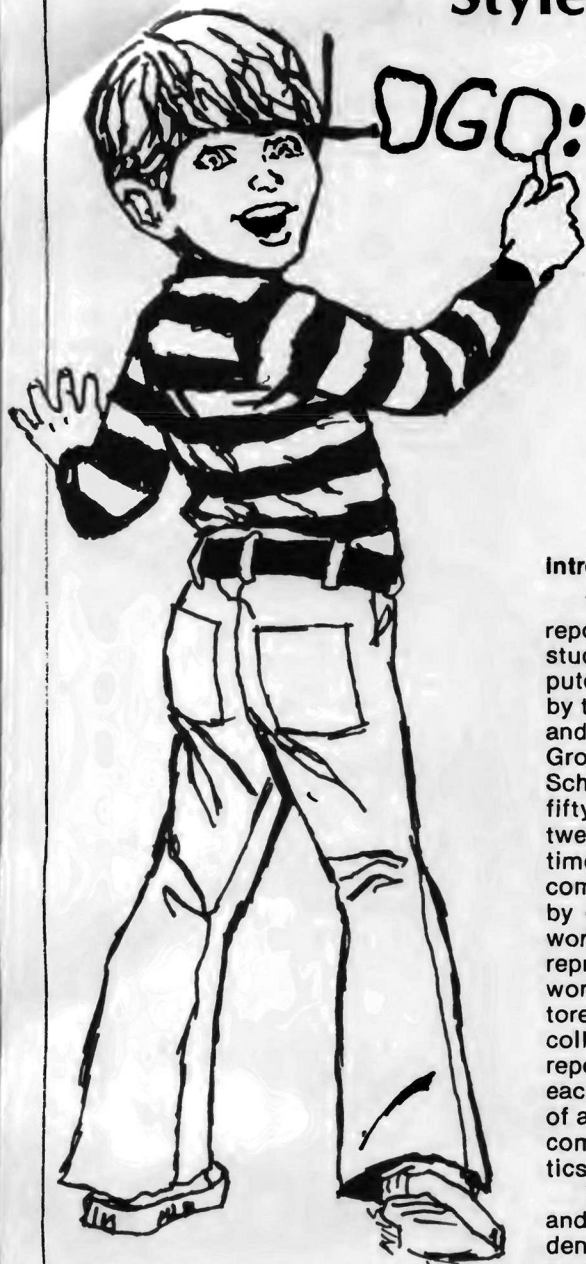


A Comparison of the Problem Solving Styles of Two Students Learning



LOGO: a Computer Language for Children

Daniel H. Watt

Introduction

This paper is one of a series of reports resulting from an 18-month study of an elementary school computer laboratory.¹ The study, funded by the National Science Foundation, and carried out by the MIT LOGO Group in collaboration with the Public Schools of Brookline, MA, provided fifty sixth-grade students with between 24 and 36 hours of computer time and instruction in LOGO, a computer language designed for use by elementary school children. The work of sixteen of these students, representing a range of interests, work styles and abilities was monitored in great detail. From the data collected we have prepared a series of reports outlining in detail the work of each child, and analyzing the learning of all sixteen children in the areas of computer programming, mathematics and problem solving.^{1,2}

The contrasting work of Donald and Deborah, two of the sixteen students is presented in this paper. The contrasts between the students give rise to a number of pedagogical questions and issues which will be considered in the concluding section.

The computer language, LOGO, was developed approximately eleven years ago as a vehicle by which children could learn computer programming, problem solving and mathematics in a learning environment that grew out of the cognitive psychology of Jean Piaget and an approach to the design of computer languages, based on ideas arising from work in Artificial Intelligence.

4,5,6 Children are typically introduced to LOGO by using the computer to control a "TURTLE," an imaginary creature which "lives" on a graphics display screen, the movement of which is controlled by commands typed at a keyboard; for example: FORWARD 100, BACK 50, RIGHT 90, LEFT 45, etc.

FORWARD 100 moves the TURTLE forward "100 TURTLE steps" and leaves a trace on the display screen, while LEFT 45 causes the TURTLE to rotate 45 degrees to the left. Drawing pictures with the TURTLE is an initial programming activity with an immense resonance for most children. Children can identify with the TURTLE, imagining themselves going through its motions, as it carries out a particular task. At the same time, controlling the TURTLE becomes a metaphor for controlling the computer itself. Like the TURTLE, the computer responds to an ordered series of commands, and to procedures, defined as series of commands.

The way in which the action of the TURTLE can lead to a geometric design, as well as the method used by children to define procedures is illustrated in the following simple examples. Having drawn a square by repeating the steps, FORWARD 100, RIGHT 90 four times, a student can define a procedure by choosing a procedure title (BOX, for example) and typing a series of commands in order.

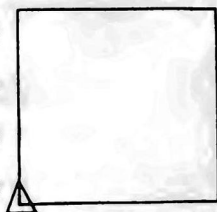
When the student types the new command, BOX, the shape shown in figure 1 will be drawn on the display

The work reported in this paper was supported in part by the National Science Foundation (under grant number 77-19083SED) and conducted at the Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts. The views and conclusions contained in this paper are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the National Science Foundation or the United States Government.

Daniel H. Watt, Visiting Research Associate, Division for Study and Research in Education, LOGO Group, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139.

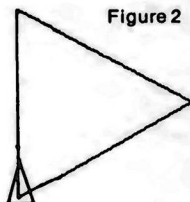
LOGO, con't...

Figure 1
TO BOX
 10 FORWARD 100
 20 RIGHT 90
 30 FORWARD 100
 40 RIGHT 90
 50 FORWARD 100
 60 RIGHT 90
 70 FORWARD 100
 80 RIGHT 90
 END



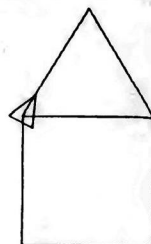
screen. (The small triangle shown in figure 1 represents the TURTLE). A similar procedure, TRI, could be defined as follows:

TO TRI
 10 FORWARD 100
 20 RIGHT 120
 30 FORWARD 100
 40 RIGHT 120
 50 FORWARD 100
 60 RIGHT 120
 END

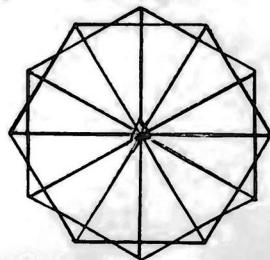


Having defined procedures BOX and TRI, the student can now use these as commands to the computer, in effect developing his own "private language" as he goes along. In particular, they can easily be used as sub-procedures in creating the drawing, HOUSE, or the abstract design, FLOWER.

Figure 3
TO HOUSE
 10 BOX
 20 FORWARD 100
 30 RIGHT 30
 40 TRI
 END



TO FLOWER
 10 REPEAT [TRI RIGHT 30] 12
 END



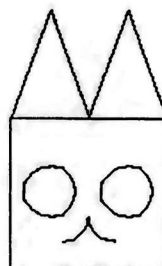
These TURTLE commands, when embedded in a suitable interactive, procedural computer language such as LOGO, provide the basis for a rich universe of activities called Turtle Geometry, which includes cartoon drawings, simple and complex geometric designs, mathematical theory building, and computer games and animation. Extensions of Turtle Geometry have proven fruitful when used with advanced high school students

or MIT freshmen and sophomores. The universe of turtle geometry provides a conceptual framework for such aspects of mathematics as coordinate systems, positive and negative numbers, the use of variables, symmetry and similarity, the significance of special angles (30-60-90, 180, 360, etc.). The computer programming involved in beginning LOGO activities can include the notion of procedures and sub-procedures, recursion and iteration, the naming of procedures and variables, the use of conditional logic in the form of simple "stop rules" for repetitive procedures, the understanding of procedural hierarchy, and the use of "debugging" strategies.

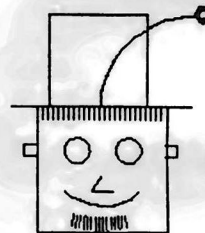
Within the "universe" of turtle geometry there is room for different students, working individually, to create their own sub-universes or "microworlds" with their own limited but expandable set of concepts, and related activities and projects. The pedagogy of a LOGO classroom can be seen as the task of helping each child to create, explore and extend his or her own "microworlds." In this paper we describe the "microworlds" of two students, Deborah and Donald, who approached a very similar project, in very dissimilar ways. We feel that by focussing on these two individuals we can demonstrate how the computational environment of

TO RABBIT
 5 HAT
 10 LITTLE EYES
 15 FACE
 20 PENUP
 25 FORWARD 70
 30 FORWARD 3
 35 RIGHT 20
 40 PENDOWN
 45 EARS
 50 RIGHT 90
 55 FORWARD 50
 60 FORWARD 3
 65 FORWARD 60
 70 FORWARD 5
 75 FORWARD 3
 80 RIGHT 90
 85 RIGHT 20
 90 EARS
 END

Figure 5



TO HEAD
 1 BOX
 2 EYES
 3 NOSE
 4 MOUTH
 5 BEARD
 6 HAIR
 70 EARS
 80 HAT
 85 FLOWER
 END



LOGO fosters the learning of children with very different developmental levels, learning styles and academic abilities.

A Comparison of Donald and Deborah's Work

Deborah and Donald each carried out one major project as part of their LOGO experience. They each programmed the computer to draw a cartoon-like "head." Both projects, and the superprocedures used to draw them are shown in figure 5.

Deborah is considered by her teachers to be a "slow learner." In her regular classes she often appears to be withdrawn, unrelated to the subject matter or to her fellow students. When she began work in LOGO, she was totally dependent on the teacher — requiring his reassurance on matters as routine as when to type a carriage return.

Deborah was able to slowly build her confidence and understanding by limiting her choices of LOGO commands and inputs, limiting the goals of her work, and by working in a way that minimized the chances of error. While Deborah began her LOGO experience by asking for help at literally every turn, she had a deeply ingrained resistance to new ideas or concepts. For a long time she rejected the use of subprocedures, which could have greatly expanded her possibilities. It was as though she deliberately provided herself with a very definite and restricted "microworld" in which to operate.

Deborah used as few different commands as possible in her work. Basic turtle commands along with arc primitives RARC and LARC were almost the only commands she used. For inputs to TURTLE commands, she used only multiples of 10, up to 100. If a larger effect was needed, she would use two steps, as in FORWARD 90, FORWARD 30.

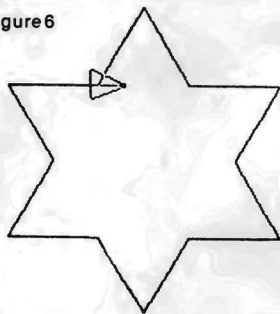
Deborah began by using only inputs of 30 for all TURTLE commands, and gradually expanded to include other numbers, while continuing to use 30, 60 and 90 as her favorites. This "microworld" which Deborah chose for herself, is very nearly as rich as all of Turtle Geometry. It includes squares, triangles, "circles" "stars" "men" "rabbits," and a variety of abstract designs, as well as the mathematical concepts of perpendicularity, inverse operations, the total turtle trip theorem, symmetry, similarity, estimation of lengths and angles, planning and debugging, and procedure writing.

LOGO, con't...

By limiting her inputs to numbers such as 30, 60 and 90, Deborah actually enhanced the possibility that random explorations would produce interesting results. At the same time, she seemed to have a high degree of visual intuition, often choosing precisely the correct input to produce a desired effect. Her patience in a one-step-at-a-time mode of operation was remarkable. Her format was quite stereotyped. (1) carry out one turtle step (turn, move or penup); (2) check to see if that looks right on the screen; (3) if so, write down the step and continue; (4) if not, clear the screen, retype all the steps previously written down and try another choice for the questionable one; (5) when a design looks good, choose a name for it, and copy all the steps to make a procedure.

As an example of Deborah's success with this approach, we examine the way in which she drew a six-pointed star without making a single mistake. She began by turning the turtle RIGHT 30, and used a combination of FORWARD 70s and RIGHT 60s to complete the star. The actual rotations required to construct the star were RIGHT 120 at each point, and LEFT 60 at each inner vertex. The way Deborah accomplished these rotations was quite typical of her work. After each forward step, Deborah would turn the TURTLE RIGHT 60. She kept turning it RIGHT 60, until the TURTLE was headed in what seemed to her to be the right direction. This required two repeats of RIGHT 60 at each point, and five repeats of RIGHT 60 at each inner vertex. At one inner vertex she missed the correct orientation, and calmly repeated RIGHT 60 a total of eleven times until the TURTLE was aimed in the right direction. When she copied the steps in her notebook, she copied all eleven RIGHT 60s without any hesitation.

Figure 6



It was not quite late in the series of classes that Deborah was ready to undertake a major project. She drew a picture of a rabbit in her notebook, and asked the teacher if he thought that would make a good project. He suggested modifying the rabbit, to make use of straight rather than curved lines, and helped her redraw the picture in a more simplified form.

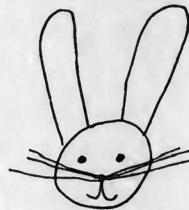
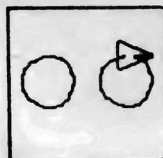


Figure 7



Although Deborah began by trying to draw the rabbit as a long series of commands, she soon accepted the suggestion that she break the problem into parts, and make each part a separate subprocedure. Although her work was directed toward an overall goal, and involved a certain amount of "top-down" planning, she constructed each part of the rabbit piece by piece, in an exploratory fashion. Once again her choice of inputs to FORWARD and RIGHT commands were such that it was relatively easy for her to make the design come out the way she wanted. Without any apparent planning, she chose the length for the sides of the rabbit's head (FORWARD 90 FORWARD 30) in a way that made it easy for her to locate the eyes and nose symmetrically. To locate the eyes she positioned the TURTLE at one side of the head, and moved it FORWARD 30. Deciding that that "looked like the right spot for an eye," she drew a circle of radius 20. She then moved the TURTLE twice more in steps of FORWARD 30, decided that that looked about right for the second eye. Thus she was able to locate both eyes symmetrically. It was apparent that she had not **decided** that the rabbit's head was 120 steps across, and that she could therefore locate the eyes symmetrically 30 steps in from each edge. More simply, she "knew" that 90 and 30 were "useful numbers" and that if she kept trying FORWARD 30 "it ought to come out right."

Figure 8



With the completion of her rabbit project, Deborah had almost totally reversed her initial feelings of dependence and incompetence. She invited her parents, teachers and school principal to visit the computer lab, and in many ways, demonstrated to her visitors and classmates her new found sense of confidence, satisfaction and power.

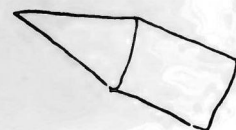
Donald's work, which provides a striking contrast to Deborah's, was characterized by a strong component of advance planning, and the creating of structures within which problems could be solved. At the same time, Donald was quite ineffective at the visually-based, exploratory modes of problem solving which were so useful to Deborah. He had difficulty estimating angles, and making use of the visual feedback provided by his explorations to improve his next attempt.

Throughout his work Donald was extremely receptive to suggestions from the teacher, often making use of new ideas before he fully understood them. In this way, he was able to incorporate into his way of working, strategies that would continue to prove useful, as he gradually came to understand them through use in more than one context. He seemed to have the confidence that he could make use of the teacher's suggestions effectively and that he would eventually understand them, even if the concepts were a bit hazy at first.

As an example of both the effectiveness and the difficulties associated with Donald's structured planning approach as well as the frustration he experienced with visual approaches to problem solving, we consider his construction of a "house" from a square and a triangle, a common LOGO task, tackled by many students at an early stage of their LOGO experience.

At first Donald attempted an exploratory approach to solving this problem. He began by having the TURTLE draw a triangle on the screen, making use of the TRI procedure described above. Having started by drawing a triangle, Donald established a "framework" for solving the problem corresponding to the initial orientation of the triangle. Asked to draw a picture of what he was trying to accomplish, he made the diagram shown in figure 9. Since

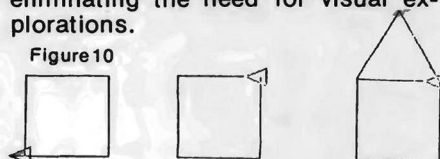
Figure 9



he was now dealing with two disorientations, the gap between the TRI and BOX procedures, and the tilted orientation of the entire shape, Donald had more difficulty than he could handle, and in an hour of exploration, never succeeded in resolving the problem in this form.

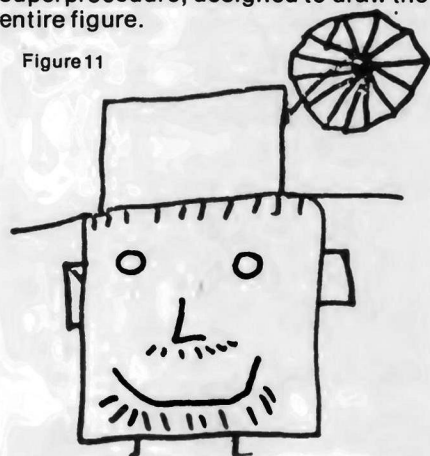
At the next class, the teacher suggested that Donald draw the BOX first. This suggestion provided him with enough new insight to devise a plan for solving the problem. Donald's plan allowed him to avoid the usual problem of finding the rotation needed to attach the triangle to the upper left hand corner of the box. Instead, he moved the turtle to the upper right hand corner of the box, reversed its direction, and then drew the triangle so that its first side was along the top of the box. In this way he resolved the problem analytically, eliminating the need for visual explorations.

Figure 10



Donald's HEAD project occupied him for more than 12 class periods — more time than any other student devoted to a single project. This project also began with a plan, a cartoon-like drawing of a man's head, which formed the basis of Donald's work. After one session of exploratory work, Donald drew a revised plan, and worked with his teacher to create a superprocedure, designed to draw the entire figure.

Figure 11



Donald's original superprocedure, TO HEAD, included the first six features of the head: the outside (BOX), EYES, NOSE, MOUTH, BEARD and HAIR. Once the superprocedure was written, each subprocedure became a mini-project, requiring one or two classes to resolve. Each feature of the head required its

own "construction plan," a combination of the analysis and exploration needed to carry it out. While working on the features of his head, Donald made use of a great deal of teacher assistance, especially in developing approaches to geometric analysis that were necessary to overcome his difficulties with visual problem solving.

When Donald came to his subprocedure, EYES, for example, he could not resolve the problem of how to locate the eyes symmetrically, by means of visual exploration. After several unsuccessful attempts, he asked for advice from the teacher. The teacher helped him set up a kind of "grid" with which he calculated the location of the eyes, and he was then able to solve this problem without any further difficulty.

Other, more complex problems that Donald encountered in the course of the project were resolved in similar fashion — by creating "structures" which enabled him to solve the problem analytically and avoid visual explorations. In the course of his work Donald encountered estimation of distances and angles, the geometry of arcs and circles, the total turtle trip theorem, and the use of both grid-based and intrinsic coordinate systems. He learned to use sub-procedures and sub-sub-procedures, to use patterned procedures making use of a REPEAT command, to make use of variables to control the size and shape of his "hat" and "flower" and to use a procedure with a conditional stop rule. Although Donald only "learned" these approaches to the extent necessary to solve the particular problems inherent in his project, each succeeding use of the same concept, reinforced his exposure to it, deepening his sense of mastery.

Donald's final figure, drawn by the superprocedure, HEAD, represents an almost literal mapping of his revised plan, figure 11, into a computer program.

Conclusion

In summary, we observe that Deborah's and Donald's problem solving styles were markedly different in several ways:

1. Planning

Donald was a "planner." He made an overall plan for his project, and created separate plans for each subpart of his project. Deborah had an overall idea of what she was trying to accomplish, but her actual project was carried out in an exploratory manner, one step at a time. Donald's superpro-

cedure HEAD, was written to serve as a framework for his project; Deborah's superprocedure, RABBIT, was written as the final step in completing her project.

2. Analysis vs. Intuition

Deborah relied on an accurate visual intuition and a very limited set of inputs to TURTLE commands; Donald's visual intuition was poor. He relied on mathematical analysis to resolve each problem he faced.

3. Comfort With New Ideas

Donald absorbed a great deal of new information and tried a number of new approaches as he carried out his project — confident that he could learn the skills he needed as he went along; Deborah resisted new learning — her learnings consisted largely of consolidating and reinforcing old skills and methods.

4. Obtaining Assistance From A Teacher

Donald made regular use of assistance from his teacher, while Deborah insisted on "doing things in her own way," and resisted suggestions which she did not fully understand.

The observed differences give rise to pedagogical questions that are not limited to teaching children to use computers: How can teachers learn the learning styles of their students? How can teaching approaches in normal classes be flexible enough to support such a wide variation in learning styles? How can a teacher help Deborah acquire some of Donald's analytic approach? Can Deborah's intuitive understanding be learned by Donald?

The success of both Donald and Deborah in a LOGO environment leads us to believe that additional experience in such an environment may be productive in beginning to answer these questions. □

Footnotes

1. "An Assessment and Documentation of a Children's Computer Laboratory," LOGO Memo 48, 1977.
2. "An Interim Report of the Brookline LOGO Project," LOGO Memo 49, 1978.
3. "Final Report of the Brookline LOGO Project," to be published Spring, 1979.
4. "Programming Languages as a Conceptual Framework for Teaching Mathematics." Feurzeig W., Papert S.A., Bloom M., Grant R., and Solomon C. Report No. 1889. Bolt, Beranek, and Newman. Cambridge, Massachusetts. 1969.
5. "Twenty Things to do With a Computer." Seymour Papert and Cynthia Solomon. LOGO Memo 3, June 1971.
6. "Teaching Children to be Mathematicians vs. Teaching About Mathematics," Seymour Papert. LOGO Memo 4, July 1971.
7. *Turtle Geometry*, Abelson and diSessa, to be published by MIT Press in 1979.

Sine POLYs: Some Geometrical Explorations

Neil Rowe

Analytic geometry on the computer doesn't have to be just drawing graphs of known functions. Here are some simple but powerful things to explore.

I use abstract things called "turtles" to draw the patterns. That is, things that can either go FORWARD a specified distance, drawing a line, or turn RIGHT a specified angle in degrees. The classic "turtle" program is called POLY, and consists of repeating over and over again the pair of instructions.

```
FORWARD S
RIGHT A
```

where S and A are some numbers. For instance, if A = 90 degrees, it's a square; if A = 60 degrees it's a hexagon. S just controls the size of the drawn figure. This program will draw all regular polygons and regular "stars". Note when S and A are both small, the angles of the figure are hard to see, and you'll appear to be drawing a smooth circle.



Figure 1: AMPOLY, K = 1 and A = 1.

In what follows we'll consider programs of the POLY form, but where sines and cosines are involved.

1. AMPOLY

Consider repeating forever:

```
FORWARD SIN(T)
RIGHT A
ADD A TO T
```

or equivalently:

```
FORWARD SIN(HEADING)
RIGHT A
```

where HEADING is a variable representing the total amount the turtle has turned right since it started. (Note that a heading of 360 degrees may

Neil C. Rowe, Dept. of Computer Science, Stanford University, Stanford, CA 94305.

look like a heading of 0, but we consider them here two different things.) The above program draws a cycloid (Fig. 1).

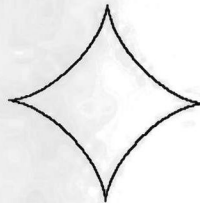


Figure 2: AMPOLY, K = 2 and A = 1.

Similarly,

```
FORWARD SIN(K * HEADING)
RIGHT A
```

and related programs draw cycloid-like closed figures (Figs. 2-4). (The * stands for multiplication.) I call this program AMPOLY because it's a POLY whose FORWARD step size is being "modulated" at some frequency.

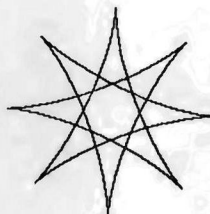


Figure 3: AMPOLY, K = 4 and A = 1.

2. FMPOLY

You can also have a "frequency modulation" POLY:

```
FORWARD S
RIGHT (CF + (MI * SIN T))
ADD MF TO T
```

where CF denotes center frequency, MF modulation frequency, and MI modulation index ("How much" modulation you want). For small S in these programs, the amount of curvature of a circle is varied periodically, so when MI is small they look close to circles. See Figs. 5-9 for examples.

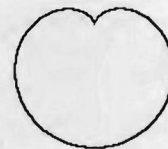


Figure 4: AMPOLY, K = 1 and A = 3.

3. HSQUARE

There are lots of interesting variants on the preceding. One of my favorites is the repeated loop.

```
FORWARD SIN(HEADING + HEADING)
RIGHT A
```

These designs (see Figs. 10-12) show many interesting symmetries, and are highly varied for different values of A.

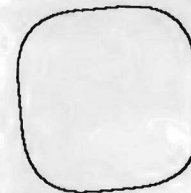


Figure 5: FMPOLY, CF = 2, MF = 8, MI = 2.

4. More Familiar Curves

Turtle programs of this form can also draw some of the familiar curves of Cartesian geometry. For instance, a parabola (Fig. 13):

```
FORWARD S
RIGHT (SIN(HEADING) * SIN(HEADING) * SIN(HEADING))
```

or a hyperbola:

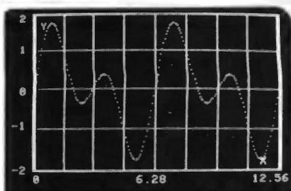
```
FORWARD S
RIGHT (SIN(HEADING) * SQUAREROOT(SIN(HEADING)))
```

or an exponential curve:

```
FORWARD S
RIGHT (SIN(HEADING) * SIN(HEADING) * COS(HEADING))
```

5. Some Questions To Explore

(a) Why does AMPOLY have K "points" if K is an odd integer greater than 1, but 2K if K is even ($K > 1$):



FUNCTION PLOT **\$24.95**



APPLE INVADERS GAME



I-CHING	\$15.95
---------	---------



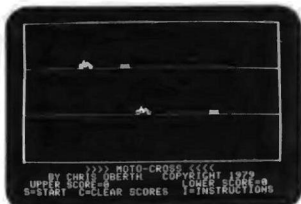
TRIVIA BOX **\$19.95**



CASSETTE	\$15.95
-----------------	----------------



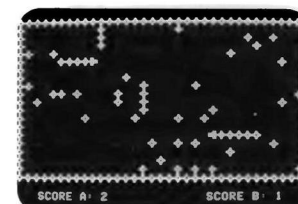
SHAPE BUILDER **\$19.95**



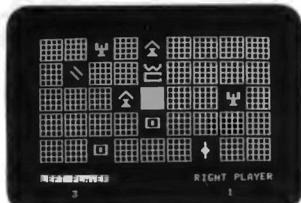
MOTO-CROSS **\$9.95**



DISKETTE	\$19.95
----------	---------



BLOCKADE **\$9.95**



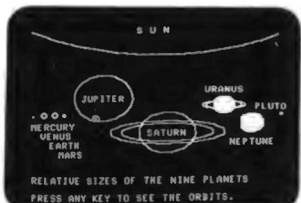
FRUSTRATION	\$9.95
-------------	--------



GUIDED MISSILE	\$15.95
----------------	---------



LASER BLAST **\$9.95**



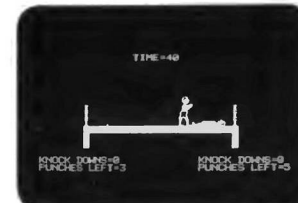
THE PLANETS	\$15.95
--------------------	----------------

AND MORE...

ACTIVE FILTERS	\$24.95
ALIEN INVASION	9.95
AMPERSORT II	15.95
APPLE ALLEY	6.95
BASEBALL	15.95
BATTLEFIELD	9.95
BREAKTHRU	9.95
CHECK BOOK	34.95
DATABASE MAILER	29.95
DEATH RACE	15.95
EARTH QUEST	19.95
HOME BUDGET	24.95
HOUSEHOLD FINANCE	24.95
MINI GENERAL LEDGER	59.95
MOUSE HOLE	6.95
PEG JUMP	9.95
RICOCLETTE	9.95
STAR VOYAGER	15.95
STUNT CYCLE	15.95

All orders must include 3% postage and handling. California residents add 6% sales tax. VISA and MASTERCARD accepted.

Apple II is a trademark of Apple Computers, Inc.



BOXING **\$9.95**



ALGEBRA I \$15.95



SPACE WAR **\$9.95**



SIRIUS **\$15.95**

CIRCLE 184 ON READER SERVICE CARD

**PROGRAMMA
INTERNATIONAL, Inc.**
3400 Wilshire Blvd.
Los Angeles, CA 90010
(213) 384-0579
384-1116
384-1117

Dealer Inquiries Invited

Sine POLYs, con't...

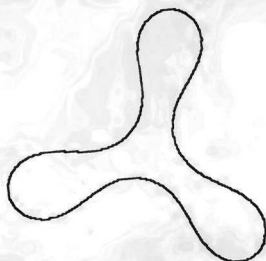


Figure 6: FMPOLY, CF=2, MF=6, MI=8.

(b) AMPOLYs can also be drawn by another quite different program. It's generally called DUOPOLY, and consists of the repeated loop:

```
SET HEADING TO BE H1
FORWARD S1
RIGHT A1
SET H1 TO BE HEADING
SET HEADING TO BE H2
FORWARD S2
RIGHT A2
SET H2 TO BE HEADING
```

Among other things, this program can draw hypocycloids and epicycloids, figures representing the addition of two circular motions where the rate of circumference travel is the same for both. Show that AMPOLYs are hypocycloids and epicycloids, and find the exact relationship between the AMPOLY arguments and the parameters of the circular motions. (You might want to do reading about amplitude modulation.)

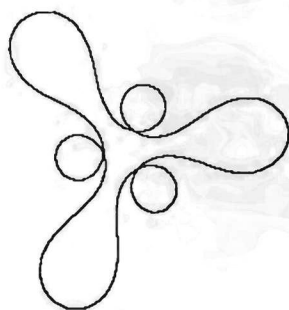


Figure 7: FMPOLY, CF=2, MF=3, MI=9.

(c) DUOPOLY (see above) draws ellipses when $A1 = -A2$. Find out why. Then investigate why FMPOLY when $MF = 2 * CF$, the repeated loop

```
FORWARD SIN(HEADING)*SIN(HEADING)
RIGHT A
```

and the repeated loop

```
FORWARD(1 + (COS(2 * HEADING)))
RIGHT A
```

all draw things that look sort of like ellipses but not quite. (You might want to do some reading about frequency modulation, and you might try the preceding question first.)

(d) One interpretation of that last variant in the preceding question is that the turtle is "going against a grain" when it moves at HEADINGS near 90 and 270 degrees as opposed to 0 and 180 degrees, so it can't go as far per each "step". Investigate other such figures of the form

```
FORWARD(1 + (B * COS(C * HEADING)))
RIGHT A
```

Why is the figure closed for $B = 1$ and $C = 2$ but not for $B = 1$ and $C = 1$?

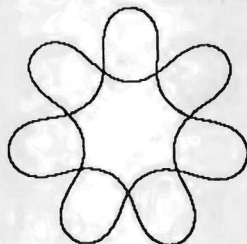


Figure 8: FMPOLY, CF=2, MF=7, MI=8.

(e) (for people who know some physics) Try to model a "turtle on a sloping plane" where the amount of effort the turtle must exert against gravity and friction per each "step" it takes is constant (hence step size must vary with direction). Try drawing some figures on this slope and see how they're distorted. Maybe try some irregularly sloping surfaces.

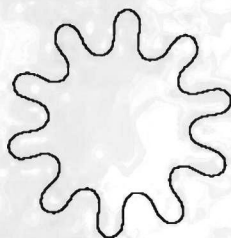


Figure 9: FMPOLY, CF=2, MF=20, MI=30.

(f) Suppose you write a program for a turtle in one of these "warped" or "sloped" worlds. How can such a turtle compensate so as to draw the correct thing, even when it doesn't know exactly how it's being warped? (Maybe use drawing a POLY as an example. You might want to read about numerical analysis.)

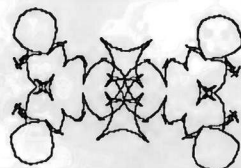


Figure 10: HSQUARE, A=17.

(g) Why is the following repeated loop a closed figure only when P is even?

```
FORWARD(SIN(HEADING) TO THE PTH POWER)
RIGHT A
```

(Assume A is not 0).

(h) What is the rotational symmetry of an FMPOLY figure in terms of CF, MF, and MI? Can you relate it to the rotational symmetry of POLY?

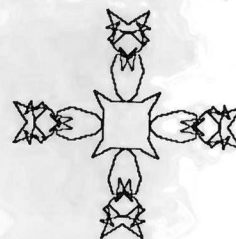


Figure 11: HSQUARE, A=34.

(i) Prove that HSQUARE always has both horizontal and vertical bilateral symmetry for any integer A .

(j) HSQUARE can also have rotational symmetry in addition to bilateral symmetry (as Fig. 12, which has rotational symmetry of order 4). Develop a general method for predicting this as a function of integer A .

(k) Why is it that the only HSQUARE figures that are not closed are for A a multiple of 8? And why not all multiples of 8?

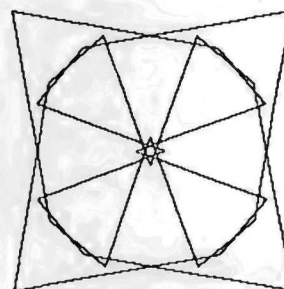


Figure 12: HSQUARE, A=20.

(l) (for people who know some calculus) Show that any curve $f(x)$ can be drawn by the repeated turtle loop.

```
FORWARD 1
RIGHT -(SIN(HEADING) * SIN(HEADING) * f''(x))
```

where $f''(x)$ is the second derivative of $f(x)$ with respect to x . Use this formula to draw arbitrary functions of your choice. (Note that often it's possible to express $f''(x)$ as a function of HEADING, as in the case of the parabola, hyperbola, and exponential curve examples given before.)

(m) How is it that the above formula draws a curve knowing only the second derivative? Don't you "lose information" in finding the derivatives, and don't many functions have the same second derivative? — for instance, $x(x-1)(x+1)$, $(x-1)(x+.85i+.5)(x-.85i+.5)$, and $x*x*x$ all have $6x$.

Sine POLYs, con't...



Figure 13: parabola.

(n) Use (1) to find the "mystery curve" (Fig. 14) drawn by

FORWARD 5
RIGHT SIN(HEADING)

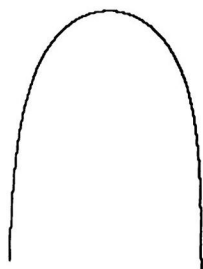


Figure 14: curve for question (n).

For hints on the questions, write to the author at Department of Computer Science, Stanford University, Stanford CA 94305. For many more ideas about turtle drawings, see a forthcoming book, **Turtle Geometry**, by Hal Abelson and Andy DiSessa. □

While LOGO will probably not replace BASIC, it is still an outstanding simple, one-command language which young children can understand and learn quickly. For readers unfamiliar with LOGO, we recommend the following articles:

1. "Turtle Geometry Without Hardware" by McLean and Pagano, *Creative Computing*, Vol. 1, No. 5 (pp 29-31) or *Best of Creative Computing*, Vol. 1, (pp 186-188).
2. "Twenty Things To Do With Your Computer" by Papert and Solomon, *Educational Technology*, Vol. 9, No. 4 (pp 39-42).
3. "Toward A Human Computer Language" by A.B. Cannara, *Creative Computing*, Vol. 1, No. 5 (pp 38-40) or *Best of Creative Computing*, Vol. 1, (pp 58-60).
4. "Grammar As A Programming Language" by N. Rowe, *Creative Computing*, Vol. 4, No. 1, (pp 80-85).
5. "A Comparison of the Problem Solving Style of Two Students Learning LOGO" by Daniel Watt, *Creative Computing*, Vol. 5, No. 12.

OHIO SCIENTIFIC CIP

Graphics design program — draw any image directly on screen using movable cursor without POKE. Make full use of graphics. Store image on tape.

Requires 8K. \$12.00

4K version w/o tape storage \$10.00

Plot program — plot any function over any range on X-Y coordinates. Plot scattergram for any data.

Requires 4K. \$10.00

BENCHMARK SOFTWARE

7533 Venetian Way
Indianapolis, IN 46217

FREE SOFTWARE CATALOG

For a free 20-page catalog of Creative Computing Software for all computers circle 217 on the reader service card.

← CIRCLE 118 ON READER SERVICE CARD

Plug in and Print



Built-in Interface for TRS-80, PET and Apple II Computers

Axiom has made it simple for TRS-80, PET and Apple II users. Just go to your computer store and pick up an Axiom printer with the appropriate built-in interface. Take it home, plug it in and start printing. We even supply the cable and connector.

There are two models: The EX-801 prints upper and lower case alphanumeric characters and all the

graphic symbols used by your computer. The EX-820 goes a step further, providing precise alignment of both horizontal and vertical dot patterns for a true hardcopy of computer generated graphics. With up to 128 dots per inch resolution, the only limit is your imagination. The price is right, too. Just \$535 for the EX-801. So visit your computer store today.

AXIOM

5932 San Fernando Road, Glendale, CA 91202
(213) 245-9244 • TWX: 910-497-2283

CIRCLE 116 ON READER SERVICE CARD