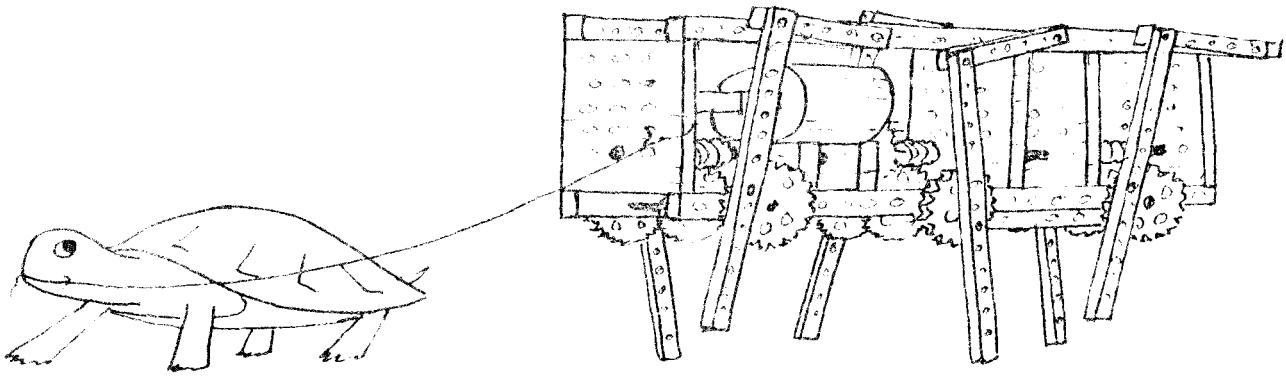
POALL

A Journal for Logo Users



Walking Machines

Volume 3 Number 3 August 1988

LEGO continues to be 'flavour of the month' with the recent visit of Steve Ocko and the imminent release of TC Logo. We've been privileged to have a review copy for several weeks, and it works well. There's a review in this issue, with a reference sheet that you may find useful to copy for students. Our lead article is also LEGO based.

There's another of James Muller's articles from the now defunct *Turtle News*, good stuff, but I wish there was more local material to publish.

The changeover from SACAE Magill to Plympton led to a few minor hiccups with distribution of the last issue, which we hope are now sorted out. Look at your mailing label and you'll see that there's a completely new mailing list system. At top right of the label is the number of the issue to which you are paid, which should make it easier to keep up to date.

Peter

POTS

Stepping out...

Legged Robots in LEGO

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.

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 grond 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 that below:

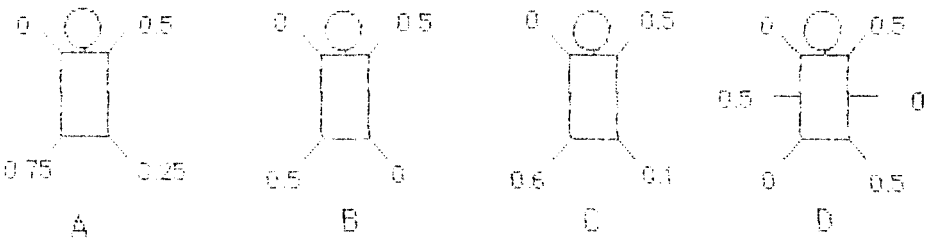


Figure 1 A Amble B Trot C An efficient Turtle gait
D Alternating tripod gait for hexapods

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 was being designed 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, manoeuvering 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 dumbell 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'[1]. 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 (there's a project for someone), 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.

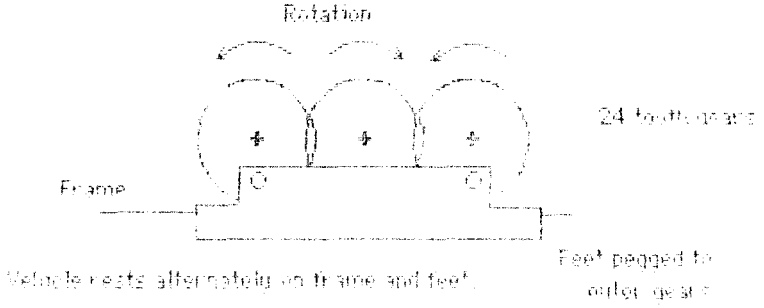


Figure 2 Basic walking mechanism

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. Take care when assembling the drive shaft that the legs are properly phased.

The machine was surprisingly stately in its movements, gently rising and falling as it progressed. 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. You can experiment with legs of different lengths, and perhaps with 40 tooth gears instead of 24.

---

[1] We're making no attempt to reproduce the LEGO method of diagramming models. You'll have to imagine the fine details.

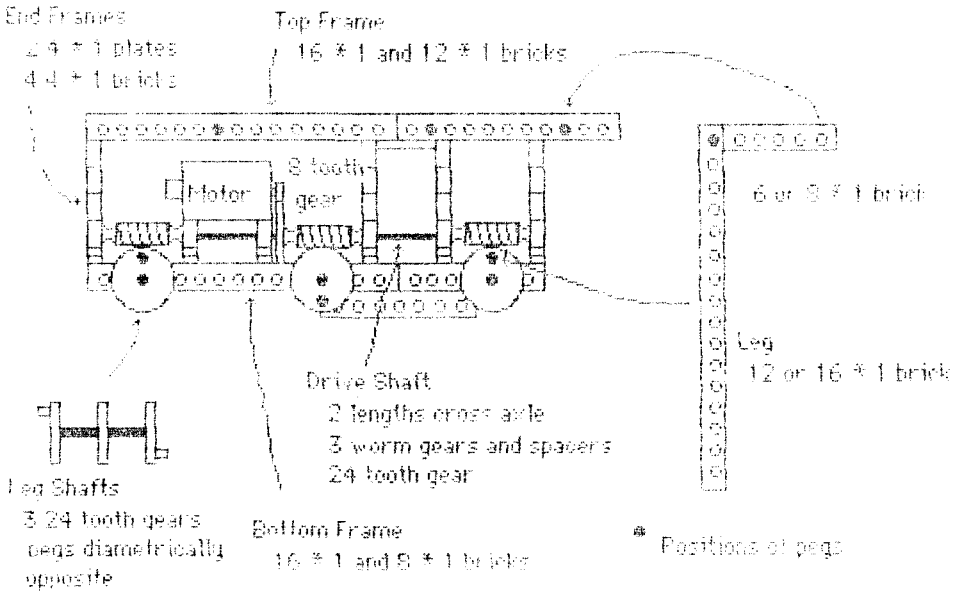


Figure 3 Hexapod 1

Another leg mechanism we 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, an extra degree of freedom is needed in each leg, in other words, another hinge and more mechanism. Another possibility would be to connect two or more such machines as a train, with universals connecting their drive shafts, and steered by 'bending' the train with a separate motor and linkage.

Legs like these can be used for quadrupeds, and our 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.

When I was very young I was given a toy tortoise, a '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. It can hardly be called real walking, perhaps shuffling or crawling, but worth trying.

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

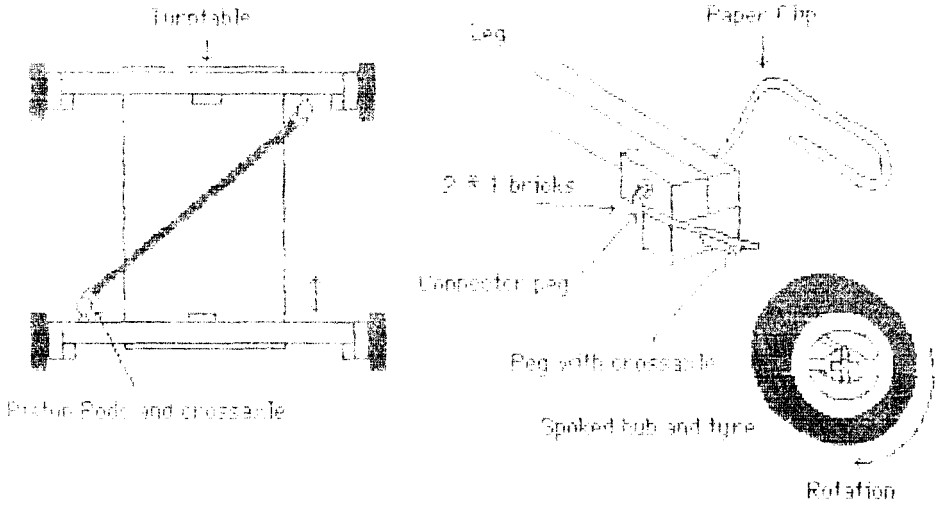


Figure 4 Shuffling vehicle with ratchet wheels

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 should be 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 would require 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.

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

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

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

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

Pearson, K. and Franklin, R. '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 1984

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

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

Todd, D. 'A Pneumatic Walking Robot', in *Robotics Age*, January 1985

Todd, D. *Walking Machines: An Introduction to Legged Robots*, Kogan Page, 1985

Waldron, K. *et al* 'Configuration Design of the Adaptive Suspension Vehicle', in *The International Journal of Robotics Research*, Vol 3 No 2, Summer 1984

Steve Ocko visits Adelaide

BigTrak and the MACOS films are but two of the many achievements of Steve Ocko. The one-time architect, film maker and toy maker is now Research Associate at the Media Laboratory of MIT (Massachusetts Institute of Toys), a colleague of Seymour Papert. Hosted by CEGSA, Steve was in Adelaide July 10..13th.

On the Monday, Steve visited Angle Park, where he was interested in NEXUS, which it seems, is well ahead of anything he had seen before. Overall, it appears we share the same sorts of problems with educational computing, and many other aspects of teaching, as our counterparts in the US. They too, are hampered by a lack of freedom and the threat of 'accountability'.

Aberfoyle Park High School was the venue for the next morning's activity, a LEGO/Logo session with nine students and a number of observers. They set about with enthusiasm to build and program a number of models: an alligator with snapping jaws, a double lifting bridge, and a polar axis robot arm. They had all had some experience with LEGO and Steve showed them the essentials of TC Logo so that their models operated, although perhaps not quite as they had hoped. Part of the philosophy of LEGO/Logo is the debugging process. Meanwhile, several of the observers built themselves a paper crimping machine.

In the afternoon, Steve visited the LEGO exhibition at John Martin's, in through the back way courtesy of Di Jackson, LEGO's South Australian educational representative. He was fascinated, and impressed by the scale of the models and the level of animation.

The evening meeting was at SAIT, and we watched a video tape of work at the Hennigan School, as well as footage of some intriguing LEGO 'animals'. Hennigan School in Boston is in the sort of area that even taxi drivers refuse to go: it has all the social problems of a deprived inner city area. However, it has, through the generosity of Big Blue, 100 PC Juniors for 200 hundred students, and because of them and the use of LogoWriter, a vibrant computer culture where children regard the computer as essentially a powerful pencil. It was into this environment that LEGO/Logo was introduced. As the tape showed, the results are impressive: children working creatively together, discovering and exploring solutions to the problems that they have set for themselves, and developing positive attitudes to school and each other. The changing attitudes are not limited to students; parents too become enthusiastic about school, many of them for the first time, and attendance at parent meetings is very high. Those who have been through the program and moved elsewhere have continued to show the beneficial effects, even in school environments that place the usual restricting stress on testing and rote learning.

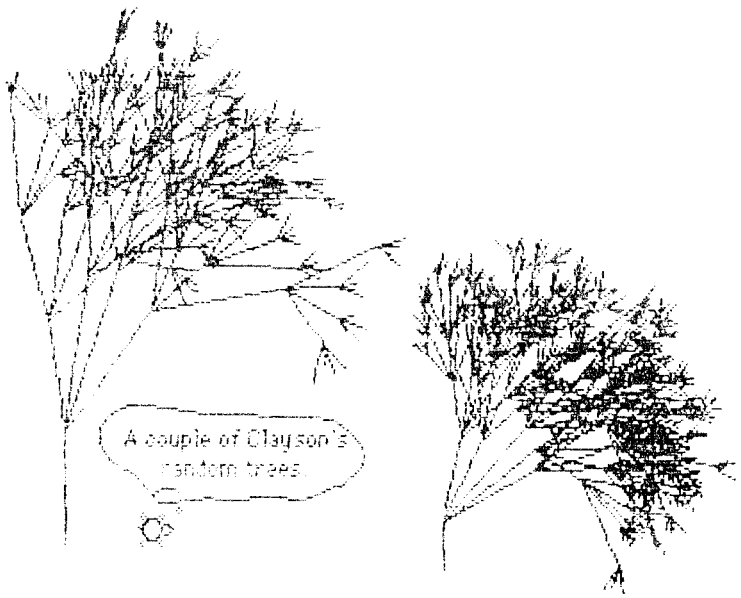
The videotape was aired again the next morning when Steve visited SACAE Magill and discussed the work at Hennigan and MIT, with a live demonstration of a line following buggy and other machines.

Steve described the progression in LEGO/Logo work, from children testing their theories by rolling vehicles down ramps, through small machines, to impressively complex ones. The series of animal machines had come about after children had studied real creatures and their movements; there were machines that crawled, others that jumped, one that extended and contracted like a worm, and another that shook itself across the table. Along the way, the children had had to think critically about the advantages and disadvantages of legs and wheels, the role of sensors and senses, and the need for coordination with computers or brains. They learned at first hand the limitations of technology.

LEGO/Logo is deliberately designed for primary age children; it is not robotics in the high school computing studies sense, and it is also designed to appeal to girls as well as boys. Throughout, there is 'heads in' as well as 'hands on', and children are encouraged to record their work in 'inventor's notebooks'. Some classes even set up their own patent offices. Children often take over the role of teacher, guiding their peers and younger students, and Steve described how some children had built 'fraction machines' to teach fractions to children in lower grades. Children also learned to be comfortable about rejecting the ideas of teachers, just as many teachers had to be prepared to let students take control of much of their own learning. There was a new vision of what it meant to make a mistake, and many children branded as 'failures' because they could not cope with verbal learning found new skills and self esteem as they were able to express themselves through their spatial and mechanical abilities.

Steve shares a quality that I have seen in other members of the MIT Logo team; a deep commitment to children and their education in environments where they are free to develop at their own pace, free to use the power of modern technology in interesting and constructive ways, and free to develop positive social interactions. At a time when others are suggesting that teachers be replaced by machines, it is reassuring that there are people concerned not just with using high technology, but with enhancing human values in schools through that technology. It was stimulating and encouraging to meet and hear him.

Special thanks are due to LEGO Australia for bringing Steve to Australia, to Anne A'Herran for her organisation, and to Phil Roach and his family for their hospitality.



Sanders' Paradox: Given the rate of production of LEGO bricks, why aren't we swimming in them?

LEGO TC Logo

In simple terms, LEGO TC Logo for the Apple is a version of LogoWriter with some new primitives, and a few missing. The possibilities it opens up are enormous, as it reaches back, in some ways, to Logo's roots, with the concrete 'object-to-think-with'. This time, however, the concrete objects are LEGO constructions.

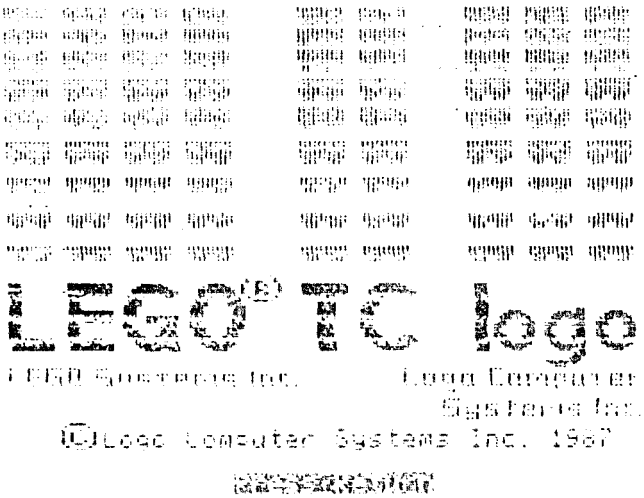
LogoWriter, and hence TC Logo, combine the power of Logo with the ease of use of a word processor. Unlike a conventional Logo with its command mode and editor, the new versions use the concept of a 'page'. On one side things happen; the Turtles (there are four) scurry about, text is printed among the graphics, the music plays and so on. The other side is the editor, and one 'flips' between one and the other with ⌘-F; none of the ED "wotsit business. In word processing mode there are the necessary tools, cut, copy and paste, while the Logo side has redefinable Turtles , music and list processing facilities. Loading and saving are virtually automatic, with loading reduced to selecting a page from a menu, the CONTENTS page. Overall, much easier and more intuitive than earlier Logos.

TC Logo's new primitives are few and easily understood, and are designed to control the LEGO 9750 interface. The following page gives an overview of the system. The emphasis is on addressing the LEGO machine and not the bit-twiddling that might be engaged in by high school robotics students. Most primitives have abbreviations, eg. TTO for TalkTo.

TC Logo will be available in two packages in September. Set 9545 includes a master disk, an operating disk, reference manual and teacher's and students' guides, set 9549 is a set of 4 operating disks. The manuals are well presented, with good ideas and suggestions to lead to further exploration. There will be a version for BBC users in the near future, but versions for other machines are not expected for some time. That does not mean that you can't use LEGO and Logo together: an article in *POALL* Vol 3 No 1 explains how it's done.

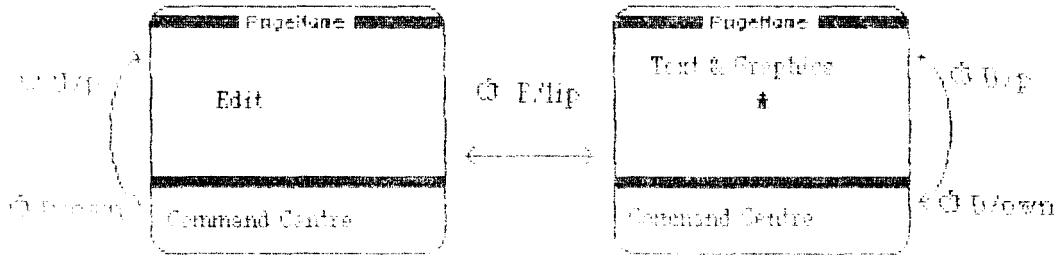
On the hardware side, there will be a new kit, Technic Control 0 (9700), which contains 2 motors, an optosensor and two touch sensors, to supplement the existing Technic Control 1 and 2 sets. For those with existing Technic sets, the sensors and motors are available as spare parts.

LEGO and Logo work well together, and enable children to actively explore many important concepts. TC Logo is the language to easily tie the two together.



A Quick Guide to LEGO TC Logo

The Screens and Cursor Movement:



O S/top

The command centre may be scrolled with the ↑ and ↓ keys.

CC	Clear command centre
ClearText	Clear text from graphics screen
Print (PR)	Print text to graphics or edit screen, whichever is visible
Show	Print text to command centre, with [] and carriage return
Type	Print text to command centre, without carriage return

Turtle Operation:

For multiple turtle use, see LogoWriter manual, otherwise TC Logo is like any other Logo, but note:

CG	Clear graphics screen

SetC SetBG Not implemented

List Processing:

As other Logos, but note that reporters (predicates) end in ?, eg. **Empty?**

LEGO:

TalkTo (TTO)	Address output port(s), eg. TalkTo [A B]
On Off	Turn output(s) on or off
AllOff (AO)	All output off
SetEven SetOdd	Set direction of output port(s)
ReverseDirection (RD)	Reverse direction of output port(s)
OnFor *time*	Turn output(s) on for specified time
ListenTo (LTO)	Address input port(s)
Sensor?	Report sensor true or false
WaitUntil *condition*	eg. Waituntil [Sensor?]
Counter	Count pulses from sensor, eg. Waituntil [Counter > 50]
ResetC	Zero counter

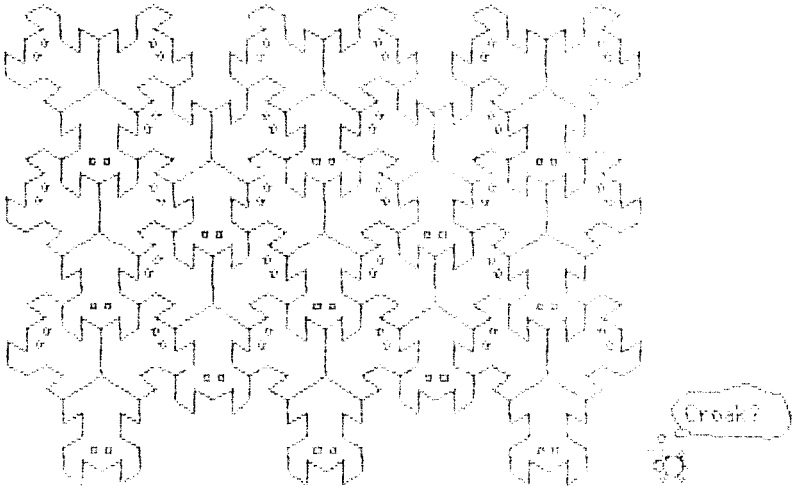
File Operations:

NamePage "*name* (NP "*name*)	Name the current page
<Esc>	Close session, saving current page
Contents	Save current page, display disk catalogue
PrintScreen	Print text and graphics to ImageWriter
PrintText PrintText80	Print all text on visible page to ImageWriter

Resources

Two books on exploring the visual this time, covering similar fields in some ways, but for two different 'markets'.

The first is *Tessellations Using Logo* by Margaret Kenney and Stanley Bezuska of the Mathematics Institute of Boston College. Their book is aimed at older primary and junior secondary students and is designed as a progression of exercises, beginning with discussion of polygons and extending to patterns like those used by Maurits Escher, like this frog pattern:

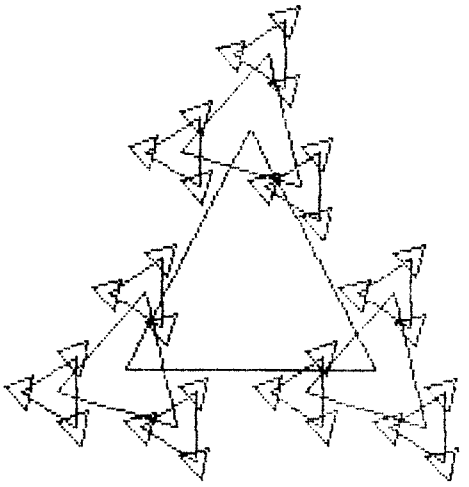


Although there are many procedures listed in the book, including answers to problems, the emphasis is on students experimenting. For most of the topics there is explanation, followed by challenges to write the procedures to draw the given pattern, and the topics are in a logical progression. After the expected experiments with tiling triangles and hexagons, Kenny and Bezuska move on to mixed tessellations, pentominoes and Islamic patterns.

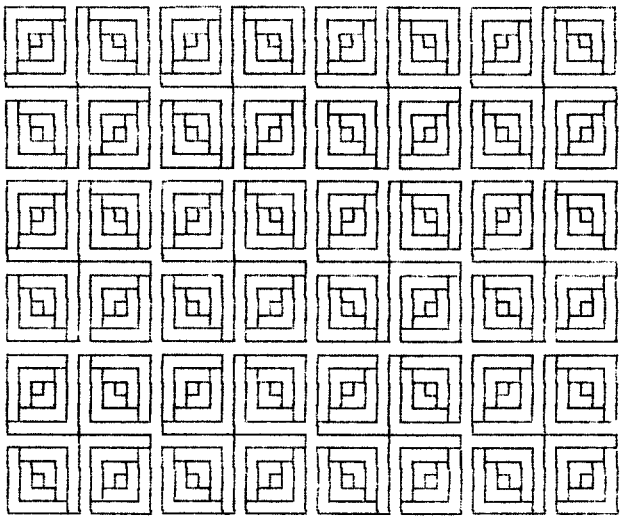
The style of Logo is a little odd in places, with single letter variable names, and the authors admit to preferring REPEAT to recursion. Nevertheless, the procedures all seem to work, and the book is an interesting introduction both to Logo and the world of tessellation and could be successfully used by newcomers to the language. We're using it at Plympton with a Year 9 class and some Year 8 students, who seem to have taken to it well.

Visual Modeling With Logo by James Clayson is the second volume in MIT's new Exploring with Logo series. Clayson is an artist, working in Paris, and his book is one for artists, not mathematicians or computer scientists. Set the task of teaching a course named 'Problems in Visual Thinking' his problem '...was to discover how to teach visual thinking to those students who had problems doing it naturally.' He decided that the best approach was for the visually oriented art students, generally lacking in analytical skills, and the mathematically inclined group to be combined, with 'the Logo language...as an appropriate medium of instruction-just enough of the visual and just enough of the analytical.' The book is, in effect, the class notes from the course.

Clayson begins with discussion of Logo, with the warning that the book is not a Logo text, and this interesting comment: 'I am saddened to think of you learning BASIC before Logo, and this book will give you the chance to right that terrible wrong.' Nevertheless, there's enough explanation to guide the newcomer to Logo to be able to do interesting things, and tinkering with the procedures in the book is positively encouraged. Polygons are the first topic, not just their geometry, but their arrangements to make interesting images:



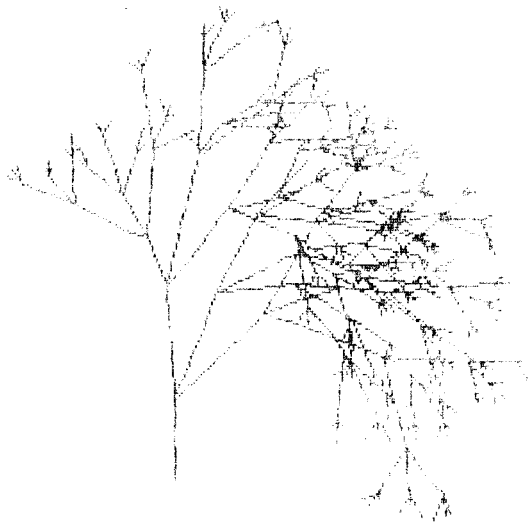
There is considerable space given over to patterns on grids, both regular and random. The pattern below is based on Mayan designs, and there is a whole chapter devoted to Islamic patterns.



Recursion appears early in the book but there isn't any discussion until about a quarter of the way through:

'Recursion is the most elusive concept in this book. I believe it is also one of the most powerful because it provides you with an entirely new metaphor for visual thinking. Working and thinking abour recursion will encourage you to look-think at your world differently. In a strange way, recursive thinking gives you the power to *make* the world look different.' (p 97)

Clayson's explanation, naturally a visual one, is as good as any, and leads the way, eventually, to fractal trees. There happens to be an error in a procedure as it is printed on p 306, so here it is, with its subsidiary procedures so that you can experiment for yourself, remembering that the REPEAT line is one line.



```
TO RANDTREE :size :number :angle1 :angle2 :scale :level
IF :level < 1 [STOP]
LOCAL [position dist]
RecordPos
FD RRandom ( 0.5 * :size ) ( 1.5 * :size )
RT RRandom ( 0.5 * :angle1 ) ( 1.5 * :angle1 )
REPEAT :number [MAKE "dist RRandom ( 0.25 * :size ) ( 1.5 * :size / 2 )
   FD :dist
   RandTree ( :size * :scale ) :number :angle1 :angle2 :scale :level - 1
   BK :dist
   LT RRandom ( 0.5 * :angle2 ) ( 1.5 * :angle2 )]
RestorePos
END
```

```
TO RecordPos
MAKE "position LIST POS HEADING
END

TO RestorePos
PU
SETPOS FIRST :position
SETH LAST :position
PD
END

TO RRandom :lo :hi
OP :lo + RANDOM ( 1 + :hi - :lo )
END
```

What's a tree?

A recursive definition: a tree is a branch attached to a tree. (p 282)

Clayson quotes Constable: 'Painting is a science and should be pursued as an inquiry into the laws of nature. Why, then, may not landscape painting be considered as a branch of natural philosophy, of which pictures are but the experiments?' and this sets the whole tone of the book, a fascinating exploration of the visual.

Bibliographic Information:

Clayson, J. *Visual Modeling with Logo: A Structural Approach to Seeing.* MIT Press, 1988, 400 pp (Copy from MIT Press, $US19.95)

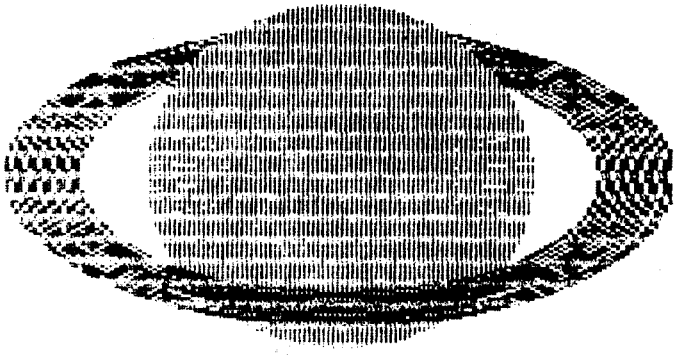
Kenny, M and Bezuska, S. *Tessellations Using Logo.* Dale Seymour Publications, 1987, 90 pp (Copy from Scientific Supplies P/L $19.95)

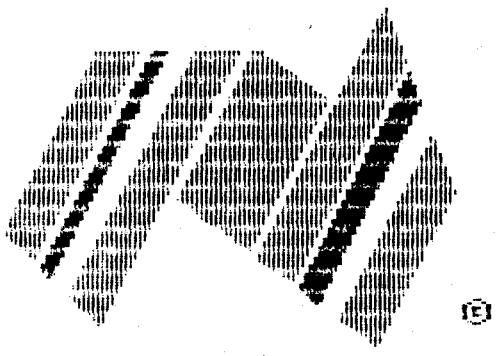
Logo ... Should it be taught? Should it be learned?

by James H. Muller, President
Young Peoples' Logo Association

Logo *is* one of those words which conjures up a variety of reactions within the educational community. It has been classed as everything from the savior of the educational system to just another fad.

But what is Logo? Is it something to be taught? Or should it be learned? Can the learning of Logo be effectively integrated into a standard school curriculum? Or does it require another approach? Is there a wrong way to teach Logo? Is there a right way to learn Logo? As the use of computers and Logo increase at all grade levels, it would serve us all to explore these questions.





Logo is a computer language and a lot more. More than anything else, the Young Peoples' Logo Association has found it to be an adventure into the fascinating world of the imagination. Think of it as you would the game of chess. Very young children can learn how the different chess pieces move and then begin to explore the game through straight, linear games; one move independent of the next. Soon strategy begins to enter the game, often unconsciously. Players soon find chess to be a battle of wits, an endless adventure in strategic opening, middle, and endgame combinations.

Very young children can quickly learn the Logo primitives. Then, as with chess, they can spend many lifetimes exploring the seemingly infinite combination of primitive commands. Starting with the most simplistic turtle graphics commands of FORWARD, BACK, LEFT and RIGHT, children can discover increasingly complex two and three dimensional graphics, the manipulation of lists, the exactitude of mathematical logic, and the fascinating ambiguity of linguistics.

On the surface, it would appear that Logo may well be the emancipator some people have tried to make it. But is that really the case?

The development of Logo has been chronicled by innumerable editors, writers, software developers, salesmen, educators, and others. Now it is far more interesting to look at where and how did it get where it is today?

The acceptance of Logo is very much like the acceptance of many other consumer electronic products: hi-fi systems, CB radios, microwave ovens, video games, electronic toys and household appliances. When a new technology is introduced, first come the hobbyists. These few innovators are more interested in exploring the technology than in exploiting it. With curiosity and imagination, they expand the application of these new products. Soon, the exploiters enter the scene directing their hi-intensity promotions to the fringe element who will pay any price to be the first on the block with the newest piece of "hi-tech gee-whiz." Then come the mass-

(continued)

marketers who tend to drive the price down, anxious to establish their products as the standard of the industry.

One of the more visible examples is the Polaroid camera. First there were the large, clumsy, and expensive Polaroid Land Cameras offering instant pictures followed by seemingly instant fading of the prints. As the technology was perfected, these early models were followed by the sleek, brushed stainless steel and leather models styled and priced to recoup the development costs of the technology. Now we have the plastic and vinyl versions priced for the mass-market. And all of this happened within one company.

Until those such as Forth and APL addicts can be shown the value of converting to Logo, it will have no value as a programming language.

Logo first appeared on the home computer market as a limited, high-priced, memory intensive language of interest primarily to educational innovators and very young children. It had none of the power of current versions. Competitors soon emerged with other versions attempting to capitalize on the publicity generated by the innovators. Now there are more than 20 versions fighting it out in the marketplace. Sometime soon, probably in 1985, one syntax will be confirmed as the standard. Others will either conform or drop out of the market.

As the more popular versions of Logo battle for dominance in the marketplace, the developers are working to make Logo more useful as a practical, general-purpose programming language. Some first steps have been made with recent versions of the language. But until the introduction of a good Logo compiler and the expansion of the general programming capabilities of Logo, there is no incentive to develop commercial Logo programs. Until those such as Forth and APL addicts can be shown the value of converting to Logo, it will have no value as a programming language.

This raises two questions. if there is no foreseeable job market for Logo programmers, why bother teaching it? More fundamental, considering the capricious syntax of the language, is the quesiton of whether it is even worth the effort to try to develop Logo as a competitive programming language.

Seymour Papert and others do not favor teaching Logo as merely another programming language. He sees Logo, the computer, and mathematics as tools for thinking. His concept of the computer goes far beyond its utilitarian value as a word processor, audio visual aid, or high-priced calculator. For Papert and others, Logo and the computer are far more personal.

It is obvious that few understand the application of these personal tools. Otherwise there would not be the flood of texts, tutorials, and curriculum guides which present Logo as just another programming language. If these authors understood the technology, they would not continue to teach Logo and computer-literacy as they have taught for years — by rote. Webster's New Collegiate Dictionary defines rote as "the use of memory ... usually without intelligence."

Those who choose to believe mathematics is nothing more than a series of memorized tables and formulas for calculating a correct answer will forever remain bogged down with the basics of arithmetic. They'll never understand the fun of mathematics as a language, as a universal system for solving problems.

Those who see Logo as a programming language through which children will become instantly computer literate (whatever that means) will remain forever bogged down in an outdated, myopic concept of computer science. No computer language will turn a group of math students, for example, into instant Einsteins.

There are numerous Logo curriculum guides, Logo tutorials, not to mention Logo classrooms which remind me of my days as an Air Force Tactical Instructor. On commands of FORWARD, BACK (To the Rear), Column LEFT, and RIGHT, I could get a squadron of 80 officers or airmen to act as one disciplined, totally subserviant unit. On the drill field, peer pressure, the fear of embarrassment, quickly turned 80 men into a single unit that moved and thought as one. Such discipline was essential in the military environment. Lives depend upon it. But what about the classroom? Unfortunately too many teachers try to turn their Logo classes into the same disciplined, single-minded (or mindless) entities. But why?

Somewhere, somehow, people need to understand that Logo and the personal computer are not entities unto themselves. Learning to use Logo and the computer is of little value unless the user can transfer that new literacy to other individualized learning environments. That won't happen automatically as some would have you believe. These are skills that must be developed.

Could it be that we're putting the cart before the horse? Are the means becoming more important than the end?

One small triumph at the time Ray developed confidence in his ability to analyze a problem, to solve it — to take risks, to explore, to discover new methods.

Is being computer literate becoming more important than the effective use of that literacy? Of what use is literacy if there is no sense of self for which to apply the literacy? If this doesn't seem to make much sense, take another look at the Logo tutorials and curriculum guides flooding the market. Readers end up doing nothing more than memorize the commands of a computer language which offers them nothing. Unto itself, it has no practical value. Of what value is this language unless students are taught to synthesize the learning into the multiple enviornments of their everyday lives?

What would happen if children were taught the fundamentals of problem solving, with or without a computer, with or without Logo? What would happen if they learned that mathematics, calculators, Logo, Forth, PILOT, Pascal, Lisp,

(continued)

and personal computers are merely some of the tools which can aid the problem-solving process? What would happen if, in addition to becoming computer literate, they become self-literate?

With the current concentration on rote learning, few children learn problem solving skills in the classroom. It is so much easier and safer for them to be a passive learner of facts. To participate in the learning process requries far too active a role, too much risk.

Rather than be stimulated by problems which don't have an immediate answer, children are often stymied by them. Very often it is peer pressure which makes students fear the embarrassment of failure. Unless forced into it, few take risks. They abandon complex problems, those for which there is no immediate and safe answer. They become content to absorb information, dissect, reassemble and regurgitate it on command. That's all the "system" ever asks of them.

But do they ever learn to use their learning, to apply it?

One of my more fascinating experiences with Logo was working with a very tenacious young man who would sit at the computer for hours if allowed. One of Ray's first projects was drawing the skyline of Dallas from memory. His first attempt was a straight linear procedure. But after losing his work a couple of times, he found that it worked better when he cut the procedure into pieces. The result, after a number of two hour sessions, was a Dallas skyline which changed from day to night and back again.

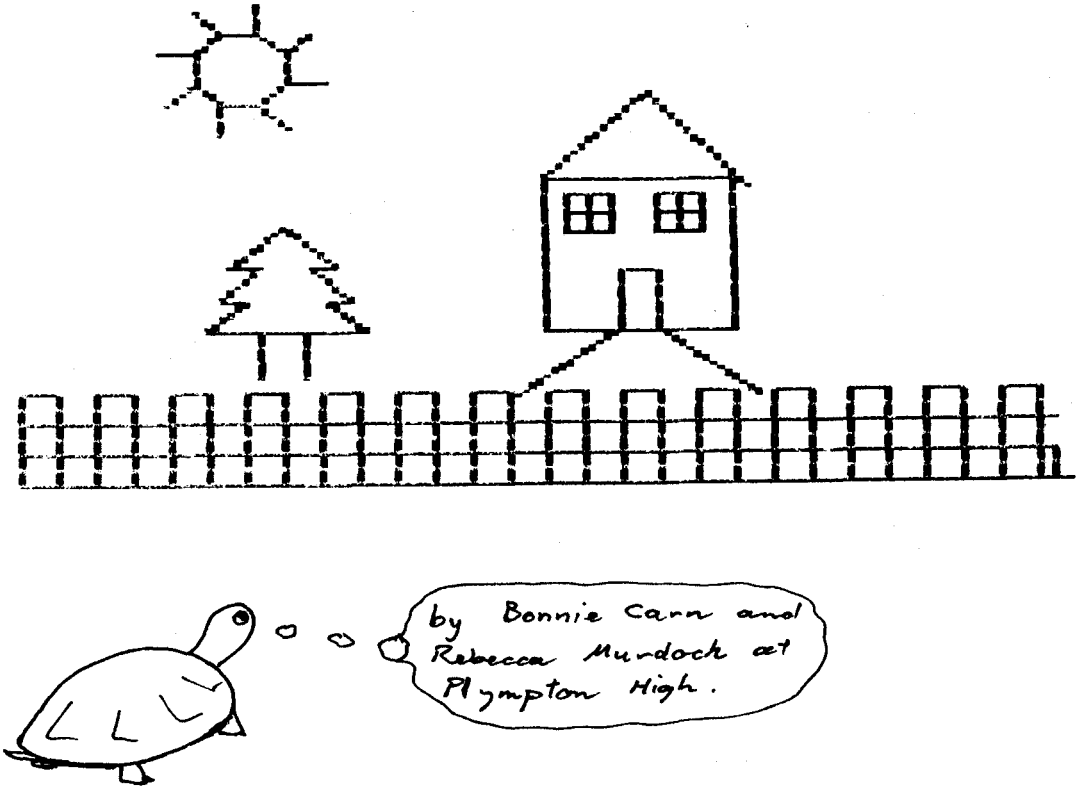
Ray was far from the brightest boy in our group. He tended to be a bit shy and introverted. Only when his interest was sparked did he seem to come out of himself, a turtle coming out of his shell. When school started I was very surprised to learn that Ray had a permanent pass to the counselor's office. It seems that he couldn't sit still in class for more than twenty minutes at a time. That was certainly not the case at the computer.

What Ray did at the computer over that summer of 1981 was to apply the same scientific method to his problems as I did to my engineering classes thirty years ago and my father did to his graduate chemistry studies thirty years prior to that.

1. Identify the problem
2. Gather relevant data.
3. Formulate a hypothesis.
4. Experiment.
5. Interpret the results of the experiments.
6. Draw conclusions.

While the other boys did much the same thing working in small groups, Ray tended to work alone. At the computer there was no fear of failure, no teasing from his peers, no embarrassment. One small triumph at the time Ray developed confidence in his ability to analyze a problem, to solve it — to take risks, to explore, to discover new methods. As Ray

(continued)



developed into one of the more creative programmers, others obviously lagging behind would tease him far more than usual. Where once Ray would have jumped off his stool swinging, he now enjoyed merely jeering back at them. Over the next school year, at meetings every other Saturday, Ray matured significantly — one small triumph at a time. It was a most enjoyable, though sometimes irritating experience to watch the pendulum swing the other way, to where Ray became the swaggering braggard eager to display his new-found macho image.

A more dramatic example of this problem-solving approach came in a computer club the YPLA ran for third and fourth grade students last year. There were about a dozen students in the club but only two computers. Thus we divided them into small groups or "learning centers" with each group working on a different facet of the same problem.

Why is it some kindergarten children can do so much more in the Logo editor than some other local fifth and sixth graders can do in the immediate mode?

One of the club members had a difficult time relating to the computer. It wasn't until we discovered his love for soccer that we were able to spark his interest in the computer. After dividing the club into groups, we challenged them to see who would be the first to duplicate the soccer ball pattern in a Logo procedure (draw a pentagon surrounded by hexagons).

It was interesting to note some of the different approaches to the problem. Some were random experiments, patterns without thought or reason. Some tried to draw the pattern on paper first. Some tried to fit paper or plastic shapes together.

A group of girls were the first to discover the solution to the problem. They found they could only duplicate the soccer ball pattern in three dimensions. It couldn't be done on the two-dimensional screen without allowances for the spherical shape of the ball. After dumping twelve patterns to a printer, they cut them out and made their own small soccer ball.

How as it that these third grade girls were able to think beyond the abstract two dimensional computer screen and perceive the their dimension? Certainly they had not been trained to do this. How was it that Ray, a hyperactive, learning disabled boy with a history of a very short attention span and poor grades, could perceive the skyline of Dallas and then spend hours methodically recreating it on the screen?

How was it that after roughly 60 hours of exploring Logo, four elementary school children (two sets of brother and sister, 6 and 8, and 8 and 10) could more accurately visualize random angles than trained engineers who spent eight or more hours per day at a CAD (Computer Aided Design) terminal, or on a drawing board.

Why is it some kindergarten children can do so much more in the Logo editor than some other local fifth and sixth graders can do in the immediate mode? How is it that elementary students from the Utah State School for the Deaf have developed complex animated stories using the same techniques used in Disney and other cartoons — a series of standard backgrounds over which another series of changing cells are projected to tell a story. And all of this is done in Apple Logo™ using Turtle Graphics.

Seymour Papert sees the computer as, "an entity that precipitates and changes the way you learn."

An 11-year-old Logophile from the San Diego area designed his own futuristic airplane. He sent us orthographic projections of his plane, drawn in turtle graphics. I spent more years in school than that young man has been alive, learning to do the same thing.

A 13-year-old from California developed a simple word processor using Logo. She's blind, and has found it easier to write letters and assignments using her own procedure.

Each of the programming challenges sponsored by the YPLA has produced some outstanding programs by children from all over the world. More than the procedures themselves, is the different levels of competency obvious among the same age levels at different schools. In that we're dealing with hundreds of students from all over the world, there has to be some relevant information in an examination of their proficiencies.

Seymour Papert sees the computer as, "an entity that precipitates and changes the way you learn." He sees it as an entity, "that changes how you think about yourself and other people ... and changes the way you learn because of its presence in a complex social and epistomological surrounding." His world is one of microworlds (Ray's world). His microworlds give the child a safe place to enter, a secure world where they never get into trouble, there are no right or wrong answers, no failures.

This is so different from so many of today's classrooms. These classrooms are places where the System is supreme. The System doles out carefully-shaped bits of information, that which it directs the masses should know at a specific age. That this information is incomplete and sometimes inaccurate is irrelevant. When the masses are eligible, another piece of information will be added. That the new information may conflict with the earlier bits, merely adding to confusion rather than to understanding, is irrelevant.

This is the place where a young mind can start with what it knows, explore, discover, and build — one small triumph at a time.

This Orwellian world does exist. This description may be a bit cynical. But having listened at length to the young participants of this system, it serves us well to balance this extreme with Papert's microworlds.

A child's microworld is a place rich in discovery; without embarrassment, without admonishment. It is a place where a child never need question the validity of newly-discovered knowledge (wondering when adults are going to change the rules again). The microworld is a secure place where the child directs the analysis of new knowledge. It is here that children explore it, discover its meaning, confirm it, and then use it to assimilate new discoveries. This is our world of the Great Logo Adventure.

In this world of adventure, there is no fear or misconceptions of the technology; only the childlike acceptance of the ever-expanding opportunities offered by the imagination. This is the place where a young mind can start with what it knows, explore, discover, and build — one small triumph at a time. This is the microworld where students can experience one small piece of infinity, analyze it, define it, and then build upon that to develop a whole new sense of confidence, a new perception of themselves, and of others.

Does it really matter that thee microworlds are so much more enjoyable, so much richer, so much more powerful than the regimen so many adults had to endure when they were young?

Is the microworld a little girl develops at the computer any different than the fantasy world she develops when she dresses up in her mother's clothes to play house? Not really. The fantasy world of "dress-up" starts with the child's perception of reality. She then explores other chunks of reality in her attempt to emulate her perception of her mother's world. The more she seeks to emulate, the more she comes to know the limits of her emulation. But the microworlds a child creates at the computer are limited only by the imagination.

"He who has imagination without learning has wings but no feet." When used to advantage, Logo and the computer can provide the security on which children can plant their feet so their minds can soar with the eagles. Where the fantasy worlds of children have distinct limits, Papert's microworlds are very real, limited only by the imagination. And when the imagination is not fettered by adult conditioning and inhibitions, it is an extremely dynamic and powerful force. Given the opportunity to soar, the eagles pale in comparison.

Not too long ago, a parent asked if this form of learning wasn't adding to the generation gap? How could parents and teachers cope with what appeared to be a lack of discipline, this lack of control?

Is discipline and control what is really required? Why can't learning be fun?

The most common complaint I hear from young people is that their teachers don't understand them. (Often their parent's don't either.) Adults are way behind the times. They don't care. They don't communicate. Of the 100 or more teachers and professors I studied under, a few communicated their ideas effectively. Of these, only a few truly facilitated learning..

At the 1984 Spring CUE Conference in Pasadena, Mary Cron of Microcomputer Resources Inc. offered a presentation on problem solving. In this, she defined the role of the Facilitator:

- Keeps students goal-oriented
- Reinforces problem-solving as a skill
- Gives encouragement and reinforcement
- Good listener
- Asks open-ended questions
- Assures that the necessary content has been covered
- Supplies information
- Encourages collaboration and sharing
- Manages logistics
- Maintains evaluative overview
- Encourages self-evaluation

D.H. Lawrence offers a lesson on how to become a facilitator in this quote from his essay, Fantasia of the Unconscious and Psychoanalysis and the Unconscious:

"And this is the way to educate children: the instinctive way of mothers. There should be no effort to teach children to think, to have ideas. Only to lift them and urge them into dynamic activity. The voice of dynamic sound, not the word of understanding. Damn understanding. Gestures and touch, and expression of face, not theory. Never have ideas about children — and never have ideas FOR them."

In prehistoric times, curious children were encouraged to learn by emulating their parents using such tools as a straw doll or sharpened stick. In that our learning environment has expanded a bit since then, children now seek to explore their environment on a higher, more intellectual plane. Logo and the computer are among the facilitators. As the straw doll and sharpened stick helped the cave children prepare for adulthood, Logo and the computer can make the preparation of thinking skills a bit easier.

But how does Logo accomplish this? Logo and the computer do nothing by themselves. Someone needs to direct — to facilitate their effective use.

Should Logo be taught? Should it be learned? It might be more accurate to consider that Logo is an adventure to be experienced. In such adventuresome learning environment, who is the teacher and who is the student? One becomes the director while the other is the facilitator.

As a programming language, a good descriptor for Logo is to paraphrase Sir Winston Churchill: Logo is the worst of all languages, except for all others. The computers on which Logo is used are the worst of all computers, except for all others.

More relevant is the Logo experience. Through this friendly, conversational, interactive language, the healthy, the physically and mentally-disabled are experiencing a dynamic new world of confidence and opportunity. As their imaginations discover the limits of today's technology, its exciting to realize these young people — regardless of age — will move on to other computers, other languages, other powerful ideas ... riding on the back of a turtle.

NOTE: Joseph Power of Digital Research, Inc., Pacific Grove, California, and Mary Cron of Microcomputer Resources Inc., Los Angeles, California, made invaluable contributions to this article. I wholeheartedly thank them for sharing their ideas.

The Special Laws of Computer Science

1. The need to change the program will be communicated to the programmer no more than five seconds after the coding is complete.

2. In any program, the parts which are obviously correct beyond all need of checking, will contain the error, and any routines which have been copied from an unimpeachable source will be found to cause the execution errors.

3. In any miscalculation, the source will never be found if more than one person is involved.

4. All constants are variable.

5. Never simplify the design of a program if a way can be found to make it more complicated, and therefore, more wonderful.

6. Student programmers should always keep their disks filled with miscellaneous and useless files. This shows how busy they have been.

7. Before studying a problem, make sure you first understand it thoroughly.

8. Interchangable parts won't.

9. The probability of failure of a peripheral on an interface is inversely proportional to the ease of repair or replacement.

10. A falling tool will always land where it can do the most damage.

11. Only the most delicate components will fall onto concrete floors. All others will land on the carpet.

12. Plotters will always put more ink on the operators than the paper.

13. After all 38 screws have been removed, it will be found that the wrong cover has been removed.

14. The availability of any replacement part will be inversely proportional to the need for that part. Similarly, if a new system requires n parts, there will be $n-1$ available locally.

15. Any wire cut to length will be too short. At least one technician will attempt to solve the problem by cutting off more wire.

16. A fail-safe circuit will destroy all others. It thus stands to reason that if a circuit cannot fail, it will.

Computing at Entropy House

Heard at a Steve Ocko event were some senior high school Computing Studies types wondering why they shouldn't use the Apple games port instead of the LEGO interface card. No reason at all why they shouldn't. The point about LEGO/Logo of course is that it's designed for primary school teachers who don't want (and don't need) to know about interfacing and all that hardware stuff. Just plug it in and go.

LEGO hardware has its limitations; I'm often frustrated at the limited range of angles and the rather cumbersome structures often necessary, but then I'm accustomed to building kayaks and, in the past, model aircraft. Recognise the limitations and it's a quick and easy means of putting a machine together, just as Logo is a quick and easy means of putting a program together.

That TC Logo graphics screen a few pages back was produced by the old trick, ie. crashing out of Logo to DOS 3.3 with ‹CTRL› ‹Reset› and doing a BSAVE name,A$2000,L$2000. It was then printed with the ImageWriter Tool Kit. You can do the same with any other Logo pic. Why not boot the Tool Kit directly? Because it scrambles the graphics screen. (Bleh)

As for the games port, Peter Harris, technical whiz at Random Access, is developing a cheap interface kit for use in computing and science laboratories. The idea is to have something sufficiently cheap that, if you blow it up, you can throw it away and make another. Ring 223 2505 and ask for sequential access to Peter or Terry Cox.

Late last year an imposing looking document entitled *The Impact of Changing Technologies on Primary Schools* was released. The writer clearly has no concept of what Logo is really about. A couple of quotes:

'5.10 ...its [Logo's] tendency towards experiential impoverishment (two dimensional, mechanical, unemotional and sensually impoverished procedures);'

'5.12 We have the spectre of children manipulating the turtle to command a perfect square and yet not being able to draw it by themselves. When children move more fully into the concrete-operational level at about eight years, they are more fully able to take advantage of the computer challenge.'

The document quotes Joseph Weizenbaum, arch critic of computing, to support its slender arguments, superbly demolished by Anne A'Herran in a recent CEGSA Newsletter. But Weizenbaum also wrote this:

'The other side of the coin is the belief that one cannot program anything unless one thoroughly understands it. This misses the truth that programming is, again like any form of writing, more often than not experimental. One programs, just as one writes, not because one understands, but in order to come to understand. Programming is an act of design. To write a program is to legislate the laws for a world one first has to create in imagination. Only very rarely does any designer, be he an architect, a novelist, or whatever, have so coherent a picture of the world emergent in his imagination that he can compose its laws without criticism from that world itself. That is precisely what the computer may provide.' (Computer Power and Human Reason, p 108)

The writer might also be recommended to read Cynthia Solomon's and Sylvia Weir's books (reviewed in a previous *POALL*) for something of what Logo is really like in the hands of good teachers.