will be within certain boundaries but students should not be too rigidly constrained. Some microworlds will facilitate this more than others but freedom can be given by making a problem definition free enough to allow a degree of choice. Even the seemingly simple task of drawing a square can be done in different ways. One group may define a square in terms of four corners requiring a repeat of FD 30 RT 90 FD 30 while others will (more conventionally) repeat drawing a side and turning.

It is important that some of the goals are clear to the students in order that they realise that they are learning. There can otherwise be a degree of frustration (Pearson, 1986). Where mathematical microworlds are being used, there is less chance of this problem.

Generally, group work in pairs or threes seems to be ideal. When Logo was first used, groups tended to be used for pragmatic reasons - there were insufficient computers to spread further. Both experience in teaching in this situation and research has indicated that group work is preferable to one student per computer.

Collaboration between students using Logo was specifically studied in the Logo Maths Project. Student discussion of their learning was advocated strongly in the Cockroft report (Cockroft, 1982). Discussion helps students to consolidate their understandings of what is happening.

Within the Logo Maths Project (Hoyles and Sutherland, 1987), the researchers found that within pairs students would provide each other with new and challenging ideas, keep projects going, change the level of representation (concrete to abstract and back), seek to make the activities fit in with understanding, reflect and record and check. They also found that as the length of time increased, there was a change of roles within the pairs to fit the needs of different tasks. Effectively, there was a high degree of peer tutoring. Such a collaborative classroom is a non-competitive environment.

One way in which the use of the computer facilitates group work and shared discussion is by its public nature. It is somewhat difficult to hide what appears on the screen! On the other hand, the fact that new responses can be gained and "unexpected results" accounted for -

and that obvious errors do not have to remain forever - means that students are more willing to experiment.

Role of the Teacher

A change in classroom style means a change in the role of the teacher. Instead of being the information/knowledge provider, the teacher becomes a resource person to help students learn for themselves. In many cases the teacher becomes a fellow explorer where a new problem or variant has been chosen where the solution is not obvious to the teacher.

Students will not explore in a way that will develop skills without some direction, so the teacher is also a catalyst.

Teachers can create a non-competitive environment. Avoiding giving Logo assignments competitive grades is one element in this. Insistence that certain tasks are completed is necessary - no matter what the environment, some students may still be reluctant to do anything that may seem like "work". More importantly, some students will be hesitant getting started when there is a fear of failure that has been built up over time.

In contrast to the common approach in mathematics where the teacher spots the problem, provides the solution and moves on to the next group, the Logo teacher needs to be prepared to let students make mistakes and to search for the solutions.

The biggest difference in giving help is in not giving the answers. Some hints may be necessary to point to a particular direction. Some common bugs such as 0 in place of O may be spotted along with incorrect spacing. Such technical bugs can be pointed out quickly but intervening to tell students a rule or idea is unlikely to have any effect unless the need for such a rule has been found.

Working with a class of year nine beginners with Logo, I found that the need to allow the students to make errors was reinforced. The "traditional" Logo house was being drawn. One group of four was having a great deal of difficulty. Their procedures for square and triangle used different sizes to start with. During the course of a lesson, the roof ended up in every possible location both inside and outside the square. When they succeeded in getting their picture the way they wanted just before the end of the lesson, there were cries of jubilation. Despite the fact that the task that had been completed by other groups very quickly (they moved onto other problems), this group left the room chatting about how it had worked. (In later lessons their problem solving techniques did improve.)

One of the advantages of students working on different variations of a problem, is that the teacher will not always know the answer. The problem is then real rather than one of working out the solution that the teacher knows. Having the teacher as a fellow learner can be motivating to the students. The problem solving strategies suggested by the teacher (when needed) demonstrate a way of solving problems rather than illustrating a known method.

Class organisation can include student responsibilities. This will include distributing disks, distributing manuals or summary cards, organising turns on the computers and even in providing limited assistance. Some teachers have found a rule of "ask two first" works well, ie asking peers from other groups before asking for assistance from the teacher. This may work well especially at introductory stages of Logo when the operating system and editor may cause distractions from the task as well as in picking up "obvious" bugs. As with teacher intervention, it is important that complete solutions are not given.

Many mathematics lessons have a degree of "right" and "wrong" answers. In some cases, students are reluctant to write anything for fear that it may be incorrect. They also show reluctance to go back over their errors and prefer to go onto another. The idea of learning from errors is one that students dislike. With Logo, it is essential that students learn to "doodle, design and debug". Rather than errors, students should get "unexpected results". This change in attitude must be engendered by the teacher. Students need to be encouraged to look at what happens and find explanations for the results. Sometimes the result will help at a later stage.

An example of such a result is in the traditional activity of drawing regular polygons. There will always be some students who try to draw a triangle using a 60° turn. This can be put to advantage in drawing a hexagon.

If students are merely provided with Logo, it is naive to believe that they will make mathematical discoveries on their own. Appropriate problems must be offered in order to provide a framework for experimentation. In a computing studies class, the range of such problems will be wider than where the objective is to develop mathematical skills. There is, however, a wide range of experiments that can relate closely to the mathematics course and enhance understanding. These need to be presented to students in such a way as to motivate them.

An important aspect of the teacher's role is in developing links to other mathematical work. In many cases, while students may get value from a particular activity in itself, the value will be stronger if links to other knowledge are developed. This helps to develop the students' mental models in ways that will better allow such models to be built on at higher and more diverse levels.

An example of these links is in generalising drawing shapes. A particular strength from using Logo is in having students conceptualise objects clearly enough to draw them. If students are drawing a rectangle, they need to identify what makes a rectangle clearly enough to write them as move commands. These commands can be generalised to provide any sized rectangle at any point on the screen.

This uses the idea of a variable which one would want to see applied in general algebra. This, as with other links, will not happen automatically so teachers must help students to identify the links between Logo and other mathematical topics.

This is particularly important where the perspective used in Logo is different to that being used in other mathematical activities. Apparent contradictions must be shown to be compatible. As an example, the difference between an angle as a measure of turn and the trail left by a turtle turning must be made clear. In this case, it is essential that students recognise which turn is being measured.

Teachers need to provoke reflection on the process and structure of a problem. Willingness to persevere and experiment needs to be preserved by ensuring that students have success in their efforts. By this I do not mean that students should be helped as soon as they find a problem, but rather that they are assisted with just enough additional information to avoid frustration.

As well as creating links to other mathematical ideas and encouraging experimentation, there has to be a point at which the teacher helps to open up new ideas. Concepts of recursion, variables, procedures will all need to be introduced. This is particularly so in the introductory phase but continues on as students get more familiar with the language. Generally, it is better to introduce the ideas within

the context of their problems. Some students may need to be guided more directly to reach certain goals.

Experimentation helps students to develop mental models. An advantage of using Logo is in the discussion that is generated. As well as seeing the product, the teachers get an opportunity to see the process by which students are solving problems. Some such discussion can reveal faulty reasoning or mental models. These may be challenged directly by the students themselves as they get unexpected results. The process of recognising the problem and debugging can lead students to alter their models. In some cases, there may be a lack of willingness to alter the model or to find a way of reconciling the new situation. An example of this is where a student has recognised that the size of the turn required to draw a polygon is the result of dividing 360° by the number of sides but still expects a 60° turn to provide a triangle.

Clearly, there is a need for teachers to be on the watch for difficulties such as this. Where possible, the reasons that two seemingly conflicting results are compatible (such as by seeing that the measurement of 60° is not the angle being turned by the turtle) need to be explored. In other cases, teachers must challenge faulty mental models revealed through discussions and further examples.

Some difficult mathematical ideas may be avoided by students. In the Logo Maths Project, two students were seen to develop strategies to avoid using decimal input (Sutherland and Hoyles, 1986). A new project was presented to the students where the use of decimals was necessary. Finding that the results were "cute", the students proceeded to use the procedures with decimals extensively.

As well as providing the opportunity to explore a particular problem, teachers need to encourage further related experi-

mentation. Slight variations can be suggested to reveal new facts. Students will not always stumble on discoveries without some prompting.

Students will also need to be assisted in learning to plan. The tendency is to start typing before thinking about the solution. Making notes and writing about what has happened - be it in text or mathematical formulae - will not happen without teacher encouragement.

The teacher's role is therefore crucial in the Logo mathematics classroom. Real change requires that the students can experiment and feel relaxed enough to be able to get "unexpected results" (not mistakes) and that they are encouraged to share ideas and methods between groups and to connect such ideas with non-Logo activities. The difficulty for many is in the different learning style from many used in mathematics as Logo is a more student centred approach. The teacher becomes a facilitator. Hoyles described the challenge of the 80's as

> *to use the power of Logo to shift the focus of mathematical activity in schools away from learning the formal properties of symbolic codes and syntax to the discovery of their semantic meaning, and in so doing to allow the pupil's interpretation of the curriculum to become more visible and more influential on the way learning is organised.*
>
> (Hoyles, 1985, p 252)

Getting Started

Starting points for students with Logo will be dependent on what is to be achieved. Some microworlds require little knowledge of the editor and if no more Logo is to be done,

there is little point in spending time on this. While those presented in part two all use turtle geometry, other microworlds use list processing so turtle geometry would not be needed. More usually, the details of list processing can be ignored.

Before using the screen turtle, students can learn to "play turtle". The age of the students does not matter. Even adults find that giving instructions to a blind folded person to move around a path, is not too threatening and increases understanding. Well past the introductory lessons, I have found that suggesting that students walk the path can shed light on possible solutions. Chalk on the ground or tape on the carpet can be walked on to find exactly what moves and turns are required. Similar results were found in the Maths-Logo Project (Kingsnorth, 1986).

No matter what amount of time is to be spent, there must be time allowed for experimentation with turtle movement. In answer to how big is a turtle step, the simple answer is try it and see. Activities with mazes either programmed or on transparencies stuck on the screen abound. Target games and parking the turtle in a randomly drawn garage also help the students to estimate distances and angles of turn.

Simple pictures can be drawn - providing a context in which to introduce procedures. Series of polygons can be drawn - providing a context for inputs. As procedures are created for patterns, they can be used in superprocedures to illustrate the power of the tool. A simple rotation of a square or triangle provides a challenge to explore further.

Even in introductory activities, it is important that there is room for students to stamp their own ideas on what they create. A task like drawing a face can lead to alien figures or robot faces. A task of use this square and this triangle to

make the face shown on the board would be anathema to the use of Logo. A class in which I suggested the task of a face, created some wonderful drawings in which they calculated relationships between different features, used the setscrunch command to draw ovals, and made great use of estimation and trial and error.

Conclusion

There is a lot to be gained from using Logo in mathematics both from the point of view of the students seeking to understand mathematics and from the point of view of teachers seeking to understand students' understandings.

While we do not have a weighty body of evidence to prove that it works, we at least have evidence that it is at least as effective as traditional mathematics. Increasing use of Logo is being seen as teachers become convinced of its efficacy. This seeming contradiction can be partly explained in that research has tended to measure only a part of the use of Logo.

It seems that the usefulness of Logo in mathematics will increase as more microworlds are developed and become known and as more teachers are willing to adopt a less traditional approach to teaching.

Bibliography

Aieta, J. (1985), Microworlds: Options for Learning and Teaching Geometry, *Mathematics Teacher*, *78* (6), 473-80

Brown, K. (1986), Mathematical Investigations using Logo Part 1 *Mathematics in School*, May, 39-42

Brown, K. (1986), Mathematical Investigations using Logo Part 2 *Mathematics in School*, June, 35-39

Clements, D. and Gullo, F. (1984), Effects of Computer Programming on Young Children's Cognition, *Journal of Educational Psychology, 76* (6) 1051-1058

Computer Education Unit (1986) *Using Computers for Problem Solving in Mathematics*, NSW Dept of Education

Emihovich, C., & Miller, G. (1988), Learning Logo: The Social context of cognition, *Journal of Curriculum Studies, 20*(1), 57-70

Goldstein, R. (1988), Algorithms and Logo, *The Computing Teacher, 4*(2), 29-31

Goldstein, R. (1986), Mathematics after Logo, *Mathematics Teaching 115*, June, 14-15

Horton, B. (1988), Logo with a Junior Secondary Mathematics Class, *Information Transfer, 8* (3)

Hoyles, C. (1985), Developing a context for Logo in school mathematics, *The Journal of Mathematical Behaviour, 4*, 237-256

Hoyles, C., Sutherland, R. (1987), Ways of Learning in a computer-based environment: some findings of the LOGO Maths Project, *Journal of Computer Assisted Learning, 3*, 67-80

Kingsnorth, R. (1986) Playing turtle, *Micromath, 2* (3), 29-30

Leron, U. (1985), Logo Today: Vision and Reality, *The Computing Teacher*, 12 (5), 26-32

Niess, M. , Logo learning tools build informal geometry ideas, *The Computing Teacher, 15* (8) 1988, 11-13

Niess, M. (1988), Logo learning tools build informal geometry ideas, *The Computing Teacher, 15* (9), 22-28

Noss, R. (1987), How do children do mathematics with LOGO, *Journal of Computer Assisted Learning , 3*, 2-12

Oakley, J. (1986), Can Logo live up to its promise? in Frederick, B. (ed) *What to do with what you've got*, Proceedings of 4th annual conference, NSW Computer Education group, Sydney

Papert, S. (1980) *Mindstorms*, Harvester, Sussex

Pea, R. and Kurland, D. (1984), On the Cognitive Effects of Learning Computer Programming, *New Ideas in Psychology*, 3, 137-169

Sutherland, R. and Hoyles, C. Mathematics and Logo - A Development in our intervention strategies in Hoyles, Noss and Sutherland (eds) *Proceedings of the Second International Conference for Logo and Mathematics Education*, University of London, London

Structure and Process Microviews: Partial Understandings of Recursion in Logo Programming

Anne McDougall

Monash University

Clayton

Introduction

This paper reports part of a case study of development of understanding of recursion (McDougall, 1988; McDougall, 1990a). It presents evidence that, part way through the study, the subject had two apparently separate views of recursion. It interprets one of these as being related to recursion in structure and the other as describing recursive process. A time-line of the subject's experiences related to recursion was used to investigate the generation of these two views and the later development of a working relationship between them. Some implications for teaching are noted.

Data Collection for the Case Study

Data collection for the study took place over a three year period, beginning when the subject was nine years old and in fourth grade, and ending when she was eleven and in grade six. At varying intervals during this time 48 Logo programming sessions, totalling almost 45 hours, were held

in a home setting. Audio tape records of all these sessions were kept, as well as computer dribble files, hard copy print outs and some handwritten notes. Since the researcher and the subject were mother and daughter, other recursion-related incidents in the subject's everyday experience could be observed and recorded. As one means of gaining information about the subject's understanding of the ideas met in her Logo programming work, a peer teaching technique was used. In the latter sessions the subject taught Logo to a school friend of the same age.

"Two Kinds of Recursion"

The subject regarded the topic, recursion, as 'hard to teach' and asked for my assistance. I suggested some initial activities based on two turtle graphics procedures, one tail recursive and the other embedded recursive.

```
TO PATTERN N
IF :N =0[STOP]
SQR :N
PATTERN :N-15
END

TO SQR :N
REPEAT 4[FD :N RT 90]
END
```

```
TO SUPERPATTERN :N
IF :N=0 [STOP]
SQR :N
SUPERPATTERN :N-I5
SQL :N
END

TO SQL :N
REPEAT 4 [FD :N LT 90]
END
```

After some time working with her pupil on these procedures, the subject became highly critical of my choice of turtle graphics procedures as an introduction to recursion in programming.

> *It's not a very good pattern because the recursion in this activity or whatever that we're doing was meant to be in the words and numbers. But it was just a coincidence that it was a recursion-ish picture.*

> *... two different sorts of recursion. Like there's recursion in pictures, and recursion in words and numbers and things... No - the same sorts of recursion but they just - they are different... The recursion you were looking for was the words and numbers one, I'm sure.*

She could not explain the difference between her two sorts of recursion. However she seemed very confident in her own mind about the distinction.

She argued that the 'pictures recursion' in the products from the turtle graphics procedures would distract a learner from the 'words and numbers recursion' in the procedure itself, the latter being what we were trying to teach. In her view, list processing procedures, with no distracting recursive picture products, would provide a far better introduction to recursion in programming.

> *It'd be better to show [the pupil] the words and numbers. It'd be better to do it with The Cat in the Hat one rather than for a thing to make a picture, because then the recursion pops out and it looks like it's the picture rather than the words and numbers.... It's not a good example of words and numbers recursion because it's mixing it up with picture recursion - because the picture just coincidentally happens to be a recursive picture.*

Recursion in Structure and Process

My conjecture is that the subject's two different kinds of recursion were closely related to recursion in structure and recursion in process. Her distinction between 'pictures' recursion and 'words and numbers' recursion is a distinction between recursive structures, in which a static output or product looks recursive, and recursive processes, such as

those exemplified by many of the list processing procedures she had encountered. The list processing procedures have recursive structure in as much as they contain a self-referent subprocedure call, but the outputs or results are not obviously visually recursive in the structural way in which recursive pictures are. Thus, to show a learner about writing procedures for recursive processes, she argued that list processing procedures would provide a better introduction than turtle geometry procedures. With list processing procedures the learner might focus on the recursion in the procedure itself and not be distracted by a recursive picture resulting from running the procedure. In the discussion that follows I shall use Lawler's (1985) term, microviews, to refer to the subject's cognitive structures concerning recursion.

Development of Recursion Microviews

In this study the subject's early work with recursion occurred away from the computer. She was introduced to the idea with a picture-within-a picture illustration on a book cover (Balian, 1974), the 'in your third wish, wish for three more wishes' example (Abelson, 1982), and *The Cat in the Hat Comes Back* (Seuss, 1958), a story in which cats successively produce smaller cats from within their hats, perform a cleaning task, and then return, in reverse order, to their respective hats. The child subsequently recognised and generated many examples of self-reference, repetition with variation, and nesting of levels in pictures, stories, everyday materials and play situations. Some recognition examples were a ventriloquist whose doll itself played the part of a ventriloquist with a smaller doll, nested toy plastic cups and wooden Russian dolls, a doll's patchwork quilt in which some of the patches were printed with a patchwork design, and self-referent fragments in stories such as *The Mouse and His Child* (Hoban, 1976), *Winnie-the-Pooh* (Milne,

1926), and *The BFG* (Dahl, 1982). Situations in which she deliberately generated examples included designing a Christmas tree decoration shaped like a Christmas tree, standing between two mirrors, drawing finer and finer branching veins in a leaf picture, and writing a diary entry which ended an account of the day's activities with "Next I filled in my diary. In my diary I wrote ... " and repeated the whole day's entry. A microview concerning recursive structure would have been developed during the subject's experiences with these recursive pictures, stories and playroom situations. Independently of this a microview concerning particularly powerful Logo procedures was developed from experiences with list processing procedures (some but not all of which coincidentally used recursion) early in this subject's Logo work. Important among these was a procedure called PRETTYLIST (McDougall et al., 1982).

```
TO PRETTYLIST :A
IF :A = [ ] [STOP]
PRINT :A
PRETTYLIST BUTFIRST :A
PRINT :A
END
```

On her first encounter with this procedure, in her 7th Logo session, the subject did not use the word recursion to describe it, although she did use the word to describe a command containing nested REPEATS (in her 14th session): REPEAT 4 [REPEAT 4[REPEAT 4 [FD 55 RT 32]]] and a procedure to draw a nested squares design on which she worked in sessions 16 - 18. I surmise that the reason for this was that her mental model of recursion at the time of her first encounter with the PRETTYLIST procedure had been developed almost completely from experiences with recursive structures and she did not associate recursion with the

process she was exploring as she worked out how the PRETTYLIST procedure ran. However by her 37th Logo session, in which she criticised my choice of nested squares patterns for showing recursion to her friend and talked about her two kinds of recursion, she clearly had developed a far more complex model of recursion, at least part of which included consideration of recursive process. When, between her 7th and 37th programming sessions, and how, was this process-related part of her mental model of recursion developed?

From the study data a time-line could be made listing in chronological order all of the recorded incidents related to recursion over the three-year period of data collection. In her early Logo work, examples which the subject related to recursion, the nested REPEATS, attempts to make a nested squares design, and her recognition of recursive calls in two other pre-written procedures, were still essentially concerned with recursive structure. The issue of interruption or suspension of action in a procedure at a recursive call, characteristic of recursive process, did not arise overtly. This was not the case for the next recursive procedure she encountered, in her 32nd Logo session. We developed a Logo-like procedure that embodied the recursive process she had seen illustrated in the *Cat in the Hat* story.

```
TO HAT.CAT :CATLETTER
IF :CATLETTER = Z CLEAN.SNOW [STOP]
HAT.CAT :CATLETTER + 1
RETURN.TO.HAT
END
```

This procedure included an action, the return of each cat to its respective hat, as the recursion unwound. As we discussed the procedure I wrote out, indented, the recursively

called subprocedures to several levels, showing the suspension of action in the present procedure at each new recursive call and the resumption of action at each level as the recursion unwound. Although I was not fully aware of it at the time, it seems clear from the later data that this was the subject's overt introduction to recursive process. Our developing the HAT.CAT procedure prompted her to associate the idea of recursion with the PRETTYLIST procedure. Subsequently PRETTYLIST, in tail and embedded recursive versions, became for her a valued model for recursive programming. She used this procedure frequently as a reference in her own later programming work and as an example in her teaching and programming activities with her friend. The two microviews continued to exist as separate, independent entities. That they were quite robust is illustrated by another incident. After her criticisms of my using graphics examples the subject seemed much more confident about her own ability to teach her friend about recursion, and I withdrew from the teaching role. The subject showed her pupil some overhead transparencies I had made to illustrate the pictures, story fragments, and so on of her early recursion experiences. Near the end of the set of transparencies they came to the recursive diary entry she had written, and she commented to her friend, "That's about the first words one I think you've understood." Her use of 'words' here must be interpreted as meaning process-related, as earlier in the set of transparencies the children had looked at several story fragments (such as the excerpts from *Winnie-the- Pooh* and *The BFG,* clearly word rather than graphical examples) but these were examples of self-reference in the structure of the story rather than recursion including the suspension of action for a recursive call at a particular level in the process of the story, as was the

case with her diary entry. Thus the subject spoke about two kinds of recursion, identifiable as recursion in structure and recursion in process, and she did not understand recursion in product and in process to be necessarily mutually inter-dependent.

Developing a Working Relationship between the Microviews

It seems most likely that the subject developed a working relationship between her two disparate recursion microviews with the writing, with her friend, of what they called 'Overall' procedures, first to make PRETTYLIST run continuously (procedure OV in her 40th Logo session) and then to make on-going displays of spirals (the O procedure, session 44) and reflected nested polygons (the O2 procedure, sessions 45, 46).

```
TO OV
PRETTYLIST [HAIRY MONKEYS AT THE ZOO]
OV
END

TO O :N :AN
IF :AN = 360 [STOP]
SP :N :AN
CS
O :N :AN+1
END

TO SP :N :AN
IF :N = 0 [STOP]
FD :N
RT  :AN
SP  :N-10 :AN
END
```

```
TO O2 :N :T
IF :T = 360 [STOP]
FP :N :T
CS
02 :N :T+5
END

TO FP :N :T
PU HOME PD
IF  :N = 0 [STOP]
PU HOME PD
F  :N :T
PU HOME PD
FP  :N-15  :T
PU HOME PD
F :N - :T
PU HOME PD
END

TO F :N :T
REPEAT ABS 360/:T [FD :N RT :T]
END
```

Lawler describes the process of learning in a number of ways, in terms of the modification of existing microviews or the development of new microviews. One process for new microview development he calls the elevation of control. This is the creation of a new control element which subordinates previously independent microviews, in the sense of permitting their controlled invocation. One example of this process given by Lawler is the building of a higher level procedure to manipulate for investigation a Logo design procedure of interest (Lawler et al., 1986). The developing of these 'overall' procedures bears a striking resemblance to this form of Lawler's process of elevation of control.

Implications for Teaching

Recursion in programming is regarded as a difficult topic for students of all ages, which is not learned without teacher assistance (Lavallade, 1985; Dawson and Bell, 1985; Bowman and Seagraves, 1985; Carmichael et al., 1985; Noss, 1985; Hoyles et al., 1984; Kurland and Pea, 1984; Samurcay, 1986; Leron, 1985). Areas of difficulty with the topic have been outlined in the research literature on the teaching of recursion (see for example McDougall, 1990b). Some of these problems are reflected in inaccuracies and confusing treatments in some books on Logo (a review is provided in McDougall, 1989). Riordon (1984) advocated using off-computer examples of recursion, similar to some of those used in the study, to introduce and aid understanding of the topic. A number of writers have used this approach in books on Logo (see for example Bearden, 1983; Bowman and Seagraves, 1985; Cory and Walker 1985; Sharp, 1984; Watt, 1986a; Watt, 1986b). The account presented in this paper might appear to present a case against the use of off-computer examples of recursive structure as an introduction to the topic, but this would be too superficial an interpretation. One critical and frequent problem in learning about recursion is the development of mental models of embedded recursion as looping (see for example Kurland and Pea, 1984). In fact the literature indicates this is a fundamental and most common difficulty with the topic for learners as well as many teachers and textbook writers. At no time in their recursion work in this study did the subject or her pupil show confusion of recursion with looping or iterative processes.

That the subject in the study valued the picture and story experiences as an introduction to recursion is evident from her use of them as early recursion work for her pupil. Data from the study suggests that the pupil too, did not at first relate the recursive structure in the pictures and story frag-

ments to the recursive procedure writing activities of the children's computer work. However her full participation in their later programming projects indicated that she too had developed a working relationship among the ideas and could use them effectively in programming with recursion.

Thus introductory examples of recursion in structure did not lead directly to an understanding of recursive processes. Nevertheless these examples illustrated aspects of recursion which distinguish it from iteration, and did so sufficiently clearly to avoid the otherwise common lack of distinction between these two ideas. This teaching approach certainly did not preclude, and may have facilitated, later development of more complex, accurate and useful mental models of recursion.

Summary

This paper has examined some partial understandings and mental models of recursion as they developed in a 9 - 11 year old child.

Her early encounters with recursion, in stories, pictures and so on, had mainly been examples of recursion in structure. The process-related part of her model of recursion was developed by the making of a Logo-like procedure to represent the story about *The Cat in the Hat.* The importance of the list processing procedure, PRETTYLIST, as an early model for recursive procedures was noted. However she did not at first relate this to the recursive structures she had met away from the computer, evidenced by her discussion of two kinds of recursion. It was suggested that these referred essentially to recursive structure and recursion in processes. She did not understand recursion in process and recursion in product to be necessarily mutually inter- dependent. She preferred to use recursive list processing

procedures for teaching recursive programming, as she regarded visually recursive graphics structures resulting from procedures to be distractions from the recursion in the procedures themselves. Lawler's work on microviews in cognitive development was used to assist this interpretation. It was suggested that, although introducing recursion using examples of recursive structure did not lead directly to an understanding of recursive process, this approach nevertheless was valuable in avoiding the widely reported learner confusion of recursion with iteration.

References

Abelson, H., (1982). *Logo for the Apple II.* Peterborough: Byte/McGraw Hill.

Balian, L., (1974). *Humbug Rabbit.* Nashville, Tennessee: Abingdon.

Bearden, D., (1983). *1,2,3 My Computer and Me! A Logo Funbook for Kids.* Reston, Virginia: Prentice-Hall.

Bowman, B. and Seagraves, K., (1985). "Picturing recursion" *The Computing Teacher.* April, 28-32.

Carmichael, H.W., Burnett, J.D., Higginson, W.C., Moore, B.G. and Pollard, P.J., (1985). *Computers, Children and Classrooms.* Toronto Ontario: Ministry of Education.

Cory, S. and Walker, M., (1985). *Logoworks: Lessons in Logo Teacher's Manual.* Cambridge, Massachusetts: Terrapin Inc.

Dahl, R., (1982). *The BFG.* Harmondsworth, U.K.: Puffin Books.

Dawson, A.J. and Bell, D., (1985). "Pathways to knowing: teachers meeting turtles" in Palmgren, M. (ed.) *Logo85 Pre-Proceedings* 87-88. Cambridge: Massachusetts Institute of Technology. .

Hoban, R., (1976). *The Mouse and His Child.* Harmondsworth, U.K.: Puffin Books.

Hoyles, C., Sutherland, R. and Evans, J., (1984). *A preliminary investigation of the pupil-centred approach to the learning of Logo in the secondary school mathematics classroom.* London, University of London Institute of Education.

Kurland D.M. and Pea, R., (1984). "Children's Mental Models of Recursive Logo Programs" *Technical Report 10, Centre for Children and Technology.* (New York), Bank Street College of Education.

Lavallade, D., (1985). "In search of recursion" in Hoyles, C. and Noss, R. (eds.) *Proceedings of the Logo and Mathematics Education Conference.* London, University of London Institute of Education.

Lawler, R.W., (1985). *Computer Experience and Cognitive Development.* Chichester, U.K.: Ellis Horwood.

Lawler, R.W., du Boulay, B., Hughes, M. and Macleod, H., (1986). *Cognition and Computers: Studies in Learning.* Chichester, U.K.: Ellis Horwood.

Leron, U., (1985). "Logo today vision and reality" *The Computing Teacher.* February, 2-32. Oregon: ICCE.

McDougall, A., (1988). *"Children, Recursion and Logo Programming"* Unpublished Ph.D. thesis, Monash University.

McDougall, A., (1989). "Teaching about recursion in Logo: A Review" In Dupe, T. (ed.) *Proceedings of the Australian Computers in Education Conference.* Canberra: Computer Education Group of the A.C.T.

McDougall, A., (1990a) "Children, recursion and Logo programming: an investigation of Papert's conjecture about the variability of Piagetian stages in computer-rich cultures" in McDougall, A. and Dowling, C. (eds.) *Computers in Education* 415-418. Amsterdam: Elsevier Science Publishers.

McDougall, A., (1990b). "Student difficulties in programming with recursion in Logo" in McDougall, A. (ed.) *Back to the Future, Forward to the Past* 108-115. Melbourne: Computer Education Group of Victoria.

McDougall, A., Adams, T. and Adams, P., (1982). *Learning Logo on the Apple II.* Sydney: Prentice-Hall.

Milne, A.A., (1926). *Winnie-the-Pooh.* London: Methuen.

Noss, R., (1985). *Creating a Mathematical Environment through Programming: A Study of Young Children Learning Logo.* London: University of London Institute of Education.

Riordon, T., (1984). Helping students with recursion: teaching strategies. *The Computing Teacher.* January, 38-40. Oregon: ICCE

Samurcay, R., (1986). "Initial representations of students in using recursive Logo procedures" in Paper presented at the *Tenth Psychology of Mathematics Education Conference,* London.

Dr. Seuss., (1958). *The Cat in the Hat Comes Back.* New York: Random House.

Sharp, P., (1984). *Turtlesteps.* Bowie, Maryland: Brady.

Watt, M., (1986a). *Welcome to Logo: Intermediate Level.* Lexington, Massachusetts: D.C. Heath and Co.

Wadtt, M., (1986b). *Welcome to Logo: Upper Level.* Lexington, Massachusetts: D.C. Heath and Co.

A Child's Progression of Developing Information Handling Techniques in Logo for Databases

Jenny Betts

Queensland Sunrise Centre

Coombabah

Introduction

At the time of writing this paper I was participating in a Sunrise Project initiated by Liddy Nevile. I spent two years exploring how Logo could be used in a primary school environment when the children had access to a portable computer of their own, 24 hours a day, 7 days a week. All school work (well, as much as possible) was performed with a computer and done in many different ways.

Reporting the progressive steps children undertook in developing a menu for a Logo database used as a part of Social Studies, is the focus of this article. It reflects 18 months of observing the children's progression of endeavouring to develop a user friendly menu which is now used across the curriculum.

It would be true to say that the development which took place was not a "true" database as it did not contain what we associate with databases, and that is fields. These databases can be more closely associated with the form of

"Choose your own path" type books. Nevertheless, even though our programming methods may seem crude to expert Logo programmers, together the children and I made some startling discoveries, especially when both the children and I were very much beginners.

The Original Plan

So often children use software that is already programmed and I almost trapped myself into believing that this was the road to take with computers in education until the day I was introduced to Logo. Commercial software packages have their value within the class environment, however, I noticed Logo provided children with another set of experiences. My experiences with Logo had been very little, in fact I had never drawn the square that everyone seemed to be able to do, due to avoidance on my part. Shape creating was not the motivating force behind why I have chosen to use Logo, although I can now see the advantages in doing so. With the introduction of LogoWriter on the market, using Logo has enabled students to create MicroWorlds by manipulating not only numbers but also text. Children have immersed themselves with programming adventure games, databases (of sorts), crosswords, word games, speed reading programs etc. and all have related to the class theme at the time. Logo has made it possible for children to be in control of learning difficult concepts while having fun.

Concentrating on the development of the programming skills is never the main focus in my class, it is developing skills and accumulating knowledge. In any event, using Logo not only allows for the development of many logical thinking processes because of the programming skills involved, but it has also motivates children, allows for children at all academic levels to achieve and they accumulate a wealth of knowledge.

Problems Faced

Because of the children's limited knowledge of programming, they faced three main problems.

1. Having easy to follow instructions
2. Having a simple screen arrangement - to include instructions. text and graphics
3. Accessing Information
   - using more than one screen
   - menus
   - searching

1. Easy to follow instructions

Preconceptions of how instructions should be written were not evident. At first the children's instructions were quite lengthy (see Figure 1). Perhaps the lengthy instructions could have related to the limited knowledge of Logo programming or limited knowledge of how computer programs worked.

```
(This is a program about Romans  and
there life style. Choose one of the
following names and type in the letter
beside it in the command centre.
To get back to the database type data

A. ARCHITECTURE
B. CLOTHING
C. RELIGION
```

Figure 1. Lengthy Instructions

Regardless, patience is the key to meaningful progression. Allowing the children to experience the lengthy instructions is very important, although at times, it is difficult not to step

in and give advice. They should be left alone to create their own instructions. As we progressed, the children began to develop procedures that they shared and refined with their friends. Discussions were held on how instructions were being worded and examples were shown.

Trying to use more simple instructions seemed to mean more to the children after they had experienced the inconvenience of more involved instructions.

2. Layout of screen

Instructions

As each screen of information was shown, it was difficult to read because the instructions (see Figure 2) and the information were cramped.

Instructions were often attached to the bottom of the print. This meant that their position depended on how much information was on the screen. Instructions therefore, were never found in the same place (see Figure 3 and Figure 4). Each time a new screen was given, you had to search for the instructions. Sometimes there were no instructions at all (see Figure 5).

Figure 2. Instructions are not clearly noticeable

Greece. To continue, Push any key.

Figure 3. Different place and different instructions

Figure 4. Clear instructions but once again not consistent with the rest of the program.

CLOTHING

The two most common dress for Greeks is either a chiton, which is a robe, for just normal dress, or armour, for guards or when citizens fight. Both men and women wore these.

You would not always see Greece people dressed in white, as you would think. They wore radiant colours when they felt like it.

Normally you would see heroes such as Jason, for example, would wear armour all the time, but gods, like Heracles, would wear a chiton.

Figure 5. No instructions at all. In fact you wait.

Not all these pages came directly after one another. Notice how the user faces several different methods of accessing information.

Having the instructions and the information together also meant that less information was able to be placed on the screen at any one time. It became apparent that it might be useful to display instructions in a manner where the user would always know exactly where to look for them and the screen would not be overcrowded with information and instructions. Using the command centre seemed a great idea and it meant only a slight change to programming. Not only did the children learn about layout of a page but they now had two more commands to work with - TYPE and SHOW. These have also been used quite extensively.

Original Version

```
to continue
 Pr [Press any key to continue.]
 name readlist "any.key
end
```

Changes

```
to continue
 Show [Press any key to continue.]
 name readlist "any.key
end
```

3. Accessing Information

(i) Using more than one screen

(a) Manual

Children were writing more than one screen of information on certain areas and found themselves having to deal with

how the user could get access to all the information. One method was:-

Print all the text on the screen and then tell the user to:

- press <control>-U to move the cursor to the text centre;
- use the arrow the keys to move through the text,;
- press <control>-D to move the cursor down to the command centre.

How we showed all the text and all these instructions was very difficult but manageable. However the problem with this method was that there was no way to return to the main menu. This was not too successful because the instructions were too long and it was too much for the user to remember.

Encouraging the children to find a way that didn't require the user to know how to use LogoWriter was my role.

(b) Waiting

Another popular method was using the wait command. Some children programmed the computer to show one screen at a time, using the wait command between screens.

e.g.

```
to egyptian
 pr [information]
 top
 wait 100
 nextscreen
 wait 100
 menu
end
```

NB: "Information" is the data which you want displayed.

Catering for the slower reader was popular. They found this annoying when they were testing their programs as they had to wait quite a while before the next screen appeared.

(c) Scrolling

Another method was to display all of the text, go to the top of the page and program the computer to scroll through the text.

```
to egyptian
 pr [Information]
 top
 repeat 34 [wait 10 cd]
 menu
end
```

NB: "Information" is the data which you want displayed. The number of repeats depends on the number of lines in the text.

(d) Controlling

Being in control of the computer rather than letting it control you has been a priority instilled in the children. When it came to programming, it was still a matter of making sure the user was in control. I suggested that they should try and program so that the user can continue when they were ready rather than having to wait until the computer is ready. Two methods were used.

(i) Continue Tool

```
to egyptian
 pr [Information]
 top
 continue
 nextscreen
 continue
 menu
end
```

Continue is a subprocedure which halts the program until a key is pressed.

```
to continue
 show "Press any key to continue.
 name readchar "any.key
end
```

(ii) Arrow Tool

```
to arrows
 pr [information]
 top
 show "Use ↓ and ↑ to view text
 show "Press ESC to exit
 pause
end
```

Pause is a subprocedure which looks for the arrow and ESC key to be pressed.

```
to pause
 name readchar "cursor
 if :cursor = char 27 [menu stop]
 if :cursor = "H [cu]
 if :cursor = "P [cd]
 pause
end
```

To find the correct ascii characters for key type "show ascii readchar". Press enter. The computer is now waiting for you to press the key you wish to find the ascii character for. In this case press ESC and the computer will respond with the number 27.

To find the symbol for a character type "show readchar" and press "enter". The computer is now waiting for you to

press the key you wish to find the symbol for. In this case press one of the arrow keys. The computer will respond to this by showing a space and the capital "H". You must remember that this is a symbol and not a letter.

(ii) Menu

(a) Text Menu

Looking back it seems a minor hurdle but creating a menu was our biggest breakthrough. Ideas for these procedures were taken from students, books, the manual and visiting experts. Any new ideas were quickly seized and modified to suit our purposes. Writing procedures that would allow the user to select from a menu was not difficult but our knowledge was very limited. We all started with the simplest methods we knew and that was:

```
to startup
 rg ht ct
 pr [Do you want]
 pr [ ]
 pr [a. Architecture]
 pr [ ]
 pr [b. Lifestyle and Furniture]
 pr [ ]
 pr [c. Religion]
end

to a
 gp "egypt
end

to b
 gp "egypt1
end
```

```
to c
 gp "egypt2
end
```

This worked quite well until we found, I should say a student found, a tool that would be extremely useful for many things other than the uses we now needed. It was:

```
to menu
 pr [Choose 1 - 3]
 pr [ ]
 pr [1. Egypt]
 pr [2. Greece]
 pr [3. Rome]
 choose
end
```

```
to choose
 name readchar "choice
 if :choice = 1 [egypt]
 if :choice = 2 [Greece]
 if :choice = 3 [Rome]
end
```

This was a first major breakthrough. This toolhas been used in many of their programs and appears in many different forms. This method has allowed us to use very simple wording for instructions. For the user, the selection process is quite simple.

(b) Turtle Menu using POS

This menu (see Figure 6) was developed when the children used a commercial game where the user used the arrow keys to place an arrow next to the requested topic and then pressed enter.

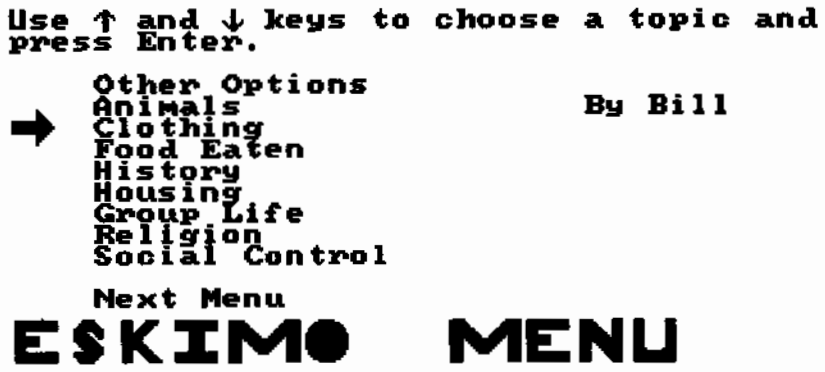


Figure 6

The procedures are:-

```
to menu
 rg ct ht cc
 loadtext "menu.txt
 loadpic "menu.pic
 setsh 1 st seth 0
 pu setpos[-150 40]
 choose
end
```

This procedure loads the graphics and the text. It sets the scene which is seen in Figure 5. The turtle is given its shape, pointed in the right direction and placed in the position near the first item, "other options".

```
to choose
 arrow
 middle?
 choose
end
```

The choose subprocedure makes the whole menu procedure run.

```
to arrow
 name readchar "up.down
 if :up.down = char 13 [position?]
 if :up.down = "H [fd 10]
 if :up.down = "P [bk 10]
 arrow
end
```

This procedure checks to see if the up or down arrow is pressed. If so, then the turtle will move in the correct direction ie if the up arrow key is pressed the turtle will move forward 10. Finding the symbol for a keyboard character and ascii character numbers is explained in the arrow tool.

```
to middle?
 if pos = [-150 50] [bk 10]
 if pos = [-150 -70] [fd 10]
end
```

This subprocedure checks to see that the turtle does not move above or below the top and bottom topics.

```
to position?
 if pos = [-150 40] [other.options]
 if pos = [-150 30] [animal.menu]
 if pos = [-150 20] [clothing.menu]
 if pos = [-150 10] [food.eaten]
 if pos = [-150 0]  [history.menu]
 if pos = [-150 -10] [housing.menu]
 if pos = [-150 -20] [group.life]
 if pos = [-150 -30] [religion.menu]
 if pos = [-150 -40] [social.control]
 if pos = [-150 -60] [next.menu]
end
```

When the enter key is pressed, the computer will check to see the turtle's position. If this is a position the turtle has been programmed to respond to the computer will run that subprocedure.

(c) Turtle Menu using CHARUNDER

This tool was invented by a group of 12 year old students. It is the most recent discovery and was developed when the children used a commercial database called PC Globe.

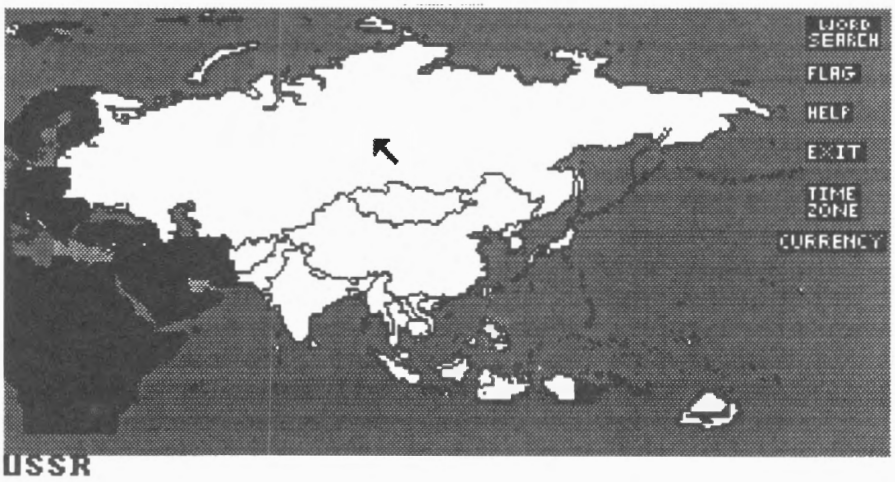


Figure 7. Map uploaded from PC Globe

It is very similar to the way the PC Globe map works. This graphic has been loaded over the top of a text file which has been loaded in the colour 0 so that it cannot be seen by the user. Each country or word from the menu has been assigned a character (see Figure 7). As you move the turtle around the screen, the turtle checks to see which character it is under. It will then show in the command centre the appropriate country.

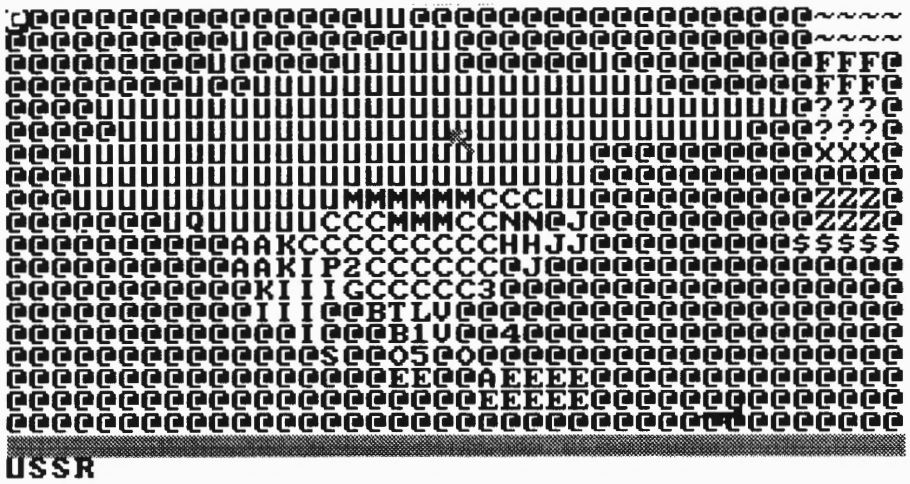


Figure 8. Text File which is loaded
underneath the graphic

If you look closely (see Figure 8) you can see how each country and menu word has been assigned a character.

The procedures for this are:-

```
line 1     to main.menu
line 2     rg cc ct pu
line 3     setsh 19
line 4     settc 0
line 5     loadtext "0.txt
line 6     settc 0
line 7     loadtext "asia.txt
line 8     top delete
line 9     loadpic "asia.pic
line 10    make "country "China
line 11    type :country
line 12    make "color 1
line 13    tell 1 pu
line 14    setpos [95 -88]
line 15    setsh 22 st
line 16    tell 0 cc
line 17    move.menu
line 18    end
```

Loading "0.txt" in line 5, using colour 0 has to be done first so that the computer will then load the next lot of text in the same colour. Colour 0 has been chosen because it is clear on the monochrome screen. "0.txt" contains only a "return" and is deleted in line 8 , after the "asia.txt" is loaded in line 7. The asia.txt is what you see in figure 3. Line 10 creates a variable, country which will change according to which character is under the turtle at the time. (see procedure "fly".) The colour is then changed back to 1 so the user can see what is written. Line 14 sets the shape of the turtle and in this case it is an arrow, and the position, which is on China. We then move onto the next procedure "move.menu" which allows the user to move the turtle around the screen

```
to move.menu
 if key? [menu!]
 setc :color
 make "color :color + 1
 repeat 5 [if key? [menu!] wait 1]
 if :color = 5 [make "color 1]
 cc
 type :country
 move.menu
end
```

The students programmed the turtle to change colour while it is not actually moving around the screen. In line 1 if a key is pressed, the computer will move onto the next procedure which is "Menu!"

```
line 1      to menu!
line 2      cc type :country
line 3      name readchar "move
line 4      if :move = char 27 [make "country
            "Exit]
```

```
line 5      if :move = "H [seth 0 fd 5 under]
line 6      if :move = "P [seth 180
            fd 5 under]
line 7      if :move = "M [seth 90
            fd 5  under]
line 8      if :move = "K [seth 270
            fd 5 under]
line 9      if :move = "_ [back.menu]
line 10     if :move = char 13 [enter.menu]
line 11     if not letter? :move [stop]
line 12     make "country (word :country
            :move)
line 13     cc type :country
line 14     before?
line 15      end
```

This procedure checks for particular keys to be pressed.

line 4 ESC key
line 5 up arrow key
line 6 down arrow key
line 7 right arrow key
line 8 left arrow key
line 9 Backspace
line 10 Enter

line 13 Depending upon the character under the turtle, a country will be shown in the command centre and then the computer will wait for another key to be pressed.

```
to letter? :letter
 op member? :letter[a b c d e f g h i j k l
m n o p q r s t u v w x y z | | ]
end
```

If the user chooses to type the choice in the command centre, the computer will output the key pressed and show it in the command centre.

```
to back.menu
 if empty? :country [stop]
 cc
 make "country bl :country
 type :country
end
```

The "Back.menu" is included because the user can also type in the command centre which country or action from the menu they wish to pursue. The boys have included a back-space so if a mistake is made when typing, the user can use the backspace to delete the mistakes.

```
to under
 if charunder = "$ [make "country "Currency]
 if charunder = "u [make "country "USSR]
 if charunder = "m [make "country "Mongolia]
 if charunder = "c [make "country "China]
 if charunder = "z [make "country "|Time Zone| ]
 if charunder = "e [make "country "Indonesia]
 if charunder = "s [make "country "|Sri Lanka|]
 if charunder = "t [make "country "Thailand]
 if charunder = "l [make "Country "Laos]
 if charunder = "v [make "country "Vietnam]
 if charunder = "p [make "country "Nepal]
 if charunder = "b [make "country "Burma]
 if charunder = "a [make "country  "Afghanistan]
 if charunder = "k [make "country "Pakistan]
 if charunder = "@ [make "country "]
 if charunder = "h [make "country "|South Korea|]
 if charunder = "1 [make "country "Khmer]
 if charunder = "n [make "country "|North Korea|]
```

```
 if charunder = "i [make "country "India]
 if charunder = "j [make "Country "Japan]
 if charunder = "~ [make "country "|Word Search|]
 if charunder = "? [make "country "Help]
 if charunder = "x [make "country "Exit]
 if charunder = "f [make "country "Flag]
 if charunder = "o [make "country "Malaysia]
 if charunder = "g [make "country "Bangladesh]
 if charunder = "2 [make "country "Bhutan]
 if charunder = "3 [make "Country "Taiwan]
 if charunder = "4 [make "country "Phillippines]
 if charunder = "5 [make "country "Singapore]
end
```

The "menu" procedure assigns "country a character. When the turtle is under a letter it recognises it will then make the variable "country the appropriate name.

line 1
```
to enter.menu
```

line 2
```
if member? :country [Exit Flag Help
|Word Search| |Time Zone|
currency] [run (se :country ")]
```

line 3
```
if member? :country [Singapore Phil-
ippines Taiwan Bhutan Bangladesh
Japan India |North Korea| Khmer
|South Korea| Pakistan Afghanistan
Burma Nepal Vietnam Laos Thailand
|Sri Lanka| Malaysia Indonesia China
Mongolia USSR] [fly start.get.info
:country stop]
end
```

The "enter.menu" will be invoked when the enter key is pressed. The computer will check which country is the "country variable, check to see if it is a member of the list of countries available. There are two lists here. Line 2 lists

the menu type actions such as Exit, Flags, Help, Word search, time zones and currency. If the character is matched to one of these it will invoke the procedures which work that part of the program. Line 3 checks to see if you have the turtle over a specific country. If it is then the computer will search for the information on that country and display it.

```
to fly
 if :country = "USSR [if not charunder
 = "u[setpos[-15 40]]]
 if :country = "Mongolia [if not
 charunder = "m[setpos[-15 10]]]
 if :country = "China [if not charunder =
 "c[setpos[-15 -15]]]
 if :country = "Indonesia [if not charunder
 = "e[setpos[20 -75]]]
 if :country = "|Sri Lanka| [if not
 charunder = "s[setpos[-45 -55]]]
 if :country = "Thailand [if not charunder
 = "t[setpos[-15 -45]]]
 if :country = "Laos [if not charunder =
  "l[setpos[-13 -40]]]
 if :country = "Vietnam [if not charunder =
 "v[setpos[-10 -37]]]
 if :country = "Nepal [if not charunder =
 "p[setpos[-45 -26]]]
 if :country = "Burma [if not charunder =
 "b[setpos[-25 -35]]]
 if :country = "Afghanistan [if not
 charunder = "a[setpos[-70 -15]]]
 if :country = "Pakistan [if not charunder
 = "k[setpos[-60 -20]]]
 if :country = "|South Korea| [if not
 charunder = "h[setpos[26 -10]]]
```

```
 if :country = "Khmer [if not charunder =
 "1[setpos[-9 -50]]]
 if :country = "|North Korea| [if not
 charunder = "n[setpos[25 -5]]]
 if :country = "India [if not charunder =
 "i[setpos[-55 -30]]]
 if :country = "Japan [if not charunder =
 "j[setpos[45 -10]]]
 if :country = "Malaysia [if not charunder
 = "o[setpos[4 -65]]]
 if :country = "Bangladesh [if not
 charunder = "g[setpos[-33 -34]]]
 if :country = "Bhutan [if not charunder =
 "2[setpos[-33 -29]]]
 if :country = "Taiwan [if not charunder =
 "3[setpos[17 -34]]]
 if :country = "Phillippines [if not
 charunder = "4[setpos[17 -49]]]
 if :country = "Singapore [if not
 charunder = "5[setpos[-15 -60]]]
 make "pos pos
 tell 1
 seth towards :pos
 repeat distance :pos [fd 1 if key?
 [stop] wait 1]
 tone 1000 5
end
```

This is a little extra animation. When a country is selected, a plane flies to that country.

Summary

Most problems encountered, such us lengthy instructions, crowded screens, difficulties with selecting topics and difficulties with accessing information, have all been overcome by improving our programming skills.

You can give children 101 tools with which to work before you start programming but they will not know the true usefulness of them until they have the need to use something similar. They then begin to see the need to enhance their ideas. The key is to be patient and introduce each tool as it is needed or discovered by the children. It seemed that, when the children were given a few tools with which to work, they seemed to work harder. We were always searching for solutions to problems, and I think this is what made programming databases exciting.

Developing programming skills is not the main focus, nor is it the developing of a database. It is the actual research skills and the knowledge gained as the children develop the other two areas.

We have come a long way from the first attempts of programming. Although we still have not reached a "true" database form, the most important element is that the children have gained a great deal of knowledge, developed academic skills (such as researching, note taking and communicative writing skills) acquired social skills and developed thinking processes, all of which are objectives of the current Queensland Social Studies Syllabus.

Logo in Preservice and Inservice Teacher Education

R. A. Schibeci

School of Education,
Murdoch University
Murdoch 6150 W.A.

(An earlier version of this paper was presented at the Australian Computers in Education Conference, September 26-28, 1988)

Logo continues to generate fierce passions among members of the educational computing community: see, for example, the debate in the pages of *Educational Researcher*: [1] to [4]. Much of the debate has revolved around the use of Logo with children; much less is known about the use of Logo and adults.

Teachers and trainee teachers constitute important adult groups. Computers have been used in teacher education in many different ways (du Boulay [5]; Wood [6]). A few studies have reported investigations in which Logo has been used in teacher education.

As with children, Logo can be used with adults in one of two ways: in an unstructured way or in a formal, structured way. The former approach is exemplified by an early study reported by Austin [7] from the MIT AI laboratory: that

study was guided by the question, "What are the problems involved in teaching adult teacher trainees the ideas embodied in the current LOGO curriculum (which was originally designed with children in mind) ... ?" (p. 2). The second (structured) approached is exemplified by studies such as those by Battista [8] and by du Boulay and Howe [9]; in these cases, attempts were make to teach specific mathematical concepts. The approach taken in this study was the less structured one. This paper describes the use of a Logo programming project with four groups of students in three different teacher education courses: one group in each of two pre-service courses ("Cultural Mathematics" and "Computers in the classroom") and two groups in an in-service course in two consecutive years ("Computers in the school curriculum").

The Study: The Rationale and the Students

The rationale

The project began when an application was made to the University for support for a study entitled "The development of informed numeracy". One strand of this project was the use of Logo with adult learners - students in a primary teacher education programme. In the application for funds, the following rationale was given.

The study was guided by a desire to investigate ways in which adults could be helped to reflect critically on themselves as *learners*. More specifically, it was designed to investigate this possibility in the context of learning mathematics. Logo was included as part of the overall project because of its treatment of geometric ideas.

We felt that much traditional CAI (computer-assisted in-

struction) is based explicitly or implicitly on a programmed instruction model of learning in which the amount of learner control is very limited. This model does not encourage students to view themselves as active learners; in particular, it does not view errors as a very useful mechanism for learning.

Logo, however encourages an active learning view – at least if used in the unstructured way outlined earlier. In particular, errors are viewed as a very valuable learning mechanism. We were therefore interested in exploring the consequences of the regular and systematic use of Logo by a group of adult learners. Our concerns included: the extent to which this Logo experience changed students' views of themselves as learners, and the extent of student development of problem solving strategies.

The students

The preservice course *Cultural Mathematics* is a compulsory course for students in the three-year primary teacher education programme which leads to the B.A. (Teacher Education) at Murdoch University. In the 1986 version of the course, students were offered a choice of projects. Among the choices was an introduction to the Logo language (Logotron Logo for the BBC Acorn microcomputer). A set of summary sheets, which introduced students to some of the major features of Logo, was based on the Logotron manual [10]. Four BBC microcomputers were available, each with the Logotron chip. Each machine had at least a single disk drive for storing student's procedures.

The project was presented as an independent project. That is, students were to work through the summary sheets provided, and note down any points of interest and difficulty in a "log". About half-way through the sheets, stu-

dents were asked to write a procedure for one of two drawing exercises which were provided as a check on the work they had done to date. As a final requirement, they were asked to summarise their reactions to the Logo project. Students (there were eighteen in all) were expected to spend between 10 and 20 hours on the project.

The course itself, *Cultural Mathematics*, was designed to help students explore the nature of mathematics through a study of a set of mathematical concepts. Unlike standard mathematics courses, it did not emphasise the development of algorithmic procedures. This emphasis, in our view, made the Logo project consistent with the general aim of the course. We were interested in student exploration of mathematical ideas embodied in the Logo language, rather than to develop algorithms to solve particular mathematical problems. The course comprised 26 hours of contact (lectures/seminars/tutorials) together with the Logo project, which was to be done at times convenient to the student. Very little time was devoted to a discussion of Logo in the normal contact time of 26 hours: the exploration of specific mathematical ideas in Logo was confined to the time devoted to work on the Logo project. The students in this course were the first to undertake the Logo project. Subsequently, the Logo project was incorporated into the other two courses.

One of these was the B. Ed. (inservice) course, *Computers in the School Curriculum.* This course is an elective within the B.Ed. programme. The overall purpose of this course is to examine critically the use of computers in schools. Students are expected to analyse issues raised by the use of computers in schools and in society generally. Assumptions underlying many of the beliefs about the benefits of computers are expected to be analysed thoroughly. The course consists of 39 hours of contact devoted to seminars and

tutorials. In addition, students are required to do a project, and have been given the option of completing an introduction to the Logo programming language. In 1986, 14 of the 16 students chose to do the Logo project; 12 were teachers and the remainder pre-service students. A second group from the 1987 offering of the course, is included in this study. In this case, there were 10 students: 6 teachers, 4 pre-service students.

The fourth adult group included in this study was another pre-service group. The pre-service course *Computers in the Classroom* was an elective offered for the first time in 1987 for students in a pre-service teacher education (primary or secondary) programme. The emphasis in this course was to give students a user's perspective on using computers in classrooms. That is, the assumption was that the student's concern was to be a primary teacher, or a secondary teacher of a particular area rather than to be specialist computing teachers. As such, the course was designed to help students develop views about what were sensible uses of computers in helping their students learn ... teaching *with* computers was the emphasis. There were 20 one-hour lectures in the course, together with two major practical activities to be done in the student's own time: the Logo project, and learning to use a word processing program to produce their major essay for the course. Students were required to do the Logo project unless they wanted to learn Pascal (for teaching specialist computing courses). As before, students were required to keep a "log" of their activities during the Logo project. There were 21 pre-service students in this course.

There were thus four groups of adults in all: one group in each of two pre-service courses, and two groups in consecutive offerings of an in-service course.

Data collection

Student reactions to the Logo project are based mainly on the "log" they were required to keep. The diary was designed to be a record of their reactions as they worked through the Logo sheets. These data were supplemented by informal observations and discussion with the students.

Selected Student Reactions to the Logo Project

Reactions to the Logo programming experience varied, and appeared to be unrelated to whether the student was in a pre-service or in-service teacher education programme. For this reason, no distinction will be made among the four groups of students in the three courses described above. Any differences among students are more likely to be linked to prior experience with computers, rather than their status as pre- or in-service students. It may be that in-service students would, in time, see the potential applications of Logo to their classrooms. However, these views were not expressed during the course: presumably, the students were too busy coming to grips with the 'nuts and bolts' aspects of the Logo project.

Logo and computers

One mature age student, who wanted to be a primary teacher, observed in the first week of the project:

I am particularly neurotic about even touching a computer!

Other students may not have had views as extreme as this one, but there was a general reluctance to work with the computer. Many felt they were doing this "for their own good". Most students, however, changed their views about computers as a result of their experience. For example, one student wrote:

Logo was not an incredibly difficult language to learn. That it uses non-threatening language and admits to being wrong occasionally (it isn't always the programmer's fault) is an asset ... Positive feedback is presented to the programmer constantly. Any feeling of despair or disillusionment can be rapidly quashed by new achievements ...

The potential for using Logo in the classroom is great - something I never would have believed before I started this.

Another student wrote:

I have had only a little experience with computers and that was enough to put me off forever. I did a computer course as an option in Year 10 at High School over five years ago. The course was a total disaster ... I had no idea how to control the computer, and no idea of how it was ingesting and using any information I happened to feed in. The course produced and confirmed the idea that computers were very difficult to use and impossible to understand. I came away worried because I couldn't understand and felt I needed to, and also with a fairly deep hatred of computers (probably because I couldn't use them).

... after the first two or three weeks [of the Logo project] I felt really elated and proud of myself as I was not only able to control it but I understood how it worked, how it used any commands and why our patterns resulted as they did. This feeling persisted for almost the entire course, any unexpected problems that arose we were able to discuss and always (nearly) found the solution...

> *The course has dispelled my previous fear and hatred of computers to an extent that I want one of my own.*

The reactions were not all completely positive, however. One of the few exceptions was the student who wrote:

> *Basically, the amount I have gained from this project, is not worth the amount of time that I have put into it. Although I have become quite confident with the computer, I think I have also become very cynical about the whole project, instead of enjoying LOGO.*
>
> *At first, it was really interesting and enjoyable, however, as the sessions went on, we seemed to be spending more time correcting spacing and printing errors than we were learning new procedures.*

Part of the frustration expressed here was a consequence of the printing errors in the summary sheets (based on the Logotron manual) I had produced. These proved to be particularly irksome for those who were conscientiously trying to grapple with new concepts.

Logo and mathematics

Although the mathematical concepts were stressed only in the preservice "Cultural Mathematics" course, many students commented on the mathematical aspects of Logo. Some specific instances of difficulties with mathematical concepts arose. For example, one student (who wants to be a high school social studies teacher, and was enrolled in the preservice "Computers in the classroom" course) commented on the difficulties she had with the 'SETPOS' command:

The notion that no maths is involved in computers is not true. O. K. it's limited maths but I spent hours trying to figure out how to create a triangle simply because I didn't remember the formulas or how many degrees in a triangle - basic stuff that [my partner] had to help me with - it's been 13 years since I last did maths ... - oh yes - Setpos required a knowledge of coordinates which I didn't have - Sure I worked it out eventually but it took me 2 hours to others' ten minutes - ... Some things I still haven't quite grasped ...(-but basically I think I've learned enough about Logo and computers to deal well with such issues when time is less pressing/precious!).

A mature age student wrote, in week three of the project, a procedure for BOAT as follows:

```
BOAT
 RT 150
 FD 150
 LT 60
 FD 200
 LT 60
 FD 150
 LT 120
 FD 350
 RT 90
 FD 100
 REPEAT 3 [RT 90 FD 50]
```

Something like this may be pathetic to the computer literate (what a term - with no definite connotation) but to me it is wonderful! Rolf Harris eat your heart out.

General Student Reactions to the Logo Project

The selected student reactions highlighted above are some of the more colourful views expressed. The important issue, however, is the effect of the project on the larger student group.

Logo and computers

Most students having their first experience with a computer reported initial frustration. The frustration arose mainly from not knowing enough about the machine to correct mistakes easily and being largely unaware of, and apprehensive about, the consequences of any action taken to overcome the problem. For this reason, when difficulties occurred, the reactions ranged from a rush to turn the machine off to a feeling of sheer panic. It soon became evident to them that human error caused the problems and that the computer merely responded to instructions. Thereafter, most students adopted a systematic approach when faced with these frustrations. That is, rather than seeking a means of simply erasing typing errors (turning the machine off or typing the commands in again), a systematic process of de-bugging was adopted.

The challenge of working with a computer proved a stimulating one for most of the students, and, generally, they enjoyed the challenge. Several reported increased confidence in using a computer, not only for Logo but for other purposes. It would appear that as their ability to use the computer increased, so did their confidence in experimenting with the machine in an effort to solve problems.

Several students complained of a lack of keyboard skills which led to slowness when typing in procedures, and a profusion of purely typographical errors necessitating tedious correction. A sense of embarrassment about this lack of keyboard skills caused some students to avoid the

typing role and leave the typing in of commands to their partners who felt more comfortable with the keyboard. Among this group of students there was a consensus of opinion that some keyboard competence is essential before attempting computer operations which require lengthy typing in of procedures and commands.

While working on the Logo project, several students claimed the acquisition of skills in computer operation in general, especially an understanding of the processes involved in loading files onto disc, and retrieving these files from disc.

Once a few operational problems had been overcome (for example: the functions of particular keys such as the "Shift" key and the necessity for commands to be in upper case), students felt in control of the situation. Several commented that they were really excited when "something happened" (graphics appeared on the screen), and, overall, students were happy with their first experience of Logo. All found it absorbing, interesting and enjoyable and commented that the time went quickly. All students claimed satisfaction at being able to produce something and being particularly pleased because they had not thought they would be able to do it at the outset.

Some students *did* experience frustration with the computer when problems occurred and didn't know how to solve them. For most, however, working with computers was a challenge which they enjoyed. Computers were new to them and very exciting. One student commented that there were only two computers in the school she had attended and only "Computer Studies" students drawn from the "Advanced" mathematics class were allowed to use them. This tended to give the impression that computing is associated with mathematics and those who were not particularly

strong in mathematics kept away from computing and often developed a fear of it. She further commented that until undertaking this particular Logo Project she had no idea that computers could do interesting things like drawing circles in colour on a coloured screen. Several students indicated that they ran out of time in their sessions to experiment with all the things which interested them and, therefore, they would have liked to spend more time on Logo but could not do so because of other commitments.

Logo and control of learning

Initially, many students felt that the instruction sheets they were given and the computer were in control of the situation. However, they came to believe that the language enabled the building up of quite sophisticated programmes from simple procedures.

For the majority of the students, the frustrations encountered in the process of experimentation and discovery were more than adequately compensated for by the sense of achievement when a seemingly-difficult procedure produced the desired result. For some, this was a feeling of elation and a pride in ownership of the finished product, but, overall, the experience of success produced a sensation of being in control of the learning situation.

According to the students, Logo was a friendly language because every command does exactly what it says it will: if it says FORWARD, it goes forward and so "you can't go far wrong." In essence, students felt that the machine was doing what they told it. Replies from the machine like "I DON'T KNOW HOW TO...." gave the user a feeling of knowing what is being done and it is the computer which does not. One student said, "Even though you know all the

time who makes the mistake, it makes you feel better about making mistakes. It puts up messages letting you know where you went wrong without being critical."

In one or two cases, even after experimentation and experience, the feeling of no real control over the situation persisted. However, the majority came to believe that Logo was a programming language for achieving some spectacular and satisfying results with a few simple commands.

Logo and errors

The fear of making mistakes appeared to give way to an eagerness to test hypotheses and experiment with features of Logo. There was abundant evidence that, for most of the learners, fear of making "mistakes" lessened with time.

Many students have expressed the opinion that Logo encourages persistence and continual testing and modification of ideas because it offers the scope to experiment and discover with the fear of making errors removed. Most indicated that they now regard their earlier concern about generating errors which would cause the breakdown of Logo or the machine as completely groundless. All of the students in one way or another indicated that progressing through the sheets, from their earliest efforts when they had no Logo or computer experience at all to the stage where they were able to produce displays from recursive procedures, has given them a tremendous feeling of achievement and greatly enhanced self-esteem.

Logo and social learning

On the issue of working in pairs or individually, opinions varied. One student commented that she believed working in pairs to be absolutely essential. One person can write, the other can type and both can observe what is taking

place. However, in this case the student had spent some considerable time before her partner arrived trying to work out SETH and SETPOS ... to no avail. When her partner arrived, they discussed it briefly; the partner suggested another solution and the problem was quickly solved. Conversely, another student whose partner failed to appear throughout the project expressed the opinion that he felt he had gained a great deal from the project by working on his own because he had no-one else to rely on and had to discover things for himself.

Working in pairs did not suit everyone. For example, in one case one student wished to persist with problems until they were solved, whilst the partner desired to abandon it and go on to something else. Frustrations occurred for the student who grasped concepts more quickly than the partner and then had to wait for the partner to catch up, but equally the student who worked more slowly experienced frustration because of the quicker partner wishing to speed ahead. On the whole, students were able to identify the differences and for the most part recognised that some tolerance was needed, but it served to highlight the fact that individuals work at varying rates, learn at varying rates and react differently to specific situations. It may be that students in pairs which were not successful may have been helped by a different partner; this would be an interesting issue to explore.

Summary

The change in attitude of the students to Logo and to computers over the period of the project was very marked. As they became more familiar with Logo commands, their confidence in using it increased. The most marked difference, however, was in their attitude to computers them-

selves. For these students, computers were no longer awe-inspiring or solely the domain of "mathematical people": computers still presented a challenge, but a challenge to be enjoyed.

My strong impression was that students generally felt that Logo provided the basis for the creation of a learning environment in a large number of subject areas in that it develops thinking. Being essentially a problem–solving exercise, the Logo project encouraged many of the qualities which go hand in hand with thinking and problem–solving, namely logical progression, recall, postulating, trying hypotheses and seeing relationships. Other learning outcomes identified were persistence in response to frustration and the need to plan ahead.

Learning in the personal-social sphere emerged as an outcome of programming in Logo. Most students felt that success with Logo was based on a measure of independence and, therefore, the opportunity to solve one's own problems and be in control of the learning, was conducive to a sense of achievement. An extension of the sense of achievement is the development of self-confidence and the perception of self as an intellectual agent.

Logo produced in most of the students a changed attitude to errors in that feedback to the effect that an error had occurred no longer produced a sense of failure but a determination to solve the problem by experimentation, testing of hypotheses and restructuring of the procedure. The positive feedback provided by Logo in turn develops a positive attitude to errors and led to the asking of questions about difficulties encountered. Questions like, "Why did that fail?" and "How can it be made to work?" led to persistence in the face of failure and frustration rather than

a sense of embarrassment and a reluctance to try again for fear of further embarrassment.

Logo in Teacher Education

What conclusions can be drawn from the experiences of the four adult groups in the Logo project? Burnett [11] claimed that teachers who learn Logo "learn something about computers and the Logo language, and they learn something about education" (p. 79). It is certainly true in this project that they learned, in a general way, how to operate a computer; this, of course, is not unique to a Logo experience. The adults also learned about programming a computer; again, this is not unique to a Logo experience.

What did they learn about education? In Burnett's words, Logo "permits highly anxious and insecure adults to cross over into the realm of computer comfort with relatively little difficulty" (p. 80). this is a very valuable feature of a Logo experience, especially for the mature novice. Many of the teachers are, of course, in a "novice" position — just as their students often are when learning new material. Hopefully, the teachers' experiences as novices will help them to be more sympathetic to their students' difficulties than they might otherwise have been. It is not possible to collect evidence in this study on this important issue; certainly, it is an issue which is worth investigating.

Some professional educators will, with the best of intentions, attempt to "package" Logo into a well-structured curriculum with carefully graded exercises and objectives. Such an approach will help students learn the Logo language more efficiently, but will the opportunity to *learn how to learn* be lost ?

The results of this study *do* suggest positive outcomes from the Logo project with adults. It's useful to remember what

Austin ([7] p.24) wrote at the conclusion of the report of the MIT AI laboratory cited at the beginning of this paper.

Let me conclude this article with the confession that I, like many others who have survived an educational institution, once believed that much of the chaos that is school today, stems directly from the "intellectual inadequacies" of the teachers employed therein. Happily this proved to be a thoroughly misguided notion. *Teachers are not dumb!* Rather they, like so often has been the case even in scientific endeavours, have been labouring very diligently to find answers to the *wrong questions* (supplied, of course, by their leaders). Given the right preparation, i.e. the right set of questions, most teachers are capable of truly exciting, creative intellectual activity.

Hopefully, we are beginning to ask the right questions!

References

[1] Becker, H.J. (1987) *Educational Researcher*, *16*(5), 11.

[2] Papert, S. (1987) *Educational Researcher*, *16*(1), 22.

[3] Pea, R.D. (1987) *Educational Researcher*, *16*(5), 4 .

[4] Walker, D.F. (1987) *Educational Researcher*, *16*(5), 9 .

[5] du Boulay, J.B.H. (1987) Computers and teacher education. In *Educational Computing* (Edited by Scanlon, E. and O'Shea, T.). Wiley/Open University

[6] Wood, S. (1985) Expert systems in teacher education. In *Artificial Intelligence and Education.* Proceedings of the 2nd International Conference on AI and Education , University of Exeter .

[7] Austin, H. (1976) *Teaching teachers Logo: The Lesley experiments* (Logo Memo AIM 336). AI Laboratory: MIT .

[8] Battista, M.T. (1987) *School Science and Mathematics*, *87*, 286.

[9] du Boulay, J.B.H. & Howe, J.A.M. (1982) *Computers & Education*, *6*, 93 .

[10] Logotron. (1984) *Logo: A new approach to educational and recreational computing*. Logotron, London.

[11] Burnett, D. (1984) Logo for teacher education. In *New Horizons In Educational Computing* (Edited by Yazdani, M.) Ellis Horwood.

Procedurality, Modularity and Problem Structure: Introducing Logo Through Music

Pam Gibbons

Catholic College of Education

Sydney

Introduction

Although the procedural nature of the Logo language is capable of enabling learners to think in a modular, structured manner, it cannot be assumed that this will happen as a matter of course. This belief has grown initially from casual observations in my own classroom and has since been supported by some investigative action research on my part to consider the phenomenon. It is proposed that learners' early experiences with Logo are critical in developing their ability and willingness to powerfully use sub-procedures as a problem solving strategy. This paper aims to:

a) identify the problem;

b) consider possible causes of the problem;

c) suggest an alternative method of introducing Logo to learners which might overcome the problem (merely foreshadowing the action research which will be discussed during the session time);

d) suggest future directions for research based on these preliminary findings.

The session time will be used to detail the actions and results of my own investigations into introducing Logo through its music facility. This will include:

a) a thorough description of the three introductory Logo lessons given to Year 7 students using music; and

b) some examples of the structure of their programs when they later moved onto turtle graphics problems.

The Problem: Resistance to Using Sub-Procedures

Logo's primary raison d'etre has long been touted by its supporters as its ability to develop thinking skills. Papert's early claim that Logo provided "*a tool to think with*" (Papert, 1980) has since been enthusiastically echoed by other like-minded educators. Dale claims that "Logo's interactive and procedural structure makes the (problem solving) process systematic and thus easier to understand. Using proceduralisation to write and follow programs also helps to develop the ability to see relationships....Logo's procedural structure can develop planning and organisational skills that use inductive and deductive thinking" (Dale, 1984, p.175).

Moreover, the development of a structured, modular approach to problem solving as a worthwhile objective in the teaching of thinking skills has been strongly supported by the educational psychology fraternity. In Bloom's taxonomy, the higher order thinking skill of analysis is defined as "...the breakdown of the material into its constituent parts and detection of the relationships of the parts and of the way they are organised" (Bloom, 1956, p.144). Of significance in justifying the use of Logo is Splitter's analy-

sis of the basis of thinking skills: "...connecting one's thoughts into organised and coherent structures, and doing so in a self-corrective and reflective manner, are the keys to building thinking and reasoning skills" (Splitter, 1988, p.40).

Despite the widespread acceptance of organised thinking and reasoning skills as worthwhile educational objectives and of Logo as an acceptable tool to pursue these objectives, there is often a gap between what Logo has promised and what has been delivered. The promise seems straightforward, as succinctly stated by James: "The proper use of Logo encourages breaking a problem into smaller manageable parts" (James, 1986, p.192). However learners' instinctive use of such "proper" and sensible structures is perhaps too naively assumed by educators. Dale states that "Of course, Logo commands can be written sequentially, but many children quickly learn that long, sequential programs without line numbers are very hard to fix...using procedural modules saves time...Seeing direct and practical reasons for analysing a problem lets children experience the value of this kind of thinking... It is a fairly natural, and not too difficult, step to apply the analytical knowledge of a program's structure to the creation of a new program (Dale, 1984, p.174).

It is a pity that such optimism is not supported by research. McMillan states that "There is a belief that working with Logo advances formal operational thinking... (but) there is little evidence to support the belief" (McMillan, 1987, p.182). Central to the concern of my paper is the phenomenon identified by McDougall, that "Ideas of procedurality, modularity and problem structure appear to be difficult for many children to grasp and use with power" (McDougall, 1986, p.53). A case study reported by McDougall captured my attention because it typified the attitude of many of my

own students towards problem solving in Logo. She writes: "One of the children ... has shown an approach to sub-procedures which is strongly affected by her tendency to see 'whole' designs, concentrating on outlines....This child was generally competent at writing procedures and in the syntactically correct use of sub-procedures. However at times she found difficulty in seeing the smaller component parts of a design, and as a result patterns which might have been easily executed became extraordinarily difficult for her" (McDougall, 1986, p.54). The mere procedurality of the Logo language does not mean that learners will automatically understand, prefer or utilise a modular approach to problem solving.

We should not be surprised by this: the problem was foreshadowed by Papert (1980) who described the resistance of a particular student to using sub-procedures in his programs. Perhaps we were mislead by Papert's confidence that most learners would eventually come to a personal recognition and self-correction of their natural, convoluted approach to programming. It is herein proposed that the success of Logo in developing specific thinking skills (eg structure, organisation, constituency) rests delicately with the conscious, deliberate and variable efforts of teachers to teach problem solving skills.

Why Do Learners Resist Using Sub-Procedures?

As has previously been stated, Papert himself encountered resistance to using sub-procedures amongst his own students. This he dismissed as a consequence of its being students' first encounter with this type of problem solving strategy: "Keith...had been exposed to the idea of using sub-procedures but had previously resisted it. The 'straight-line' form of programming corresponded more closely to

his familiar way of doing things" (Papert, 1980, p.104). The observation is not isolated: Noss described children's tendency to draw 'outlines' and observed that this often persisted even after the children had had experience with procedures and had been introduced to the idea of sub-procedures (Noss, 1985 cited in McDougall, 1986). These all too familiar scenarios would suggest very clearly that breaking a problem into Papert's "mind-sized bites" is not instinctive nor is its development as a strategy, inevitable.

I contend that the problem goes deeper than merely being the result of learners' unfamiliarity with using a procedural approach to problem-solving. The nature of turtle graphics itself may well complicate learners' progression to a comfortable and powerful use of sub-procedures. The use of graphics immediately presents an environment to which learners can readily relate: this has long been Logo's meal ticket. However the hidden anomalies between the concrete reality of drawing a picture on paper and the abstract activity of producing a picture on a monitor may hinder the development of a structured approach to problem solving. Much of the appeal of graphics is that the drawing of patterns and/or pictures is a task which most children enjoy and with which they are familiar. We should not be surprised that learners resist adopting a new technique to solve what are for them, familiar problems.

Hillel (1985, cited in McDougall, 1986) supports this point of view in stating that children identify procedures with the objects they describe, seeing them as end products. The desire to use sequential programs to draw outlines in Logo may be directly linked to the way in which one draws pictures on paper. The familiarity of picture and pattern producing problems may be attractive to students but it may

also mitigate against their willingness to adopt the use of sub-procedures: graphics may provide motivation to achieve a product rather than adopt a process. In addition, interfaces between one sub-procedure and the next create significant problems for some students. "When conceiving of a given picture as a hierarchy of subpictures and interfaces, the subpictures can be directly perceived in the original picture whereas the interfaces cannot" (Leron, 1985, p.29).

In my own classroom experiences, the conceptual difficulties presented by sub-procedure interfaces in graphics programs was sufficient to deter some students from ever breaking their procedures into sub-procedures. The familiar task of drawing a picture of a house (for want of a better example) on paper may well come to be understood in terms of combining drawings of the walls, the roof, the door and the windows. However the exaggerated importance of positioning the turtle correctly so that the next "bit" might be drawn in the correct position and with the correct orientation was difficult for many students to comprehend. When drawing on paper, it was the drawing of the "bits" that demanded attention: thinking about putting one's pencil in the right position to draw the next part was undeserving of any conscious deliberation at all. Thus the familiar task is not quite as familiar as it first appeared: too often I have seen students' disappointment in a program with interface bugs resolved by their total abandonment of a sub-procedural structure. The fact that students are able to resort to sequential programs suggests that many of the turtle graphics tasks which they are given are too easily solved without using sub-procedures. There is no need for them to adopt a new problem solving strategy when firstly, the old strategy can still be made to work and secondly, the new strategy places such disproportionate importance to that part of the task which visually seems of no consequence.

The graphics situation in which I have observed a willingness among students to use sub-procedures is in the production of patterns rather than pictures. Having defined a procedure to draw a particular shape, students enjoy using their shape procedure as a sub-procedure in the production of more complex designs. Whilst this might be a productive activity from the aspect of fostering creativity, the non-directive approach which is usually taken by students does little to enhance their problem solving skills: in some cases it offers "freedom from thinking!" The ease with which learners can have "happy accidents" to achieve an impressive product tends to work against them ever feeling a need to think a problem through. Using sub-procedures in this environment seems to be of little benefit to students using sub-procedures in graphics projects in which a specific problem has been pre-defined and sub-procedure interfaces are critical.

An Alternative: Introducing Logo Through Music

If structured problem solving is neither instinctive nor inevitable, then we must more seriously consider the manner in which it will be taught. Educators must address the problem of how to impart a particular, transferable thinking skill rather than just 'giving them Logo' and trusting that their students will see the light. By failing to use sub-procedures learners are not implementing a powerful feature of the Logo language. As such, set tasks become conceptually more difficult and Logo no longer empowers the learner to think clearly about problem solving. I thus came to the conclusion that some teaching strategy must be sought to enable learners to access this power.

For the reasons outlined in the preceding section, it is my belief that the very nature of graphics problems can work against the powerful use of sub-procedures in Logo. The almost synonymous reference to Logo and turtle graphics

led me to investigate other possible means of introducing Logo to students. It was my hypothesis that if students began using Logo in an environment in which the use of sub-procedures was more natural and instinctive, then perhaps they would transfer that proclivity to other Logo applications at a later date. To reiterate, if students' early Logo experience encouraged procedurality, then they might embrace this as a strategy in future problem solving despite any extraneous conceptual difficulties which might arise. From a teaching point of view it may suffice to say that the teaching strategy selected to introduce Logo may be significant in affecting students' overall learning experience.

The idea of using music does not represent my first attempt to use something other than graphics to introduce Logo to children. Commencing with word and list processing projects led to a different range of difficulties: the paucity of immediate mode projects rushed students into what was perhaps a premature use of procedures; the sheer number of Logo primitives needed to get started was overwhelming for many students; and, to my dismay, the sequential nature of natural language merely tempted more long, complex procedures. I had overcome the difficulties in spatial problems in sub-procedural interfacing but had re-discovered a lot of good reasons as to why everybody else was still introducing Logo through graphics. And so, I turned to music.....

The title of this part of the paper could just as easily have been "Introducing Music Through Logo." The fact that it isn't restates my motives in exploring this approach to Logo. Here I use the phrase 'approach to Logo' quite literally: it refers to the direction from which a learner's Logo experiences might commence. It was my

hope that the intuitive breaking of a tune into bars/measures and bars/measures into notes would provide natural sectioning into sub-procedures without involving any conceptual spatial leaps. Moreover students could enjoy a reasonable introduction in immediate mode and later start defining their own 'note' procedures with no more than the TOOT primitive at their disposal. Students could be gently introduced to the notion of program structure because the structure is already provided by the composer: they need only devise their own structures having had some experience in manipulating existing structures. From the inspiration point of view, I decided that everybody must know a few tunes: each student therefore should come complete with a vast storehouse of personalised introductory programming tasks.

For a musician, there are obvious hardware limitations in using most of the micro-computers which inhabit our schools. And let us not forget the software limitations of the more popular versions of Logo in schools (my work was done using the TOOT command in Apple Logo II). However in this particular context, these limitations can be turned to educational advantage. The experience is intended to provide an introduction to problem solving in Logo, not a vehicle for musical composition. The reliance on classmates' computers for extra volume (unison), harmonies, and playing rounds provided the unexpected bonuses of close interaction, co-operation and planning within the class.

In the session, it is my intention to more fully detail my use of music in introductory Logo lessons. I wish to present the structure and purpose of three Logo lessons which were given to eight Year 7 classes (11/12 year olds) as their first lessons in Logo. Also I would like to examine the impact

that this particular form of introduction had on these students' later approaches to Logo problem solving. It is well beyond the scope of this paper to include such details herein.

Conclusion and Implications

It is my contention that the type of Logo experience which learners encounter will have a significant influence on the nature of the outcomes achieved. Of particular importance is the learner's early experience as this may well set the pattern for a learner's continued approach to problem solving. In his reply to conflicting analyses of the impact of Logo, Papert states that "Logo environments differ in many relevant ways that are not mentioned in the reports of either study...Several of my colleagues and students have been probing the diversity of factors that make a difference" (Papert, 1985, p.62). This is echoed by other researchers who consider that "Many studies have not adequately addressed the type of teaching and learning experiences a (Logo) social context provides" (Au et al., 1987, p.12; Clements, 1985; Gorman and Bourne, 1983).

This paper is perhaps an embryonic attempt to address just one of those factors which Papert believes may make a difference: namely, that the manner in which a learner is introduced to Logo may significantly influence his/her total learning experience. The findings of this preliminary study suggest that this avenue may be deserving of more rigorous research projects in the future. Inherent in pursuing this direction is the belief that, despite its possibilities and, in common with any educational computing application, Logo can only be as beneficial as its method of teaching will allow. If Papert's vision for Logo is ever to be achieved on a grand scale, we must carefully investigate the role of the teacher and the impact of teaching approaches in determin-

ing desired educational outcomes. And from there? ...the implications for teacher training should be obvious.

References

Au, W.K., Horton, J., & Ryba, K. (1987) "Logo, Teacher Intervention, and the Development of Thinking Skills", *The Computing Teacher*, November, pp.12-16.

Bloom, B.S. (Ed.) (1956) *Taxonomy of Educational Objectives*, Longmans Green, New York.

Clements, D.H. (1985) "Effects of Logo Programming on Cognition, Metacognition, Skills, and Achievement", Paper presented at the annual meeting of the American Educational Research Association, Chicago.

Dale, E.J. (1984) "Logo Builds Thinking Skills", *Proceedings of NECC '84*, Dayton, Ohio.

Gorman, H. & Bourne, L.E. (1983) "Learning to Think by Learning Logo: Rule Learning in Third Grade Computer Programmers", *Bulletin of Psychonomic Society*, 21, pp.165-167.

Hillel, J. (1985) "Some Thoughts on Logo Dichotomies", paper prepared for the World Logo Conference.

James, L. (1986) "Logo Study", *Proceedings of Australian Computer Education Conference*, CEGV, Melbourne.

Leron, U. (1985) "Logo Today: Vision and Reality", *The Computing Teacher*, Feb., pp.26-32.

McDougall, A. (1986) "Children's Difficulties in Perceiving Structure and Using Subprocedures in Logo", *Proceedings of Australian Computer Education Conference*, CEGV, Melbourne.

McMillan, B. (1987) "Logo and the Teaching of Logo: A Piagetian Perspective", *Proceedings of Australian Computer Education Conference*, CEGSA, Magill.

Noss, R. (1985), *Creating a Mathematical Environment Through Programming: A Study of Young Children Learning Logo*, University of London Institute of Education.

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

Papert, S. (1985), "Computer Criticism vs. Technocentric Thinking", *LOGO 85 Theoretical Papers*, 1, pp. 53-67 MIT, Cambridge, Massachusetts.

Splitter, L. (1988) "On Teaching Children to be Better Thinkers" *Unicorn: Journal of The Australian College of Education* Vol. 14 No. 1, Carlton, Victoria.

So Papert Was Right?: Evidence of General Skill Enhancement Due to Logo Experience

Susan M. Chambers

Deakin University

Geelong

In his celebrated book, *Mindstorms,* Papert (1980) convincingly presented the case for enhancement. He argues that a computer-rich environment provides children with learning experiences which are relevant to the development of 'formal' thinking. 'Formal' thinking in Piagetian terms is characterised by combinatorial and self-referential thinking skills. Papert and his colleagues created the programming language Logo and have developed Logo environments. Papert claims that Logo environments encourage the primary school children to develop combinatorial and self-referential thinking. He emphasises the need for children to have appropriate models available from the environment which can be used to develop and understand 'powerful ideas'. Logo programming encourages children to develop models of systematic procedures. Using these models children develop combinatorial thinking skills by learning to combine, integrate and play with alternatives. Evidence of 'formal' thinking is not usually found in children less than 12 years old. Papert argues that this is prima-

rily due to the lack of relevant experience available in our culture. When we provide experiences relevant to the acquisition of the necessary thinking skills, 'formal' thinking will become evident in younger children.

To test the effect of Logo experience, the performance of a large sample of children on computing tasks, non-computing tasks and psychometric tests was assessed for children who had varying amounts of Logo experience in their previous two school years. The method used to establish the effect of Logo experience was the hierarchical R^2 analysis by sets (Cohen & Cohen, 1983). In this multiple regression technique, covariates, the experimental treatment and their interaction terms are entered sequentially in three steps of a regression analysis. A unique effect of the experimental treatments demonstrated if there is a significant R^2 change when the experimental treatment is entered and if there is no significant R^2 change when the interaction terms are entered. This method was used to assess the unique contribution of Logo experience in predicting performance on several computing task measures, non computing task measures and psychometric test scores. These measures were the dependent variables.

The covariates were age, ability, school and test experience. All of these variables may affect a child's performance on tasks irrespective of Logo experience. In terms of Sternberg's (1985) model of intelligent behaviour, age would provide a child with experiences relevant to the acquisition of global knowledge. This knowledge could be drawn on when performing novel tasks. Ability would be reflected in the more efficient use of component processes, which again would provide additional relevant knowledge, as well as more efficient processing of the new information in a task.

The particular school attended by a child could influence the relevant experiences being encountered and used by the child. Previous test experience could also be utilised in performing new or familiar tests. These four variables were used as covariates in the present study and together with the Logo experience measure, comprised the independent variables.

Altogether 312 children participated in the study. They were aged between five years six months and twelve years eleven months at the beginning of the study. Ability was measured using the Otis-Lennon School Ability Test (OLSAT) which was administered prior to the experimental Program. The OLSAT scores ranged from 62 to 176 with a mean of 105 and a standard deviation of 16.

The children attended two local private schools. The schools were selected because they were introducing Logo programming for the first time. One school was a co-educational school with boys and girls from grades 1 to 8. The other school was a girls' school with children from grades 1 to 6. There were 241 children from the co-educational school and 70 children from the girls' school. The Logo sessions in each school took place in a computer room with several Apple II computers. In the co-educational school the children's regular classroom teacher was responsible for the Logo sessions. In the girls' school a researcher and a class teacher were responsible for the instruction. The involvement of two schools allowed for a measure of the effect of school environment on the performance measures.

The fourth covariate, test experience, was included as an additional control. Some children from grades 1, 3, 5 and 7 had been administered the Raven Progressive Matrices test prior to the test program. Because the result on a Raven test

administered at the end of the program was one of the dependent measures, a test experience variable was created to take into account the effect of prior Raven test experience.

Logo experience was defined as the number of Logo sessions in which each child participated. There was a naturally occurring variation in Logo experience across the entire age range. Initially it had been planned to study a subsample of the children from years 1, 3, 5 and 7. Children from these grades participated in two Logo sessions per week in the first year, and one Logo session per week in the second year. The teachers of the children from grades 2, 4 and 6 selected the number of Logo sessions in which their classes participated. Also, a number of new children (with no previous Logo experience) enrolled at both schools during the two year period. The mean number of Logo sessions was 63 with a standard deviation of 25 and a range from 5 to 100 sessions. These naturally occurring variations in Logo experience were fortuitous since they allowed the role of Logo experience to be studied across a wide age range holding constant the school contexts.

The nature of instruction in both schools encouraged the children to use Logo commands to write simple programs. Activity sheets and games were developed (Clarke & Chambers, 1985) to encourage experimenting, predicting, using analogies, coding, analysis and planning and debugging. These were used more exclusively in the girls' school. In the co-educational school these were used in conjunction with activities designed by the class teachers. Neither school provided explicit instruction in general problem-solving skills or transfer training.

At the end of the experimental program, the children's performance on a number of computing tasks was meas-

ured. Each child wrote two programs to draw pre- specified patterns. One program was written at a computer (the ON task) and the other away from a computer (the OFF task). Measures of planning, style of programming, accuracy and mode of planning were made. A computing knowledge test was also administered. It consisted of a multiple-choice group test designed to evaluate knowledge of computers and Logo programming.

Performance on four non-computing tasks was also measured. The tasks were selected on the basis that they appeared to share some common processes with programming tasks and thus provided measures of the transfer of skills to similar tasks as a result of Logo experience. The four tasks selected were the Tower of Hanoi task, a Block task, a Rotation task and a Typing task.

In the Tower of Hanoi task the child had to solve a problem consisting of moving a pyramid of four disks from one peg to a third peg without placing a larger disk on a smaller disk. The task requires systematic, step by step planning, comparable to writing a computer program. Measures of style, reaction time and accuracy were made.

In the Block task the child had to learn a new set of commands similar to a subset of Logo commands. Then a sequence of commands was generated by the child to fill a matrix with a pre-specified pattern on a computer screen. Measures of style, reaction time, accuracy and mode were made.

In the Rotation task the letter R was displayed in one of six orientations. The child's task was to decide whether the letter was 'normal' or 'reversed'. The task requires internal visualisation and manipulation similar to that which may be used in the turtle geometry tasks. Reaction time and accuracy measures were made.

The Typing task required a child to type a sequence of single letters displayed on a computer screen. Previous investigation (Chambers, 1984) had found that keyboard skills were encouraged by Logo programming experience when performance on the same task was compared for children with one year's experience of Logo and children with no Logo experience. Accuracy and reaction time measures were made. Further details of the tests and measures are available (Chambers, 1986).

The psychometric tests were the Raven Progressive Matrices Test (the Standard version), and several tests based on tests from the Kit of Factor-referenced Cognitive Tests (Ekstrom et al., 1976). They were selected to test whether Logo experience enhanced general thinking skills involving inductive reasoning (Raven), integrative planning (Following Directions), short-term memory (Memory Span) and cognitive style (Hidden Patterns).

The results of the study are clear. Logo experience enhances computing performance and some general thinking skills, but does not enhance performance on similar tasks. Table 1 shows the contribution of the covariates, Logo experience and the interactive terms for each dependent variable.

Table 1: Results for computing task measures, non-computing measures and psychometric tests.

	Covariates R^2	Experience R^2 change	Interactions R^2 change
Computing Tasks:			
ON programming task			
Plans	.13***	.08***	.01
Style	.16***	.05***	.01
Accuracy	.14***	.06***	.01
Mode	.15***	.04***	.01

OFF programming task				
	Plans	.10***	.02***	.04**
	Style	.17***	.02***	.01
	Accuracy	.13***	.03***	.01
	Mode	.19***	.03***	.01
Computer knowledge test		.39***	.03***	.01
Non-Computing Tasks:				
Tower of Hanoi				
	Style	.16***	.00	.01
	Accuracy	.15***	.00	.00
	Reaction time	.13***	.00	.01
Block				
	Style	.22***	.00	.03*
	Accuracy	.13***	.00	.02
	Reaction time	.14***	.00	.01
	Mode	.12***	.00	.02
Rotation				
	Accuracy	.19***	.01	.01
	Reaction time	.22***	.00	.01
Typing				
	Accuracy	.02	.00	.00
	Reaction time	.30***	.00	.00
Psychometric Tests:				
Following Directions		.28 ***	.01*	.01
Memory Span		.26 ***	.01*	.01
Finding Patterns		.68***	.02***	.01
Raven		.50***	.01**	.02*

* = p<.05 ** = p<.01 *** = p<.001

For performance on the computing measures Logo experience provided a significant positive contribution in predicting performance on eight of the nine measures. The effect of Logo experience on the four non- computing tasks (Tower of Hanoi, Block, Rotation, Typing) was consistently non- significant. Significant enhancement was found on the Memory Span, Following Directions and Hidden Patterns tests. The finding of a significant enhancement on performance for the Raven test was accompanied by a significant

school by experience interaction effect. Further analysis indicated there was a significant enhancement effect for the school with children in grades 2 to 6. A similar significant enhancement effect for Raven scores was found for children in the second school when the sample was restricted to those in the same age range (six years eleven months to twelve years four months).

The cautious critic will ask whether these effects are substantial. After the effects of the covariates are removed, Logo experience is only accounting for one to two percent of the variance in test performance. Alternatively, it could be argued that the effects are surprisingly substantial given that most of the children had less than 30 hours Logo experience. It remains open to question whether, if a greater proportion of the child's time were spent in a Logo environment, the enhancement effects would be more substantial. What can be said is that the evidence is in the predicted direction, it is significant and it provides some empirical justification for what many educators and parents believe.

Previous analysis (Chambers, 1986) suggested that programming performance was predicted by task specific programming knowledge and skills, and general skills. The specific effect of general skills was on metacomponent skills such as planning, monitoring, and the allocation of memory resources, rather than on task specific performance and knowledge skills. From that analysis it was not surprising that the pattern of transfer results indicated that Logo experience enhances task specific programming skills and the general skills (measured by the Raven, Following Directions, Memory Span and Finding Patterns tests), rather than performance on similar tasks (the Tower of Hanoi, Block, Rotation and Typing tasks). The particular Logo

environments in the two schools must have provided relevant opportunities for the general skills to develop. The results may differ when more explicit training is used. For instance transfer training for skills common in programming and other tasks, may lead to enhancement on those other tasks but not for general skills. General skill enhancement may require substantial relevant experiences over an extensive time period during which the child can learn to use the general skills more efficiently. Further research on the explicit and implicit nature of instruction in a Logo environment is needed to clarify this point.

Papert predicted that Logo programming environments would promote 'formal' thinking skills in children in the seven to twelve year range. It is of great interest to note that enhancement on the Raven test was restricted to children of this age range. Further, it has been demonstrated that Logo experience enhanced other important general thinking skills. Does enhancement of performance on the Hidden Patterns test indicate that the children are able to think more analytically, focussing on the relevant rather than the irrelevant? Can the children integrate information more adequately, as suggested by the results for the Following Directions test? Is their memory span really more efficient as indicated by the Memory Span tests results? The results suggest the answer to these questions is yes.

Footnote

This paper was prepared when the author was on Study Leave at Oxford University. The advice and encouragement of Donald Broadbent is gratefully acknowledged. Thanks is also due to the Deakin Foundation for financial support and the Schools for their cooperation.

References

Chambers, S.M., (1984), "Development of thinking in primary school children using Logo", *IT, AI and Child Development Conference,* Proceedings University of Sussex.

Chambers, S.M. (1986), "Cognitive processes used by children writing Logo programs", *British Journal of Developmental Psychology* (in press).

Clarke, V.A. & Chambers, S.M., (1985), *Thinking with Logo,* Sydney, McGraw-Hill.

Cohen, J. & Cohen, P., (1983), *Applied Multiple Regression / Correlation Analysis for Behavioural Sciences*, Hillside NJ., Lawrence Erlbaum Assoc.

Ekstrom, R.B., French, J.W. & Harman, H.H., (1976), *Kit of Factor-referenced Cognitive Tests*, New Jersey Educational Testing Service.

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

Sternberg, R.J..(1985), *Beyond I.Q.* Cambridge, Cambridge University Press.

Integrating Logo into the Secondary Mathematics Curriculum

Bruce Horton

Charles Sturt University - Mitchell

Bathurst

Introduction

This is a report on the final stage of a two year project designed to investigate the possibilities for integrating Logo into the current N.S.W. state mathematics curriculum. This stage of the project was undertaken by a group of twelve Year 9 (15 year old) students at a local High School in New South Wales, Australia, during 1988. The project was supported by a research grant of $1000 by Mitchellsearch Ltd, Charles Sturt University - Mitchell. The grant was fully expended in purchasing a site licence for the school for Logowriter, a version of the programming language Logo. The licence allowed every student to have a personal disk copy of the language to be used in lessons and at any other free time as the student wished.

The students were part of a class which, during 1987, had spent an extended time working with Logo within their mathematics lessons. This class was the top mathematically in a graded set of eight classes, and the gender mix was approximately half and half.

Methodology

In 1987 the students spent approximately eighteen hours learning Logo. This consisted of a one day (six hour) introduction followed by one hour per fortnight for two twelve week terms. During this time the emphasis in the sessions was on Logo as a programming language with the dual aims of encouraging generalisation of procedures, and encouraging a modular approach to programming. The mathematical context of these activities was geometrical algebraic, with work on polygonal shapes and tessellations being a major part of the geometry covered. The concept of a variable was introduced in the context of generalising such procedures.

In 1988, volunteers were called from the class to continue with the program, and twelve of the thirty in the class responded (four girls and eight boys). For 1988 the emphasis of the sessions changed, with the major aim being the integration of the Logo programming into the existing mathematics syllabus. The role of the Logo activities was to consolidate the mathematics already met in class. The activities were designed to give the students further opportunities to apply their mathematical knowledge and skills in an intrinsically motivating mathematical environment.

For the last three of the four terms for the year, the Logo group met for one 50-minute lesson a week (a total of about 25 lessons). For that lesson the rest of the class was taught by the class teacher. As the time for this lesson was a substantial part (20%) of the total time for mathematics over this period, a concentrated effort was made to ensure that none of the Logo group was disadvantaged relative to the rest of the class with respect to the coverage of the compulsory mathematics syllabus. This was done in two ways.

First, the first 1- 15 minutes of each lesson was spent covering, in an abbreviated form, the mathematics content being covered by the remainder of the class with the class teacher. Secondly, the Logo activities were planned wherever possible to provide an application of the concepts and skills involved in the current topic of the syllabus the children were learning in the other mathematics lessons of the week.

During the year the following Year 9 syllabus content was covered in this way:

> Algebra: binomial expansions, factorisation;
>
> Review of Geometry: basic properties: angles, parallel lines, exterior angle and angle sum of triangles, quadrilaterals, polygons;
>
> Constructions: bisecting angles, intervals, constructing right angles;
>
> Trigonometry: tangent, sine and cosine as ratios; finding sides, angles and problem solving, bearings; complementary ratios, exact ratios;
>
> Surds and Indices: constructions to show position on the number line;
>
> Coordinate Geometry: plotting sets of points; plotting linear functions; formulae - distance, mid-point, gradient.

Observations were made of students during their work on the various tasks and both written and practical evaluation tasks were given periodically.

The final practical evaluation for the year took place at Charles Sturt University - Mitchell Computer Centre in December 1988. This venue was used to allow each of the

ten students (two were absent) to have his/her own computer. The evaluation took the form of three questions on coordinate geometry, as this was the most recent topic studied. The students worked individually on a computer to answer the questions over a period of three hours (with a half-hour break in the middle).

A few days earlier the students had undertaken the normal final written test on coordinate geometry along with the rest of the class. The following analysis and conclusions are based on both the on-computer evaluation session and the final written test.

Results

The results of two separate tasks are given in this section. The first task was the traditional end-of-topic written test which was taken by the whole class, that is both the Logo group and the non- Logo group. This allowed some comparisons to be made between the two groups .

The second task was a practical one done on the computer. The task was attempted by the Logo group only, and consisted of questions on coordinate geometry. The students wrote Logo procedures to answer the questions.

Task 1: Written Class Test on Coordinate Geometry

(The actual questions used are given in the appendix.)

Table 1: Results for Task 1

Question		Logo Gp		Non-Logo Gp		t-test*
	Out of	Mean	SD	Mean	SD	p
1 (a)	8	7.42	0.90	7.39	0.91	0.935
1 (b)	4	3.80	0.60	3.70	0.80	0.527
1 (c)	5	3.50	1.85	2.78	2.21	0.353
1 (d)	5	3.40	1.90	3.10	1.70	0.698
2 (a)	5	4.83	0.57	5.00	0	na
2 (b)	6	5.29	1.71	5.56	0.85	0.580
2 (c)	10	5.92	4.42	4.11	4.39	0.280

Table 1: Results for Task 1- continued

Question		Logo Gp		Non-Logo Gp		t-test*
	Out of	Mean	SD	Mean	SD	p
3(a)(i)	5	4.40	1.50	4.10	1.70	0.587
3(a)(ii)	5	4.54	1.44	3.94	1.55	0.297
3(b)	10	7.00	4.45	4.53	4.47	0.148
4	10	8.71	1.23	8.75	2.02	0.950
5	12	4.13	4.00	3.39	3.76	0.612
6	5	3.60	2.30	2.90	2.50	0.463
7	10	4.00	3.80	2.40	2.90	0.199
Overall	100	70.50	14.0	61.60	15.60	0.121

(*Using a t-test, values given are 2-tailed probabilities using pooled variance estimate, n = 30.)

The results in Table 1 show there was not a significant difference in the means for the two groups either in individual questions, or in the test overall. However, while accepting the value of n is too small to make firm conclusions, some of the figures suggest trends which could be worth some follow-up. Those questions with $p < 0.3$ (questions 2(c), 3(a)(ii), 3(b) and 7) all require the students to apply a chain of reasoning and to have the ability to deal with generalisation. It could be that these abilities are enhanced by the study of Logo over an extended period of time.

Task 2: Questions on Coordinate Geometry Done on Computer

This detailed analysis of final practical evaluation questions was done by studying the 'dribble' files created by the students as they attempted the questions on the computer. The analysis is organised question by question.

Question 1
Consider the two general points (xl,yl) and (x2,y2).
Write a Logo procedure for each of the following cases.
(a) To output the distance between the two points.
(b) To output the mid-point of the interval joining the two points.
(c) To output the slope (or gradient) of the line through the two points.

These procedures had initially been developed by the class in whole-class discussion led by the teacher, and had been written and saved in a session four weeks earlier. Over the ensuing four weeks they had used these procedures quite extensively as tools to solve other problems, but had not re-examined them or re-written them. Hence this question required the students to both remember the actual formulae and the appropriate Logo syntax needed to implement the formulae as procedures. The original procedure for the distance developed in class included a subprocedure for squaring a number. It was of interest to find whether the students would use this approach.

Over the past two years as part of the Logo programming, a certain 'style' had been suggested to students. As part of this 'style', the use of the MAKE statement had been discouraged, except where it was seen as necessary. This had been suggested partly because those students with a background in BASIC programming had over used MAKE in their procedures from the beginning. Again, it was of interest to find whether students would use MAKE in these procedures as, in this case, they were unnecessary.

The 10 students are referred to in the table by capitalised letters of the alphabet.

Table 2: Results for Question 1

	Formula Mathematically correct	Degree of difficulty in writing the Logo procedure		
		Successful first or second go	Successful with difficulties	Unsuccessful
Distance 10/10	All 5 5	A,C,H,J,K 4	D,E,F,G 1	B
Mid-point 10/10	All 1	J 6	A,C,D,E,H,K 3	B,G,F@
Gradient 7/8	A,C,D,G,H,J,K 6	A,D,G,H,J,K 1	C 3	B*,E*,F#

Used SQUARE as subprocedure:	B,C,D,E,F,G.HJ	8
Used MAKE statement:	A,D,E,G	4

@ This student eventually corrected this procedure when using it in Question (b)
* These students didn't attempt this part, so no conclusions can be drawn regarding their knowledge of this formula
# This student had the wrong formula (adding coordinates instead of subtracting) but translated her formula correctly into Logo at first attempt.

Discussion

It can be seen that all the students basically knew the mathematical formula. Student F had troubles remembering the formulae, but after a false start to the formula for the mid-point, she was able to go back and correct herself after getting visual feedback when attempting question 2. She persisted however with the wrong formula for the gradient.

Student B was the only one who had difficulty with the Logo versions of the formulae. He had been paired with student E, and it would appear that most of the learning (of the Logo at least) was done by the partner.

Overall, some common problems emerged. The first of these is illustrated by the difficulty students had with the Logo procedure developed in class for the mid-point. The difficulty arises from the use of discrete variables for the x and y coordinates of each of the points. This meant each of the procedures took four inputs. This approach to the coordinates had historical roots, as the number plane had been used for many months in the context of drawing figures using the SETX and SETY primitives. It had been decided at that early stage not to use SETPOS, as the students hadn't met list processing in Logo.

This was to avoid introducing the FIRST and LAST primitives which would have been needed to extract particular values for the x or y coordinate of a point.

About a month before these evaluation tasks were set, the class were looking at writing Logo versions of the various formulae they had just formally covered in their normal mathematics class. During the discussion on a Logo procedure for the mid-point, the decision was made to introduce the list processing primitives of FIRST, LAST and LIST, showing the students how LIST made a list out of two distinct inputs. Hence the following procedure was developed.

```
TO MIDPOINT :X1 :YI :X2 :Y2
OP LIST (:X1 + :X2) /2 (:YI + :Y2) /2
END
```

It is this step which caused the students the problems. In retrospect, it would seem the decision made was incorrect as the two possible consistent approaches were mixed. decision should have been to either:

> continue with the treatment of the coordinates of a point as two separate numbers, and hence have had two procedures for the midpoint, one for the x coordinate and one for the y coordinate; or
>
> introduce the needed list processing primitives right at the beginning when SETX SETY, and SETPOS were first met, and then consistently have treated points as lists with two items, the x coordinate being the first item and the y coordinate being the second.

The second approach now seems more appropriate. This means the Logo implementation of the distance formula would become:

```
TO DISTANCE :POINT1 :POINT2
OP SQRT (SQ (FIRST :POINT2 - FIRST
:POINT1) + SQ (LAST :POINT2 - LAST
:POINT1))
END
```

While this looks more cumbersome than the version with four distinct inputs, it is considered the benefits for the future outweigh the slight increase in complexity initially.

Another error three students initially made was to confuse the Logo syntax and the mathematical syntax of the distance formula. Algebraically, one of the terms is expressed as $(x2 - x1)^2$ and these students attempted to write this in Logo as (:X2 - :X1) SQ rather than as SQ (:X2 - :X1). However, those that did this all eventually 'debugged' successfully.

With reference to the programming 'style', eight of the ten students used the subprocedure for squaring a number within the procedure for the distance, but four still persisted with the use of MAKE where not necessary.

Question 2

(a) Write a Logo procedure to draw a triangle whose vertices are the points A(20, 50), B(-30, 40), and C(60, -10).

(b) Use the procedures you wrote in Question 1 (or get a copy of the correct procedures from the teacher) to check the theorem: "the join of the mid-points of any two sides of the triangle is parallel to the third side and half the length of the third side".

This question was designed to give the students the opportunity to demonstrate their approach to a problem in a form they hadn't previously encountered. The first part should have been trivial for all students, the second part more demanding. However, as this second part used the procedures developed in Question 1, it was expected most stu-

dents would be able to achieve some success. Those students who had difficulty with Question 1 were given a printed sheet of paper containing the procedures for distance, mid-point and gradient. It is easy to know when this took place by looking at the dribble files for these students.

Each part of the question will be analysed separately.

Table 3: Results for Question 2(a)

Successful first or second go	Successful with difficulties	Unsuccessful
A,C,D,F,H,J	B,E,K	G

Discussion

As expected, all but one student was successful with this part, with most being successful at the first or second attempt. The one unsuccessful student had problems with the syntax, and tried many variations on (20,50) except the correct one of [20 50].

Results - Question 2(b) and Discussion

Each student approached this part in his/her own way, the discussion of results starts by considering each student separately. General comments are left to the end.

Student A This student checked the theorem in immediate mode.

Student B This student didn't have much time left for this question. He started by defining a procedure which invoked an error message about PU not outputting to PR, and that was as far as he went. However he was on the way to at least draw the figure, and had used the MIDPT procedure correctly.

Student C This student is one of the more able, and her case illustrates some of the problems very well. Her overall

strategy was valid: to set up new variables for the mid-point and then use these for drawing and calculating distances. The difference between "A and :A caused her problems at this stage. She overcame the problem by SETPOS MID 60 -10 30 40, but then the syntax problem mentioned in the discussion section of Question 1 arose. She tried DIST MID 20 50 60 - 10 MID 60 -10 -30 40, but as MID outputs a list and DIST has 4 numbers as inputs error messages result. Her reaction to this was to recognise the problem and to define two new procedures, one MIDX and the other MIDY, but unfortunately the formulas she used were wrong. She also defined a new procedure HDIST to output half the distance. Eventually she corrected the problem of the wrong formula for MIDX and MIDY. Her final procedure constituted a reasonable response, despite the one bug remaining of incorrect inputs to MIDX and MIDY, (MIDX should take x coordinates of two points as input whereas her inputs are the x coordinate and y coordinate of a point). She made no effort to generalise this procedure.

Student D This student was another who was quite able, having good strategies for the solution of the task, but having problems implementing the strategies due to the confusion associated with not knowing whether to treat coordinates as two separate numbers or as a list of two numbers. He started by defining a generalised procedure, but when he started to try to calculate the gradient the syntax problems began. For example, he tried MAKE "M :X2 :Y2, MAKE "N :X1 :Y1 and OP GRAD :M :N with the thought (presumably) of creating one variable M for the point (x2,y2), another for the point (x1,y1) so that he could use them as inputs to GRAD. When this led to error messages he did not understand, he went on with part (c) of the question.

Student F This student started by drawing another specific triangle (0,60),(30,0),(-30,0). She had the right idea with the use of SETPOS MIDPT O 60 30 0, but her MIDPT wasn't working. This provided her with the feedback needed to get it working. Her procedure drew the triangle and the join of the mid-points of two of its sides, but didn't check any calculations of gradients or distances.

She then returned to the specific triangle in the question and substituted the new coordinates in the appropriate places. She included two print lines in the procedure to print out the gradients, PR GRAD 20 50 -30 40 and PR GRAD 60 -10 -30 40. But these are the gradients of two of the sides, and it took a while before she tried PR GRAD MIDPT 20 50 -30 40 MIDPT -30 40 60 -10, which didn't work for the previously mentioned reasons, so she replaced the line with PR GRAD 20 50 60 -10 and PR GRAD -10 80 -50 30, the last set of inputs being her (wrong) calculations for the mid-points of the two sides. This was as far as she reached in the time available.

Student G This student started by successfully creating a procedure to draw a triangle given six inputs for the coordinates of the three vertices. He then added statements for midpoints and gradients. This attempt exhibited the desire to generalise the procedure and, apart from the syntax problems that keep turning up, is a good approach to the task. This was as far as this student went.

Student H This student, when answering part (a), generalised to a procedure which generated random triangles. Having done that, he decided to build upon his random triangle procedure, but encountered the same syntax problems as many of the others. He tried SETPOS LIST

MIDPT :Xl :Y1 :X2 :Y2 which gave an error message, but his attempt for the distance shows some feeling for the problems he encountered, i.e. PR DIST (FIRST MIDPT :X1 :Y1 :X2 :Y2) (LAST MIDPT :Xl :Yl :X2 :Y2) -> (FIRST MIDPT :X2 :Y2 :X3 :Y3) (LAST MIDPT :X2 :Y2 :X3 :Y3).

He then returned and debugged the original MIDPT problem, and tried to improve it with printed messages, but took a while to get the syntax correct. He finished with a procedure that still had a bug in it as it did not calculate the distance correctly (he had a point (y2,y2) instead of (x2,y2) in one line), and it didn't even look at the gradients to check for parallelism. But it still constituted a good attempt at the task.

Student J This student was the only one who normally worked on his own in class. He had little patience with others, and worked quickly and impulsively at the keyboard. He was one of the students with access to a home computer (Amstrad), with which he wrote many quite complicated programs in BASIC. He was also one of the most capable students in the group with respect to computer knowledge and skills.

He started by writing a procedure to create the triangle formed by the joins of the mid-points. This was done correctly and quickly. He then decided to generalise and randomise before anything else, but ran into the same syntax problems as the others, but in the reverse context as he had defined each of his random points as a list with two items. i.e. MAKE "P1 LIST (RANDOM 280) - l40 (RANDOM 180) -90. But then the line SETPOS MIDPT :Pl :P2 caused problems, which he overcame after three attempts by using FIRST and LAST. Instead of looking at lengths

and gradients, he then made the procedure (tail) recursive and played with that for a while.

Then he returned on-task, and after a few problems with the syntax for printing messages, his final procedure created a random triangle, then calculated and printed out the gradients required. He didn't worry about the distances at all.

Student K This student also has his own computer at home (Commodore Amiga) and was the most mathematically able in the class. He was in the process of teaching himself the C language at the same time as attending these sessions.

His first step with this task was to define a procedure:

```
TO THEOREM :X1 :YI :X2 :Y2 :X3 :Y3
MAKE "A MIDPT :X1 :YI :X2 :Y2
MAKE "B MIDPT :X2 :Y2 X3 :Y3
END
```

Without testing this, he immediately edited his procedure and added:

```
PU SETPOS LIST :X1 :Y1 PD
SETPOS LIST :X2 :Y2
SETPOS LIST :X3 :Y3
SETPOS LIST :XI :YI
```

at the beginning, and

```
PU SETPOS LIST :A PD SETPOS LIST :B
```

at the end.

He then spent quite a time debugging as his understanding of LIST was incomplete. He tried:

```
MAKE "A LIST MIDPT :X2 :Y2 :X3 :Y3
```

and finally had success with SETPOS :A. He then added calculations.

```
MAKE "C SLOPE FIRST :A LAST :A FIRST
:B LAST :B
IF :C = :D [PR "LINES ARE PARALLEL" ]
```

The message had syntax problems which he debugged successfully by using

```
[PR LIST "LINES "PARALLEL].
```

He then added calculations and test for the distance:

```
MAKE "E DIST FIRST :A LAST :A FIRST :B
LAST :B
MAKE :F DIST :X1 :YI :X3 :Y3
IF :E = ( :F * 2) [PR LIST "HALF
"LENGTH]
```

When this didn't work, he used immediate mode to print the values of E and F (E was 36.055 and F was 72.111) and then experimented with an alternative test

```
IF ( :E - ( :F * 2) [< 1 [PR LIST "HALF
"LENGTH]
```

and was satisfied when that worked in this case. At all times he tested his procedure with the given coordinates, (20,50), (40,60), (60,-10), so while he had a generalised procedure, his curiosity didn't extend to testing other triangles.

However, he then did go on to the next two parts of the question, and ended up getting further through this part than anyone else.

Conclusions for Question 2(b)

As can be seen from the previous discussion, the students had a variety of strategies for tackling this problem and there was a range of degrees of 'success', where success was to check the theorem with the specific triangle in the first instance but then to generalise their procedures and

even randomise the points. The following table attempts to classify the student's responses on two criteria, the degree of 'success', and the strategy used, whether the result was checked in immediate mode or whether a procedure was used, and if a procedure was used, was it generalised and was randomness introduced.

Table 4 Question 2(b) Degree of success/Strategy used

	Successful	Successful (partially)	Unsuccessful (but started)	Unsuccessful (fully)
Immediate Mode	A			G
Specific Procedure		C, F	B	E
Generalised	K	H, D, G		
Generalised and Random	J			

All the students except one were able to get started on the this part of the question

Overall Conclusions for Both Tasks

The results show that overall the students certainly were not disadvantaged by undertaking this program when their results in mathematics over the year are studied and compared with the rest of the students in the class. However it is less clear whether the students were advantaged, even though the results of the written test on coordinate geometry suggest that the Logo group of students were able to handle multiple-step problems more successfully than the rest of the class. It is conceded that there are so many other variables involved over a long term study like this that no firm conclusions can be drawn whether or not the study of this Logo program made a significant contribution to these observed differences in the two groups.

A comment should be made concerning the frequency of the exposure the children have to Logo. During the first year of the project, the children had one lesson per fortnight at best. This led to children forgetting some of Logo syntax

and the techniques of editing etc. from one session to the next, and generally proved an unsatisfactory approach. It was partially to overcome this problem that the group was made smaller for the second year. This size of group allowed the children to work with Logo once a week at best. However, even this wasn't sufficient to overcome the problems of remembering Logo syntax and techniques. A study of the dribble files at the end of the second year reveals a surprising number of errors being made in both these areas. In many cases, these problems interfered quite significantly with the mathematical problem solving performance of the children. Several times a reasonable strategy had been decided upon for a particular problem, but the lack of facility with the Logo language caused lack of progress.

This difficulty with syntax was apparent not only when using list processing primitives such as LIST which, given their little exposure to list processing, was only to be expected, but also when using syntax with which they were expected to be familiar. Most still had trouble with spacing, particularly when dividing or subtracting two numbers. A few still had difficulty with the editor. The lesson to be learned is that the frequency of using the language is a vitally important factor, more important than the total number of hours.

A final comment concerns the use of pairs or groups of three in these activities. An individual's learning style needs to be considered first and foremost. Some children learn well as part of a small group, still others do best when working most of the time on their own. This does not mean that the role of the teacher as a facilitator is diminished, only that the teacher needs to use a variety of strategies for discussion with the students, including whole of class

discussion when setting the scene or drawing together the results of the various groups, and small group/individual discussions while the activities are taking place.

Appendix

Questions for the Final Written Class Test on Coordinate-Geometry

Question 1

(a) Plot the points A(3,), B(5,3), C(6,1) and D(-2,-1)

(b) Name the type of shape formed by ABCD

(c) Calculate the area of ABCD

(d) Find the slope of AB and BC

Question 2

(a) Write down the distance formula

(b) Plot the points A(1,-2), B(4,4) and C(- 2,7)

(c) Prove whether or not triangle ABC is a right triangle

Question 3

(a) Use the formula to find the mid-points of:

(i) (-3,-2) and (0,6)

(ii) (n,n) and (4n,7n)

(c) If the mid-point of (x,l) and (7,y) is (9,6), find the values of x and y.

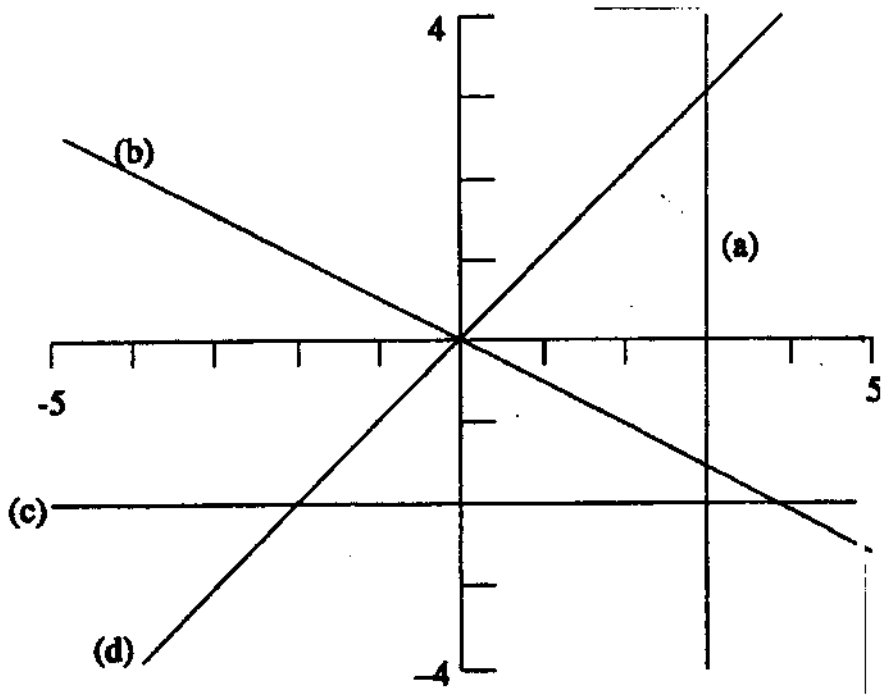
Question 4

Draw axes and then graph the line y = 0.5x - 1

Do this by selecting suitable x-values and calculating the resulting y-values.

Question 5

Determine the equations of the lines labelled (a), (b), (c), (d).



(d) /

Question 6

Find the gradient of the line passing through the points (-5,6) and (10,12).

Question 7

Put the equation 4x - 3y = 6 into gradient - intercept form and thence sketch the line.

Bibliography

Abelson, H., & diSessa, A., (1981). *Turtle Geometry: The computer as a medium for exploring mathematics.* Cambridge, Massachusetts: MIT Press.

Ball, D., (1986). *Microcomputers in Mathematics Teaching.* London: Hutchinson.

Burke, M.P., & Genise, I.R., (1987). *Logo and models of computation.* Menlo Park, California: Addison Wesley.

Computer Education Unit, 1986. *Using computers for problem solving in mathematics: an ideas book.* Sydney, CEU026, NSW Department of School Education.

Dalman, P., (1985). *Ideas: Logo and traditional mathematics.* Central Highlands/Wimmera Computer Education Centre.

Downes, M., (1987). "Using programming in the teaching of mathematics" in *Information Transfer, 8, 1,* 28-33.

Horton, B., (1987). "Should Logo programming be part of the secondary mathematics curriculum?" in *Information Transfer* ,7,3, 41-43.

Hoyles, C., (1985). "Developing a context for Logo in school mathematics" in *The Journal of Mathematical Behaviour,* 4, 237256.

Klotz, F., (1986). "When is a program like a function" in *Micromath,* 2,3,

Noss, R., (1983). "Doing maths while learning Logo" in *Mathematics Teaching,* 104, September.

Papert, S., (1980). *Mindstorms: children, computers and powerful ideas.* Brighton, England; Harvester Press.

Wilcox, J.F., (1986). "Technology: a challenge to mathematics teaching in the lower secondary" in *The Australian Mathematics Teacher, 42, 2,* 6-9.

Logo: Has It Had Its Day?: An Historical Analysis of Past and Present

John Turner

Presbyterian Ladies' College
Burwood, Victoria

At the recent Logo International Conference held in Melbourne in July 1992 it appeared to me that there was a relative newness in the Logo awareness and experience of many teachers in attendance. Such an observation is in accordance with my observations at Logo-based presentations provided at recent Mathematical Association of Victoria (MAV) conferences. Such observations are at odds with conjectures made in some Logo quarters (Hoyles & Noss, 1992) about the coming together of mathematics education and Logo computer cultures.

This paper explores the apparent contrast from a historical perspective. It presents an overview of Logo and computer issues raised through published MAV conference papers since 1980. This period is significant for two reasons. First, the advent of personal computing within schools really only took off after the 1970s. Second, 1980 saw the publication of *Mindstorms: Children, Computers and Powerful Ideas*, a seminal work by the person who is responsible more so than anyone for the initial development and consideration of Logo, Seymour Papert. Perhaps there is also some sig-

nificance in the common dating of these two occurrences? Certainly it is a good time to reflect on the 13 years of Logo, as Papert's recently released book *The Children's Machine* (1993) takes up many of the themes first presented in *Mindstorms.*

The Introduction of Logo into the Victorian Mathematics Community

Prior to 1981 considerations on the use of computers were spread across a wide range of interests. In 1980 about 20% of the conference papers were grouped under a section headed "Silicon Technology - Implications to Applications". At this time calculators and their use within the classroom was a strong theme, as well as the implications of the changes to education that technology was bringing to the student (Salvas, 1980), to the classroom (Collins, 1980; Robson & Thomas, 1980), to Mathematics teaching (Olsen, 1980) and to the education system as a whole (Maynard, 1980). Programming and Computing Studies were considered within the Mathematics curriculum (Thompson, 1980; Waterson, 1980) and option units for HSC Mathematics were provided (Rosenberg, 1980).

This was a time of mathematics education upheaval, at least internationally. In 1980 *An Agenda for Action* (National Council of Teachers of Mathematics) was released in the United States, followed in 1982 by the Cockcroft Report (Cockcroft, 1982) in Britain. Both highlighted a need to improve the way students were being encouraged to learn mathematics. Conference themes reflected a perceived need to justify and explain. In 1980 teachers were called upon to project their love of the subject, while 1981 saw concern about misconceptions that distorted the reality of the classroom.

Into this climate Logo was introduced. In presenting Logo Sandra Wills (1981) emphasised much that Papert had proposed in *Mindstorms* where he built upon criticism of school and the way computers were being used in schools. She saw Logo as "a powerful descriptive system, powerful because it enables the user to describe shapes not as objects but as processes". Examples of geometric shape procedures and graphics highlighted Logo potential. Logo programming was extolled for its value towards teaching programming within the Mathematics syllabus.

The Early Years: 1982 - 1984

Conference themes reflected a subject seeing itself under siege. In 1982 concern was expressed about reaching "parents and the students who cannot wait to 'drop Maths' ", while 1983 saw efforts to justify the "essential" nature of Mathematics. There was, in 1984, a self-reflective call for "a dramatic change in the quality of the Mathematics curriculum".

While the influence of computing studies focused papers started to wane, this period saw a continuation of papers dealing with calculators, programming and computer-based curriculum applications. All were presented within technology sections. The 1982 "Working With Mr Chips" was followed in 1983 with "The Impact of Technology" and "Technology - Mastery Versus Mystery" in 1984.

Logo examples, while strong in number, tended to primarily focus on what Logo had to offer rather than on any accommodation between the two cultures. Jones (1982), and McDougall & Adams (1983), provided views on the introduction to using Logo, while Malone (1982, 1983, 1984) addressed strategies for using Logo in the classroom in support of the educational aims of the school. Carr (1983)

appears to have confused appeal with worth in a utilitarian examination of whether Mathematics teachers should use Logo. Malone (1983) acknowledged that progress in the integration of computers within the school curriculum depended on one or two enthusiastic teachers, a point Papert (1993) later endorsed.

Examples of procedures generated by interested educators started to emerge during this period (Fox, 1984). While the wider issue of the implications of technology on education was taken up by Crawford (1984), Srivastava (1984) pointed to the disappointment of cultural assimilation as evidenced by the use of Logo through using "a well-structured sequence of steps for obtaining certain geometrical shapes". Towards the end of this period there appeared papers on other micro-computer package genres, such as spreadsheets (Pekin, 1984; Smale, 1984)

Standardisation and Debate: 1985 - 1987

This period is significant for several reasons. Franklin (1990) pointed out that, with the introduction of new technology, after periods of invention, growth, and acceptance, there is likely to occur periods of standardisation and stagnation. That such a period had been reached in educational computing in Victoria by the mid-1980s was acknowledged in computing policy statements (Walker, 1991). In addition, around this time there emerged research, such as that by Pea & Kurland (1984), that cast doubts over the educational merits of Logo as proposed by Papert and others. Such criticism was supported by articles (Kerr, 1986) that dismissed Logo as failing a cost/benefit analysis for classroom value.

This period saw a holding of the line as far as Logo papers were concerned. While Ellerton (1985), Jones & Nevile (1986, 1987), and Jones (1987) looked at where the rela-

tionship between Logo and Mathematics lay and the potential for microworld exploration of Mathematics, generally papers continued to be built around individually generated material. Fox (1986) provided examples of using Logo for numerical methods work as part of the HSC. Rule (1987) provided activities suitable for Year 7 - 10 Mathematics problem solving. Lewit and Fox (1987) looked at using Logo to investigate 3-dimensional tessellations. Finally, Nevile (1987) provided an explanation of recursion using Logo as a unique mathematical experience.

This period saw a rise in consideration of "functional" packages, including databases and spreadsheets, as part of a general trend in "assessing" software packages. The use of communications systems and robotics also were of interest . Technology papers continued to be segmented under a common heading, except for 1987 when papers were ordered in sequence from preps to tertiary, the majority of technology related papers appearing in the latter half.

Conference themes, while dealing with introspection in 1985 and change to be brought on by the forthcoming VCE in 1986 and 1987, found some time in 1987 to consider the impact of technology on the way mathematics was learned.

Climate of Educational Change: 1988 - 1990

The late 1980s saw some fundamental curriculum changes in education in Victoria. The introduction of the Victorian Certificate of Education and the release of Curriculum Frameworks saw Mathematics teaching and learning spotlighted. Calls for a broadening of teaching methods and the perceptions of what constitutes Mathematics (Ministry of Education, 1988) promoted much that Logo adherents had championed concerning investigative student-based project work. Conference reaction, as reflected

in the themes, was one of concern about the pressures that these changes were placing on the system.

MAV papers for this period continued very much in the vein of the preceding periods. Technology papers were generally sectionalised (1988 being an exception). Logo based projects were presented in the same context as other packages and new technologies. McAuliffe (1988) wrote of Logo fractals, and Buckingham (1990) provided a skills-based approach to using LogoWriter. Concerns were raised by Davey (1988) about the Logo experiences that school children were being provided with, and Herbert & Crouch (1989) provided an analytical piece on the need for changing practices in the classroom if Logo was to fulfil its potential. Others, such as Nevile (1987), had moved on to Lego/Logo, raising many of the same issues previously identified with Logo.

Of significance during this period was a survey of post-primary schools reported by Barraclough (1990) based on the return of 46 schools out of 688. The survey showed that about half of the respondent schools were using Logo in Year 7 to "assist in the development of basic geometrical concepts and shape design" but that this decreased in Years 8, 9 and 10.

VCE and Workloads: The 1990s

Since the implementation of the VCE there has occurred a period of intense pressure, highlighted by changes in workload demands on student and teacher, and a changing VCE Mathematics curriculum. Conference themes, however, seem to have taken an up-turn, highlighting student needs in 1991 and classrooms focusing on children in 1992.

Apart from papers looking at the relevance of Logo (Turner, 1991) and school models in support of using Logo in Mathematics learning (Turner, 1992), there has been a dearth of

Logo references during this period. Other technologies, such as spreadsheets (in increasing numbers), calculators (particularly because of new graphical capabilities) and specific packages such as ANU Graph and Minitab, have been to the fore.

Implications for Current Considerations

This historical analysis points to several areas of concern regarding the use of Logo, and computers in general, within the Mathematics curriculums of Victorian education.

(a) While the consideration of technology has been a consistent consideration within conferences, the allocating of technology to its own section is indicative of its treatment as a separate component of the mathematics learning culture.

(b) Logo has also been consistently treated in segmented isolation. Such an approach has extended beyond the boundary of individual teacher consideration. Both the Victorian Ministry of Education in its Frameworks (1988), and the *Australian Education Council in its National Statement on Mathematics for Australian Schools* (AEC, 1991), while acknowledging the Mathematics education merits of Logo, limit references to fragments, thus diminishing one of its basic potentials.

(c) The authors of Logo papers generally fit into one of two main groups. There are those written by tertiary and other leading educators, from a theoretical or insulated viewpoint, that promoted the merits of Logo and/or identified pitfalls and shortcomings that required addressing. The second group are practising teachers who highlighted educational outcomes they

had achieved using Logo. There is a glaring lack of consideration of Logo within the school in terms of school/Logo (or school / computer) cultures. Assimilation, maybe. Accommodation, no way.

Conclusion

Where does this leave us with Logo? Has it had its day ? I go back to Sandra Wills' introduction in 1980 of Logo as a "powerful descriptive process". We need to re-evaluate the use of Logo, and computers in general, in this light to see if we can't make more of such potential to enhance each students mathematical learning and motivation for student-based exploration and mathematical constructive learning.

There remain unresolved issues as current for other computer packages as for Logo. Of primary importance is whether, in judging the inadequacies of new technology, we are in fact judging our own capacity for developmental change. This is not to say that we need to lose our objectivity. The practicalities within most schools requires assimilation, at least on a resource level.

In *The Children's Machine* Papert (1993) warns of the dangers in measuring value in terms of the ability to adapt new entities into defensive environments. He highlights the characteristic of conservative systems (such as schools) to "accommodation only when the opportunities of assimilation have been exhausted".

Salvas (1983) perhaps best summed up the dilemma when he described the impact of technology in terms of "needing to use the new technology to help us teach Mathematics". The lack of student focus is glaring. It is to be hoped that recent themes will continue and reach out to address this missed opportunity. We need to look at beyond merely

reinforcing old ways. We need to look at Logo, and the computer, as vehicles for even more change towards student-based learning. We still have a way to go in providing the most effective developmental learning processes to our students that technology can provide.

Bibliography

Barraclough, M. (1990). Use of the computer in mathematics teaching / learning. In K. Clements (Ed.)*Whither Mathematics*, (pp. 173 - 175). Melbourne: Mathematical Association of Victoria.

Buckingham, E. (1990). Why use LogoWriter? An introductory workshop. In K. Clements (Ed.),*Whither Mathematics*, (pp. 207 - 213). Melbourne: Mathematical Association of Victoria.

Carr, P (1983). Could you use Logo - Should you use Logo. In D. Blane (Ed.), *The Essentials of Mathematics Education*, (pp. 330 - 336). Melbourne: Mathematical Association of Victoria.

Cockcroft, W. H. (1982). *Mathematics Counts, Report of the Committee of Inquiry into the Teaching of Mathematics in Schools*. HMSO, London.

Collins, R. (1980). Micro computers in the primary school. In P. Williamson (Ed.), *Learn to Love Mathematics*, (pp. 362 - 364). Melbourne: Mathematical Association of Victoria.

Crawford D. (1984). School mathematics and information technology: Issues and implications. In A. Maurer (Ed.), *Conflicts in Mathematics Education*, (pp. 292 - 309). Melbourne: Mathematical Association of Victoria.

Davey, B. (1988). Logo for little kids. In D. Firth (Ed.), *Maths Counts - Who Cares ?*, (pp. 219 - 221). Melbourne: Mathematical Association of Victoria.

Ellerton, N (1985). Children, teachers, computer and mathematics - How do they interact ? In P. Sullivan (Ed.), *Mathematics Curriculum, Teaching and Learning*, (pp. 42 - 28). Melbourne: Mathematical Association of Victoria.

Fox, C. (1984). The animated blackboard. In A. Maurer (Ed.), *Conflicts in Mathematics Education*, (pp. 363 - 367). Melbourne: Mathematical Association of Victoria.

Fox, C. (1986). Numerical methods Logo style. In N. Ellerton (Ed.), *Mathematics, Who Needs What,* (pp. 272 - 277). Melbourne: Mathematical Association of Victoria.

Franklin, U. (1990). *The real world of technology.* CBC Enterprises, Montreal.

Herbert, G. & Crouch, W. (1989). Logo in the classroom: A graphic view. In B. Doig (Ed.) *Everyone Counts*, (pp. 111 - 116). Melbourne: Mathematical Association of Victoria.

Hoyle, C. & Noss, R. (1992). *Learning Mathematics and Logo.* MIT Press.

Jones, T. (1982). Logo and turtle. In J. Dowsey (Ed.),*Working With Mathematics*, (pp. 241 - 247). Melbourne: Mathematical Association of Victoria.

Jones, T. (1987). Logo variables for the primary school. In W. Caughey (Ed.), *From Now to the Future*, (pp. 147 - 150). Melbourne: Mathematical Association of Victoria.

Jones, T, & Nevile L. (1986). Primary maths and problem solving with Logo, In N. Ellerton (Ed.), *Mathematics, Who Needs What*, (pp. 268 - 271). Melbourne: Mathematical Association of Victoria.

Jones, T. & Nevile L. (1987). Learning environments - Can they be items in a mathematics curriculum? In W. Caughey (Ed.) *From Now to the Future*, (pp. 268 - 271). Melbourne: Mathematical Association of Victoria.

Kerr, J. (1986). Logo - What went wrong. In *COM 3, 18(2).*

Levit, A. & Fox, C. (1987). Logo and 3-D tessellations. In W. Caughey (Ed.), *From Now to the Future*, (pp. 205 - 209). Melbourne: Mathematical Association of Victoria.

McAuliffe, R. (1988). Fractals in the classroom. In D. Firth (Ed.), *Maths Counts - Who Cares ?*, (pp 228 - 232). Melbourne: Mathematical Association of Victoria.

McDougall A. & Adams T. (1983). Logo and primary maths. In D. Blane (Ed.) *The Essentials of Mathematics Education,* (pp. 325 - 329). Melbourne: Mathematical Association of Victoria.

Malone, T. J. (1982). Strategies for using Logo in the classroom. In J. Dowsey (Ed.), *Working With Mathematics*, (pp. 254 - 258). Melbourne: Mathematical Association of Victoria.

Malone, T. (1983). Logo - A learning environment. In D. Blane (Ed.), *The Essentials of Mathematics Education,* (pp. 337 - 341). Melbourne: Mathematical Association of Victoria.

Malone, T. (1984). Logo - Process or product. In A. Maurer (Ed.), *Conflicts in Mathematics Education*, (pp. 355 - 357). Melbourne: Mathematical Association of Victoria.

Maynard, G. (1980). Implications of technological change in education. In P. Williamson (Ed.), *Learn to Love Mathematics,* (pp. 314 - 321). Melbourne: Mathematical Association of Victoria.

National Council of Teachers of Mathematics (NCTM). *An Agenda for Action: Recommendations for School Mathematics for the 1980s.* Virginia, United States.

Nevile, L. (1987). Recursion - Should it be avoided or does it liberate. In W. Caughey (Ed.), *From Now to the Future*, (pp. 237 - 240) Melbourne: Mathematical Association of Victoria.

Olsen, P. (1980). The use of computers in mathematics teaching. In P. Williamson (Ed.), *Learn to Love Mathematics*, (pp. 326 - 327). Melbourne: Mathematical Association of Victoria.

Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas.* The Harvester Press, Great Britain.

Papert, S. (1993). *The Children's Machine.* Basic Books, New York.

Pea R. D. & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. In *New Ideas Psychology,* 2(2), pp 137 - 168.

Pekin, M. (1984). Computer programming in business mathematics using a spreadsheet program. In A. Maurer (Ed.) *Conflicts in Mathematics Education*, (pp 355 - 357) Melbourne: Mathematical Association of Victoria.

Robson, P. & Thomas, G. (1980). Computers in the primary school. In P. Williamson (Ed.), *Learn to Love Mathematics*, (pp 365 - 367). Melbourne: Mathematical Association of Victoria.

Rosenberg, A. (1980). Practical work assignments for group I computer optional units. In P. Williamson (Ed.), *Learn to Love Mathematics*, (pp 334 - 335). Melbourne: Mathematical Association of Victoria.

Rule, T. (1987). Problem solving, computers and Logo Years 7 - 10. In W. Caughey (Ed.), *From Now to the Future*, (pp. 198 - 204). Melbourne: Mathematical Association of Victoria.

Salvas, T. (1980). Computers and your students. In P. Williamson (Ed.) *Learn to Love Mathematics*, (pp. 322 - 325). Melbourne: Mathematical Association of Victoria.

Salvas, T. (1983). Forward. In D. Blane (Ed.),*The Essentials of Mathematics Education,* (pp. 1 - 3). Melbourne: Mathematical Association of Victoria.

Smale, T. (1984). Visicalc in the mathematics classroom. In A. Maurer (Ed.), *Conflicts in Mathematics Education*, (pp. 368 - 373). Melbourne: Mathematical Association of Victoria.

Srivastava, D. (1984). Micro-computers and mathematics teaching. In A. Maurer (Ed.), *Conflicts in Mathematics Education*, (pp. 320 - 326), Melbourne: Mathematical Association of Victoria.

Thompson, L. (1980) Basic computer programming (Beginners), in P. Williamson (Ed.) *Learn to Love Mathematics*, (pp. 351 - 361). Melbourne: Mathematical Association of Victoria.

Turner, J. (1992). LogoWriter skills for the post-primary mathematics teacher. In M. Horne & M. Supple (Eds.), *Mathematics: Meeting the Challenge,* (pp. 440 - 444). Melbourne: Mathematical Association of Victoria.

Turner, J. (1991). The relevance of Logo as part of a Years 7 to 10 mathematics course. In J. O'Reilly & S. Wettenhall (Eds.),*Mathematics: Inclusive, Dynamic, Exciting, Active, Stimulating*, (pp. 407 - 413). Melbourne: Mathematical Association of Victoria.

Victorian Ministry of Education. (1988).*The mathematics frameworks P - 10.* Melbourne: Ministry of Education (School Programs Division).

Walker, R. (1991). The development of educational computing policy in the Victorian School System, 1976 - 1985. In *Australian Journal of Education*, Vol 35, No 3, pp 292 - 313.

Waterson, D. (1980). What should we study in computer studies. In P. Williamson (Eds.), *Learn to Love Mathematics,* (pp. 331 - 333). Melbourne: Mathematical Association of Victoria.

Wills, S. (1981). Turtles for teaching computing. In A. Rogerson (Ed.) *Maths Myths and Realities*, (pp. 346 - 363). Melbourne: Mathematical Association of Victoria.

Children, Computers, and Mathematical Ideas: Evaluating a Research-Based Version of Logo

Nicola J. Yelland

Queensland University of Technology

Douglas H. Clements

State University of New York At Buffalo

Jennifer E. Masters

Queensland University of Technology

Julie Sarama

State University of New York At Buffalo

When Papert (1980) wrote Mindstorms, he shared a vision of education in which the learner was both active and in control. He was interested in the ways in which computers could affect the way that people think and learn and offered Logo as a vehicle that would assist in this process, because

he thought that it could "contribute to mental processes not only instrumentally but in more essential, conceptual ways, influencing how people think even when they are far removed from a computer..." (p.4).

With turtle graphics, the idea of programming is introduced through the metaphor of teaching the Turtle a new word. In the process of programming the locus of control is with the child and not the computer. This is as significant now as it was in the 1980's, because there are still a preponderance of applications where the user is required to adopt a distinctly passive role (Clements, Nastasi, and Swaminathan, 1993). From its first implementation, it was apparent that young children liked the turtle, and that they engaged in activities with a high level of motivation and a deep level of concentration. Additionally, their interactions with the turtle enabled them to embark on "powerful kinds of learning" that involved consideration of mathematical content areas such as geometry, applications of number and the operations, algebraic relations (Clements & Meredith, 1993) and in some cases ventured into physics and explored such things as velocities. As Papert (1980) stipulated "They are learning to speak mathematics, and acquiring a new image of themselves as mathematicians" (p. 13). This was at a time when mathematics was often characterised as boring by children, because it tended to involve a) proving that you could "do" addition by repeatedly performing "sums" getting them right and not making careless errors that would scar your record, or b) engaging in abstract concepts that seemed to have no application to life out of school.

In the first instance, "Turtle geometry was specifically designed to be something that children could make sense of, to resonate with their sense of what is important ..it was designed to help children develop the mathetic strategy : In

order to learn something first make sense of it" (Papert, 1980, p. 63). While past versions and uses of Logo have liberated some children and teachers, the extent to which it has facilitated the liberation of learners, especially in the area of mathematics, is less clear (Yelland, in press). For example, while all versions of the turtle "talked mathematics," often students and teachers did not engage in that conversation (Leron 1985). In this paper, we will describe a new version of the Logo computer language designed to provide environments that are simultaneously learner-centred and mathematically rich. We will discuss what we learned in several classroom-based tests of this environment in both Australia and the U.S.

The Research Basis of Geo-Logo/Turtle Math[1]

The new version of Logo was based on research that has identified both the strengths and weaknesses of Logo as a tool for learning(Clements & Meredith, 1993; Yelland,1994a; in press). As an example of the latter, it was noted that students often use visually-based, nonanalytical approaches when directing the Logo turtle. Therefore, it was thought to be desirable that a new Logo environment should support students' mathematical development in particular ways (Clements and Battista, 1992; Clements and Meredith, 1993).

The research corpus provides directions for designing a Logo environment fine-tuned for the learning of mathematics. We will describe briefly four research-based principles and give examples of characteristics of the new Logo environment based on them.

1. Encourage construction of the abstract from the visual.

Logo can help children construct mathematical strategies and conceptions out of their initial intuitions and visual approaches. For example, there is a large literature on

children's difficulty with turns and angles. Children can build more robust ideas of these concepts using Logo because they give turn commands and receive feedback. However, if children's Logo experience is not mediated, they can maintain misconceptions (Hoyles and Sutherland 1989). *Geo-Logo* provides several measurement tools; for example, an on-screen protractor, placed at the turtle's position and heading, measures turns. One arrowhead shows the turtle's heading. The other follows the cursor, which students move with the mouse. When they click this arrowhead "freezes" and a turn command is displayed. Rulers and other measurement tools are also available. Further, we should remember that even if students do not adopt our goals, and use visual and empirical strategies, the environment should continue to support their activity. The visually-oriented measurement tools allow students to approach task in a wider variety of ways.

One of the main ways *Geo-Logo* supports the growth of the abstract from the visual lies in its overall structure, described in the following section.

2. Maintain close ties between representations

The nature of programming creates the need to make relationships between symbols (code) and drawings explicit. But students may lose the psychological connection between the two when involved in long projects. In *Geo-Logo*, students enter commands in "immediate mode" in a command window. Any change to commands in either location, once accepted, are reflected automatically in the drawing. The dynamic link between the commands in the command window and the geometry of the figure is critical. Also, the structure of the command window, long and narrow to the side of the graphics screen, instead of the

traditional short but wide placement below this screen. This permits the immediate inspection of more commands, which facilitates connecting symbols and drawings, as well as pattern searching. Further, students can easily modify the code, encouraging experimentation and supporting later work with procedures.

3. *Facilitate examination and modification of code; encourage procedural thinking. The environment should support easy creation, alteration, and use of procedures and highlight procedural-conceptual connections.*

The dynamic link also means that all commands in the command window represent a proleptic procedure. It can be defined as a procedure with a tool. A "Step" tool allows students to "walk through" commands to find errors or explore mathematical properties. Other palette tools allow easy editing and erasing.

4. Geo-Logo facilitates children's learning of mathematical ideas

In several other ways, *Geo-Logo* encourages students to build solid ideas about mathematics. For example, the Turn Rays option shows rays during turns. If you type rt 120, a ray is drawn to show the turtle's initial heading. Then as the turtle turns, another ray turns with it, showing the change in heading throughout the turn. A ray also marks every 30° of turn. *Geo-Logo* also provides coordinate grids, scaling of distances (i.e., "fd 1" can mean forward 1 cm or forward 1 inch, allowing interesting use of fractions), geometric motions via menu, mouse control, and textual commands (including slides, flips, turns and scaling).

Last but not least, *Geo-Logo* provides activities that are integrated fully into a coherent, innovatory mathematics curriculum project: The Investigations in Number, Data,

and Space project. Thus, the benefits of *Geo-Logo* go beyond learning geometry. Geometric models are critical for understanding numbers (number lines, multibase blocks), fractions, statistics, and other topics in mathematics as well as other subjects. In a similar vein, *Geo-Logo* investigations integrate many mathematical topics from mathematics and other subjects into *Geo-Logo's* geometric setting. For example, in the Geo-Face activity described later, students apply division and factoring concepts in a meaningful setting.

The Principles in Action

We have conducted several field tests of these activities in both Australia and the U.S. In this paper, we report our observations from a longitudinal study over a period of two years. We present two activities from the first year; the final two in the sequence: Geo Face and an open ended project, and an additional activity from *Turtle Math.*

In the first activity, Geo-Face, students design a face on the computer. They write procedures for each part of the face (e.g., mouth, nose). They choose what shape each part of the face might be, but each part has a perimeter given by the teacher.) In the final activity, children design their own project.

The Geo-Face activity was designed to consolidate the children's understanding and application of procedures for squares, rectangles and triangles. It was the sixth computer activity in the unit and took place in week 7 of the research. The instructions required the children to plan a face consisting of parts that were rectangles, squares and triangles according to the following rules:

1.there had to be at least one of each shape.

2.the perimeter of each part had to be:

nose 90
mouth 200
eyes 100
ears 120

3.the final command for each procedure had to be setheading 0 so that the turtle was facing up to the top of the computer screen.

The task was introduced as a class activity in which the researcher provided the children with worksheets containing the task instructions and a drawing page with the head of the face already drawn. The rules were explained and the class planned an ear together, as an example. In order to facilitate the planning process we developed a sheet with the following headings:

face part
drawing
perimeter
procedure

The children worked in pairs to plan their Geo-face before they entered each of their procedures on the computer and assembled the components. The format of the matrix enabled the children to consider each face part, make a drawing of the shape that they wanted it to be, refer to required perimeter, and finally record the procedure that would draw the item.

A high level of engagement with the activity was evident both at the planning and assembling stages of the task. Only one pair, Tim and Courtney did not follow the task demands explicitly and omitted to include a square in their face part. When entering the commands at the computer they decided

that the mouth should have been a rectangle because the square did not "look right." They created a mouth, eyes and ears that were rectangles and developed a triangular nose. The other pairs tended to create shapes that were somewhat predicted by the number assigned to the perimeter. For example, 100 and 120, can be easily divided into 4 equal parts of 25 or 30 respectively, or four sections to produce a rectangle.

Ryan and Aaron who had been adventurous all along in their development of ideas did their planning well away from the other groups so that no one could "steal" their ideas. When the other children came over to see what they were doing, as the children often did, they covered up their work so that it would remain a secret. Yet they constantly wandered around the room to check on what everyone else was doing. Ryan and Aaron decided that they wanted triangular eyes, and since the perimeter of each triangle had to be 100 they were forced to calculate the length of each side carefully. Ryan reasoned : "If it has to be 100 all the way around and the triangle has three (equal) sides, we need to have 100 divided by 3. Where is the calculator?" He entered 100 divided by three and exclaimed "Wow" when the answer appeared. He then asked if it was OK to use the number (33.333333) that appeared as the answer. He said that he knew the dot was a decimal and that the number was 33 but wondered if he had to type in all the other three's as well. We explained that he did not have to enter all the threes that were shown on the calculator, but it is evident from the completed procedure that he ignored the advice when the time came to typing in the instructions. Ryan was then keen to have another face part that included a decimal point so when he considered the nose he said that he wanted it to be a square so that he could divide 90 by 4 to get such

an answer. Accordingly his square nose had sides of 22.5. He seemed to be quite disappointed when he reconciled the task requirements to the remaining face parts and realised that he could not use this technique again. It was apparent, at this stage of the unit, that Ryan and Aaron had not only a good understanding of number in connection to the length of the sides of the shapes but also with the idea that turns for equilateral triangles were 120 and those for rectangles and squares were 90. Each of the procedures that were developed as face parts ended with a turn so that the turtle was facing the top of the screen as required. Additionally, as with all the pairs the triangle was drawn with the first command being rt 30 so that it would be "straight". The resulting geo face is shown in Figure 1. It was interesting to note that the plans for the geo face closely resembled the final product that appeared on the screen. This was not to be the case for the projects that were developed by the children in the final task, where the plans were always far more ambitious than the final product.

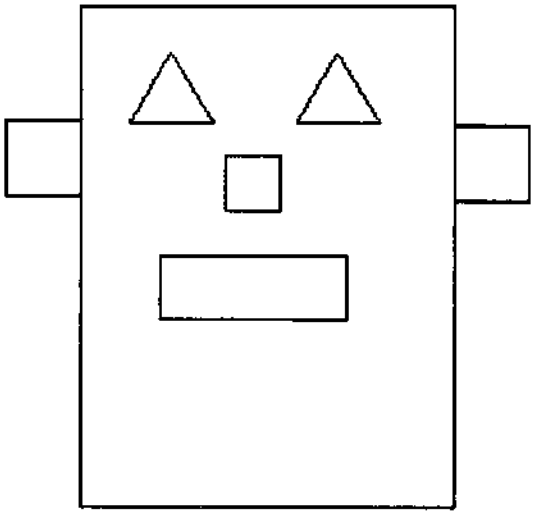


Figure 1: Ryan and Aaron - Geo-face

The assembly of the face was challenging for the children as they were introduced to the slide function for the first time and the physical skill of dragging the turtle to a new position and then sliding the face part into position initially required them to be quite dexterous with the track ball. Ryan and Aaron chose to assemble their face in the following sequence: head - eyes - ears - nose - mouth, as did most of the children, with only slight variations for the positioning of the nose and mouth.

The eyes were the only feature that did not have to be slid into place. The fact that the slide feature could be incorporated into the creation of the face was especially relevant at this stage of the sequence of activities, since it meant that the children could create the faces without having to continually manipulate each face part to exactly the correct location before executing the procedure. If they had been forced to seek the exact location for each part without the slide facility it may have led to frustration and abandonment of the task due to its tedious and repetitious nature. As such, the activity was highly successful and engaging for the children. They revealed a deep understanding of number and its role in the development of component features that were planned, developed and entered using *Geo-Logo*, in order to produce a completed "picture". This was evidenced in the final product but also at the planning level, off the computer.

Projects

In the final session of the unit the children designed projects of their own and in doing so it was apparent that they were:

- analysing geometric figures in order to determine their role/ place in the final product

- understanding that shapes can be moved to new locations, and flipped and turned without losing their essential properties, that is, the angles in a square were always 90 even when the square was tilted.
- using their mathematical knowledge, especially related to number and operating on them to produce length and turns for different functions.

At the planning stage the planning sheet developed in geo face proved to be particularly useful in assisting the children to organise their ideas in a coherent form. It also served the function of helping the children to decide what constituted a viable project. At first when they made elaborate drawings they did not appear to recognise how difficult they would be to develop as geo-logo projects. However, when they came to record their ideas as component parts and procedures it became immediately apparent that the plans would have to be considerably modified in order to enter them as code. The only pair that did not develop a collaborative project was Tim and Courtney, who chose to develop one each. All the other pairs worked together over four sessions in order to produce their project pictures. The results, as previously stated, indicated a sound understanding of basic mathematical ideas together with a well developed skill in programming involving the development and combination of procedures. In developing their House (Figure 2) Ryan and Aaron decided that they needed to use a semi circle in order to put some domes on the roof section. After thinking for a while they remembered that Angela and Denielle had incorporated a circle in one of their projects. They asked if they could go into the girls' file to "have a look at it". Once the circle was located the procedure was copied, their project re-opened and the

"borrowed" procedure was pasted into their own command centre. When they realised that it was too large they modified it to suit the size of their roof. The incident is interesting not only for the bravado of using the procedure that another pair developed but also because the logistics of entering, copying and pasting files was only done as a management aid while the research was being conducted, yet the children had observed it, and then used it when they needed to do so. The House project also reveals the kind of detail that the children were able to develop in their drawings. The security camera was created very carefully by the pair, using a top down approach. Initially, the drawing of the camera was very complex. When we suggested that this would be difficult to draw on the computer the boys made new plans that were much more simple and contained the basic shapes that they knew they could draw. This was a very practical approach and they then considered each shape separately and built up the camera before placing it on the house.

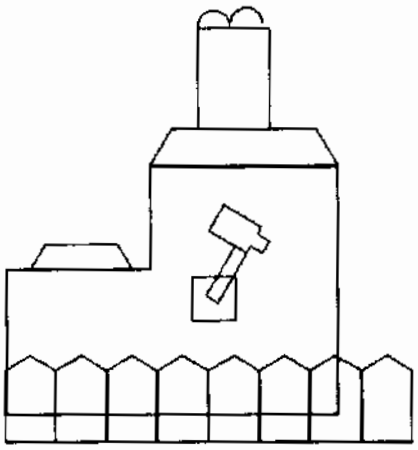


Figure 2: Ryan and Aaron - security house

It was interesting to note that in the geo face activities the children were using the "turtle turner" to *check* the amount of turns not to *make decisions* about the size of the turn. This proved to be an extremely valuable tool for the development of items that formed the components of the project.

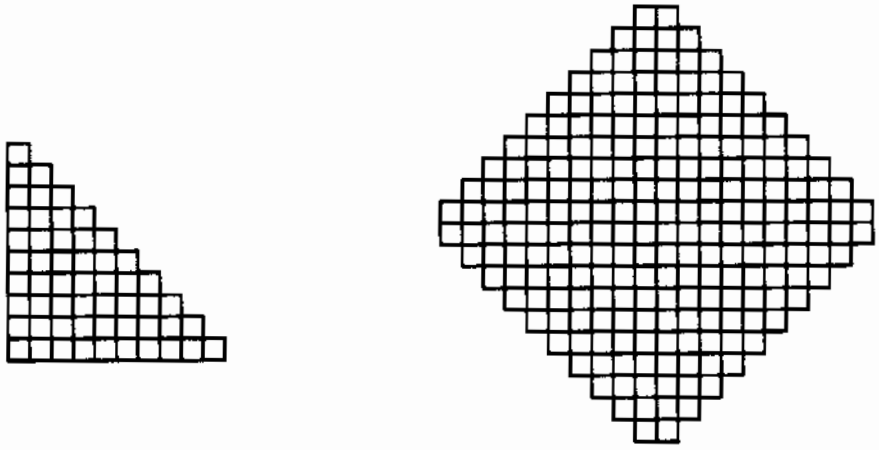
At this stage of the research the slide facility was not operating in the Free Explore mode so the children developed other strategies to assemble to parts of the projects. We demonstrated how to position the turtle at the appropriate place in the picture and explained that "jump to" (a coordinate command) could not be used since it would mean that the component would always be drawn in that position irrespective of where the rest of the drawing was. The children were able to incorporate this strategy into their project assembly and we felt that this was only possible because of the power of geo logo to accommodate changes at a micro level so that the picture could be modified without the whole code being changed.

Rectangle Patterns

In the second year, we let the children explore the activities contained in *Turtle Math.* We started by playing two games, ice hockey and whale, which required the pairs to direct the turtle to score a goal in the first game and reach a whale in the second. The children had not played with computers at school in the interim 8 months. They immediately recalled the commands and the tools that would enable them to carry out the tasks effectively. The most relevant of these was the turtle turner. This facilitated the decision making in relation to the size of turn that was needed, in 30 degree sections.

In the second session, we created a procedure for a rectangle as a group, and then introduced a *rect* procedure with variables. The children discussed the impact of entering

different numeric inputs on the size of the rectangles and were asked to use the procedure to create patterns presented to them on paper (Figure 3). This resulted in some interesting conversations about the nature of the numbers that were used and the relationship between the numbers and the rectangle pattern that was created. One pair, Cody and Shaun, developed their ideas about rectangles in an interesting way, by creating their own pattern with rectangles (Figure 4). They were interested to discover that they could use a shape that had straight sides to create a pattern that had curves in it, and speculated about how it happened.



Use the **rect** command with two

inputs to draw this...and then draw this.

Figure 3: Rectangle Challenge Activity

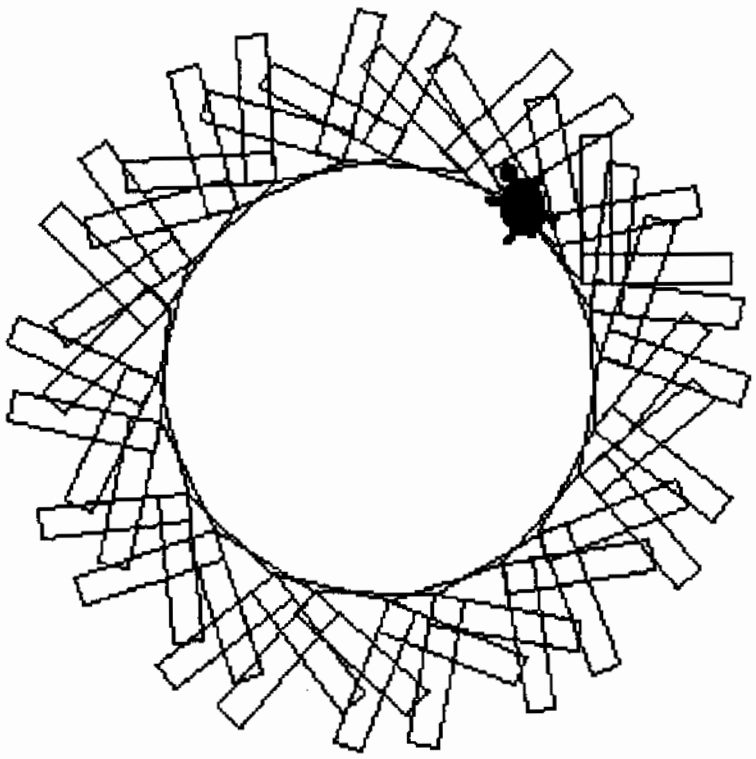


Figure 4: Cody and Shaun - rectangle pattern

Conclusions

In Mindstorms, Papert (1980) talked of learning in the context of a Brazilian Samba School. The school is characterised by a sense of social cohesion, belonging to a group of multi age learners and having a sense of common purpose. Within this context, teaching occurs in a natural, although purposeful way, whereby an expert works with a small group about a particular aspect and then they return to the main group. Papert (1980) likened the teaching of mathematics with Logo to such schools. Certainly, we found that the instances of learning and engagement with tasks in our environment had characteristics of social cohesion, shared ideas and both directive and reciprocal teach-

ing opportunities for all participants which seemed to warrant its classification as a "samba school for mathematics" (Papert, 1989, p.182). Additionally, it highlighted the relevance and supported the notion of creating a community of practitioners (Lave and Wenger, 1991).

More recently Papert (1993) suggested that "The central problem for math education is to find ways to draw on the child's vast experience of oral mathematics." (p.16) We have found evidence to support this in this data and other research (Yelland and Masters, 1994) in a context where children were learning and using mathematical ideas and principles with a sense of purpose and energy. We support the notion articulated by Papert (1993) that "Giving children the opportunity to learn and use mathematics in a nonformalised way of knowing encourages rather than inhibits the eventual adoption of a formalised way..." (p.17).

Our observations support the efficacy of both the principles abstracted from the research and the design decisions based on them. For example, the provision of mathematical tools such as the slide tool (a) provided experiences with extended concepts such as geometric motions, (b) permitted children to use individual procedures for each shape (which both encourages the use of procedures and encourages reflection on the connections between an individual shape's properties and the mathematical commands that generated it), while still (c) allowing children to complete projects that may have otherwise been frustrating. This led to the children's developing a sound understanding of geometric concepts and a high level of programming skill for their age. Other tools such as those for turn measurement contributed to the development of concepts and skill in those areas as well by helping children use their visual and kinesthetic approaches to find measurements.

Geo-Logo's structure also was beneficial to these students. Even, for example, such simple changes as the structure and placement of the command window were significant in supporting children's learning of mathematics. Most of the earliest microcomputer versions of Logo—for example, those for the Apple II line—had 4 lines of text beneath a graphics window. This was necessary due to the built-in hardware features of that machine. However, later versions tended to have similar arrangements (another QWERTY phenomenon, ironic for Logo). *Geo-Logo*'s command window, long, narrow, and placed to the side of the graphics screen allowed children to observe more commands, which facilitated their linking symbols and searching for patterns. Furthermore, because students can easily modify the code, it encourages them to take risks and experiment with their ideas and supports later work with procedures.

More signficant is *Geo-Logo's* built-in connections between the symbolic code and the graphic. When students change their Logo code, a corresponding change is made in the graphics, automatically. Our observations of children working in the environment indicate that this is the most powerful feature of the new version. Its dynamic nature enables the child to respond immediately to feedback from the graphic and thus enables them to continue on a route that was closely aligned with the original plan. The connections also facilitated children's connection of spatial and numeric representations, and encouraged exploration and planning. This stands in contrast to research projects that used Logo without such features. Frequently, these researchers reported a lack of planning; students changed their plan to match what their code produced, or abandoned the initial plan altogether (Clements and Merriman 1988; Noss 1984; Yelland 1994b). Students may evince these kinds of behaviours less because

they are poor planners or because they are programming in Logo and more because they are programming in a Logo environment that makes it difficult to make changes. This should not be taken as implying that students using Geo-Logo followed their plans exactly. In many instances, students' planned projects were more ambitious than they could realize. However, *Geo-Logo*'s structure allowed them to simplify just enough to finish their project without relinquishing their original plan.

The tasks were also relevant in this regard. For example, the restricted goals of geo face helped children properly limit the scope of their projects and also abstract basic geometric figures from complex objects.

The realization of plans was also aided by a combination of the erase one tool and the connections between symbols and graphics. For example, if children entered rt 990 instead of 90, they could just erase the last command. They do not have to, as they might have in previous versions of Logo, find the heading and subtract from it from 360 or add extra code such as lt 990.

The ease of making such "instant changes" did not make use of procedures less likely. Indeed, *Geo-Logo* makes the task of programming with procedures easier and achievable for young students . In older versions of Logo, children often restricted to using immediate mode or were taught how to use procedures didactically. The Teach tool assisted students in understanding procedures: defining them, changing them, and, in general, using them as tools for thinking and planning.

Finally, the evidence of exploratory activity argues that Logo, and especially *Geo-Logo*, provides powerful *mathematical experiences* - In these ways, we believe there is

consistent evidence that enhanced version of Logo can liberate the young learner of mathematics. *Geo-Logo* provides additional tools with which to think, and empowers young learners so that they can realise the possibility of "intimate contact with some of the deepest ideas from science, from mathematics, and from the art of intellectual model building" (Papert, 1989. p. 5).

Note

[1] 1994. D.H. Clements LCSI All rights reserved.

References

Clements, D. H., & Battista, M. T. (1992). Geometry and spatial reasoning. In D. A. Grouws (Ed.), *Handbook of Research on Mathematics Teaching and Learning* (pp. 420-464). New York: Macmillan.

Clements, D.H., & Meredith, J. S. (1993). Research on Logo: Effects and efficacy. *Journal of Computing in Childhood Education, 4*, 263-290.

Clements, D.H. & Merriman, S.L.(1988) "Componential developments in Logo programming environments." In R. Mayer. (Ed.), *Teaching and Learning Computer Programming: Multiple Research Perspectives*, (pp 13-54). Hillsdale, NJ: Erlbaum.

Clements, D. H., Nastasi, B. K., & Swaminathan, S. (1993). Young children and computers: Crossroads and directions from research. *Young Children, 48*(2), 56-64.

Hoyles, C., & Sutherland, R. (1989). *Logo Mathematics in the Classroom.* London, England: Routledge.

Lave, J. & Wenger, E. (1991) *Situated Learning: Legitimate Peripheral Participation.* Cambridge: Cambridge University Press.

Leron, U. (1985) "Logo today: Vision and reality." *The Computing Teacher*, 12,26-32.

Noss, R. (1984) Creating a mathematical environment through programming: A study of young children learning Logo. London: The University of London, Institute of Education.

Papert, S (1980) *Mindstorms: Children, Computers and Powerful Ideas.* New York: Basic Books.

Papert, S (1993) *The Children's Machine: Rethinking Schools in the Age of theComputer.* New York: Basic Books.

Yelland, N.J. (1994a) A case study of six children learning with Logo. *Gender and Education, 6*(1) 19-33

Yelland, N.J. (1994b) The strategies and interactions of young children in Logo tasks. *Journal of Computer Assisted Learning,10,*33 - 49.

Yelland, N.J. (in press) Mindstorms or storm in a teacup? A review of research with Logo. *International Journal of Mathematical Education in Science and Technology.*

Yelland, N.J. & Masters, J.E. (1994, December) *Innovation in Practice: Young children learning in a technological environment.* Paper presented at the Australian Association for Research In Education Conference, University of Newcastle, NSW.