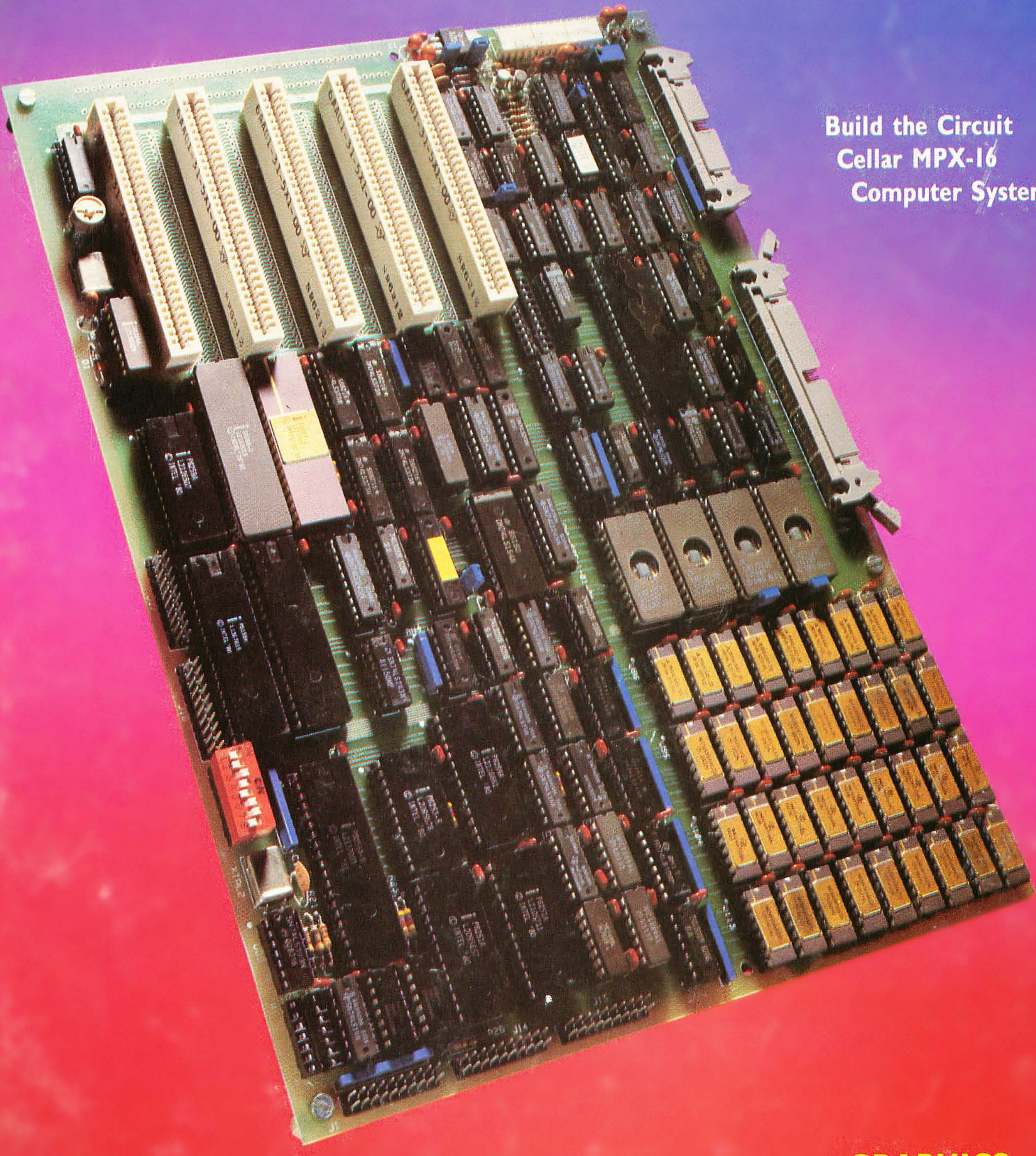
BYTE®

NOVEMBER 1982 Vol. 7, No. 11
$2.95 in USA
$3.50 in Canada/£1.85 in U.K.
A McGraw-Hill Publication

the small systems journal



Build the Circuit Cellar MPX-16 Computer System

GRAPHICS

In The Queue

BYTE Volume 7, Number 11 November 1982

Features

30 The Third NCGA and the Future of Computer Graphics by Alexander Pournelle / An overview of the state of the art in computer graphics as gleaned from a day at the fair.

48 Tronic Imagery by Peter Sørensen / A behind-the-scenes look at the development of the computer-generated graphics in Disney Studio's film Tron.

78 Build the Circuit Cellar MPX-16 Computer System, Part 1 by Steve Ciarcia / Any peripheral device designed to be installed in the IBM Personal Computer can be plugged into this 8088-based system.

118 Problem Solving with Logo by William Weinreb / Find out how a complex design can be broken down into surprisingly simple components.

174 Build a Video Digitizer by Michael Keryan / Capture any video image for processing by your computer.

194 Computer Animation with Color Registers by David Fox and Mitchell Waite / The color registers on the Atari 400 and 800 give programmers amazing animation capabilities, even in BASIC.

216 Victor Victorious: The Victor 9000 Computer by Phil Lemmons / A detailed look at a third-generation microcomputer that really gets down to business.

256 An Interview with Chuck Peddle by Phil Lemmons / The chief designer of the Victor 9000 discusses microcomputer design, marketing, and the industry's future.

272 JETSET by Eugene Szymanski / You'll thrill to the highs and lows of this simulated flight game. It's the fourth-place winner in the BYTE Game Contest.

336 The Game of Rat and Dragon by Truck Smith / You really put your game-playing skills to the test in this chase in which dragons pursue a rat that's after some cheese. As your skills improve the pace quickens. It captured third place in the BYTE Game Contest.

379 An Introduction to the Human Applications Standard Computer Interface, Part 2: Implementing the HASCI Concept by Chris Rutkowski / Details of an easy-to-use, consumer-quality computer console.

386 A Short History of the Keyboard by Phil Lemmons / The widespread use of keyboards as input and control devices for microcomputers has generated renewed interest in an old problem.

394 User's Column: Terminals, Keyboards, and How Software Piracy Will Bring Profits to Its Victims by Jerry Pournelle / The columnist answers a few letters and passes on some interesting observations from readers.

416 Inexpensive Transducers for the TRS-80 by William Barden Jr. / A practical look at the devices that put real-world interfaces to work.

448 A Graphics Primer by Gregg Williams / Microcomputers can create quite a variety of graphics.

474 Interactive 3-D Graphics for the Apple II by Andrew Pickholtz / Understanding the theory of perspective helps you to represent three-dimensional objects on a two-dimensional screen.

508 Microvec: The Other Type of Video Display by Billy Garrett / Vector displays produce images far sharper than "high-resolution" raster types.

Reviews

138 The Graphics Magician by Peter Callamaras

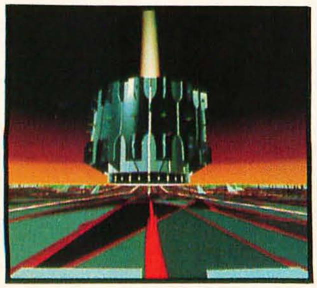
148 Cambridge Development Lab's High-Resolution Video Graphics System by James R. DeKock

164 Executive Briefing System by Peter Callamaras

324 Colonial Data Services' SB-80 by Arthur Little

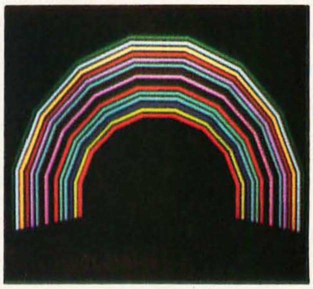
Nucleus

6 Editorial: Deus ex Machina of the Technological Age
14 Letters
529 Software Received
532 Ask BYTE
534 Event Queue
539 Clubs and Newsletters
540 BYTELINES
548 Books Received
553 What's New?
605 Unclassified Ads
606 BOMB, BOMB Results
607 Reader Service

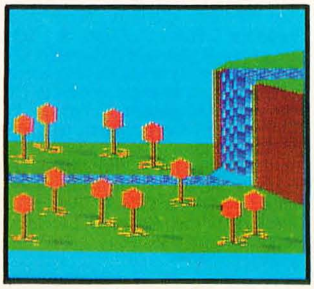


Page 48

Page 138



Page 148



Page 194

Problem Solving with Logo

Using Turtle Graphics to Redraw a Design

William Weinreb
Logo Computer Systems Inc.
368 Congress St.
Boston, MA 02210

Every computer graphics artist has sat before a console and created a design. This article describes a slightly different experience: *recreating* a design, i.e., solving the riddle of an already existing pattern.

It began when a friend presented me with a pattern (see figure 1) devised by Christopher Keavney for the Massachusetts Institute of Technology's student information-processing board. My friend also furnished the three-page PL/I program that generated it. Thinking that it might be possible to draw the design using turtle graphics instead of PL/I graphics, she challenged me and a colleague, Glenn Forester, to write a Logo program for the design. We accepted the challenge because we too were struck by how similar the design was to pictures we had drawn using turtle graphics. In this article, I recount our attempts, initial failure, and eventual success at writing such a program, which, remarkably, consisted in the end of only seven short Logo procedures that can be run on an Apple II using Apple Logo.

Searching for Clues

Imagine that someone handed you a pattern and a program that draws it and said, "Write a program in another language which will draw this." Where would you begin? I chose to concentrate on understanding the program, hoping this would reveal clues to the structure of the design. Instead of bringing me closer to a solution, however, this approach led me astray. As it turned out, the PL/I program for this design consists of several obscure subroutines, each responsible for calculating a series of Cartesian coordinates that are the endpoints of lines in the design. To me this was very discouraging news: even if I translated the entire program, I would wind up with nothing more helpful than a few point-plotting algorithms written in Logo instead of PL/I. Plotting points in Logo is not much different from plotting points in any other language. What made the project exciting was the prospect of writing a simpler and more elegant program using turtle graphics. Clearly the original program would be no help to me here.

Logo encourages you to solve a problem by breaking it down into smaller pieces, or subproblems, that are more easily grasped.

So I put it aside for good and refocused my attention on the design. Seeking a new approach, I recalled that Logo encourages one to solve a problem by breaking it down into smaller pieces, or subproblems, that are more easily grasped.

A Piece of the Puzzle Is Solved

Applying this technique of breaking down problems to the design led to the first crucial breakthrough. Glenn was certain that the design's border was made up of only one or two patterns repeated many times. Eventually he figured out that the rim of the pattern is merely pentagonal spirals arranged in such a way that

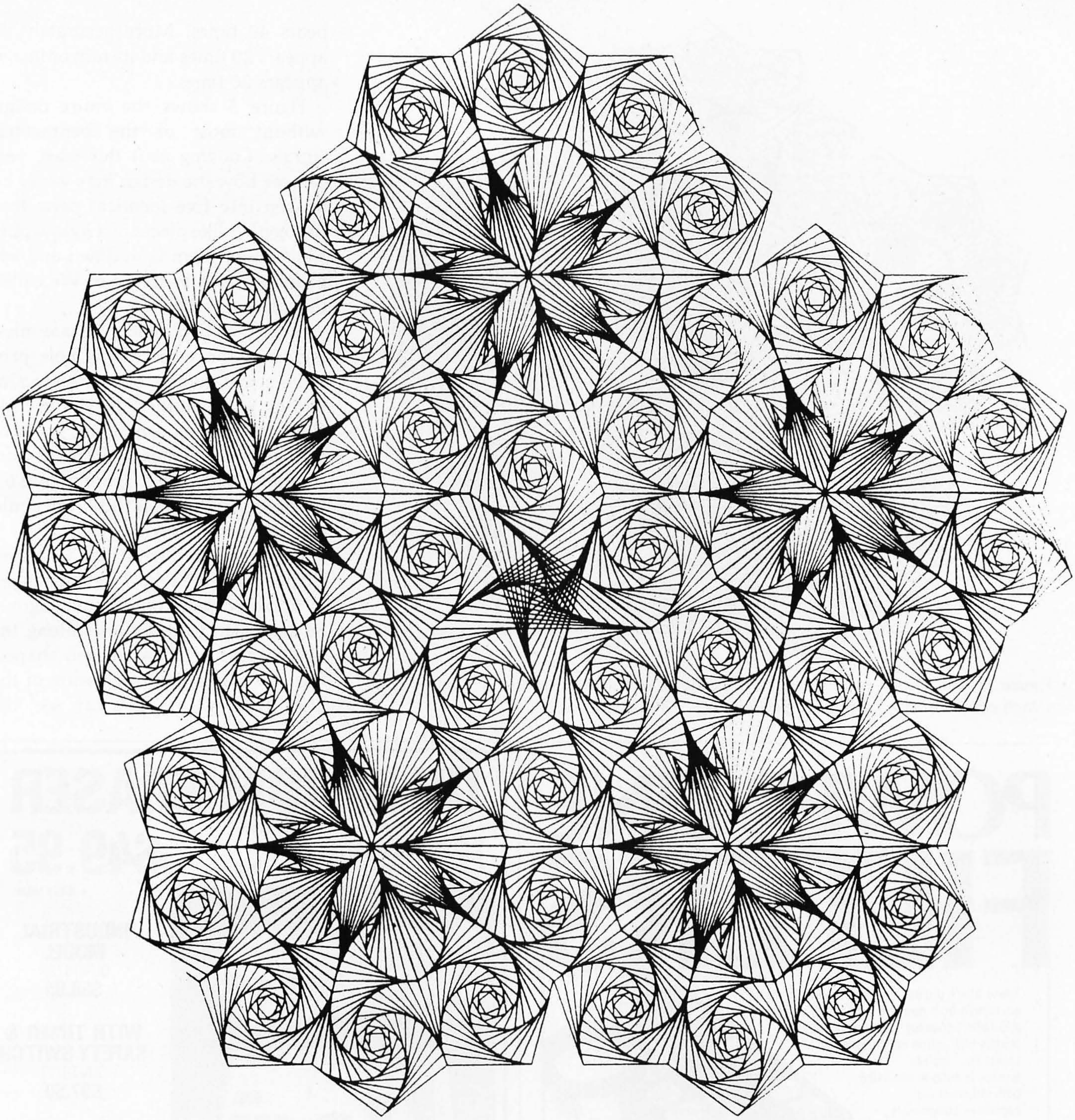


Figure 1: *The challenge was to reproduce this design using Logo. Could this complex design, which was produced by a three-page PL/I program, be redrawn with a simple Logo program?*

their edges blend and are difficult to distinguish (see figure 2). Each spiral fits inside a pentagon. The spiral figure is one that Glenn and I recalled from other turtle drawings; it is a pattern made by a Logo procedure sometimes called POLYSPI.

What is a POLYSPI? As we have worked with Logo turtle graphics, certain procedures and the figures they create have become part of our vocabulary. Two such procedures are familiarly called POLY and POLYSPI (see listing 1). POLY can produce polygons, and figure 3 shows some of the patterns you can produce by giving POLY different inputs. The procedure POLYSPI is like POLY except that in the recursive call, we make the turtle draw a new side that is a little shorter than the last. These ever-shortening sides make the final figure a spiral. Figure 4 shows some figures POLYSPI can draw.

Notice that when POLYSPI is given inputs of 20 and 75, it draws a pentagonal spiral. Now take a close look at the full design (figure 1). The same spiral fills the pentagons that we found in the design. This spiral ap-

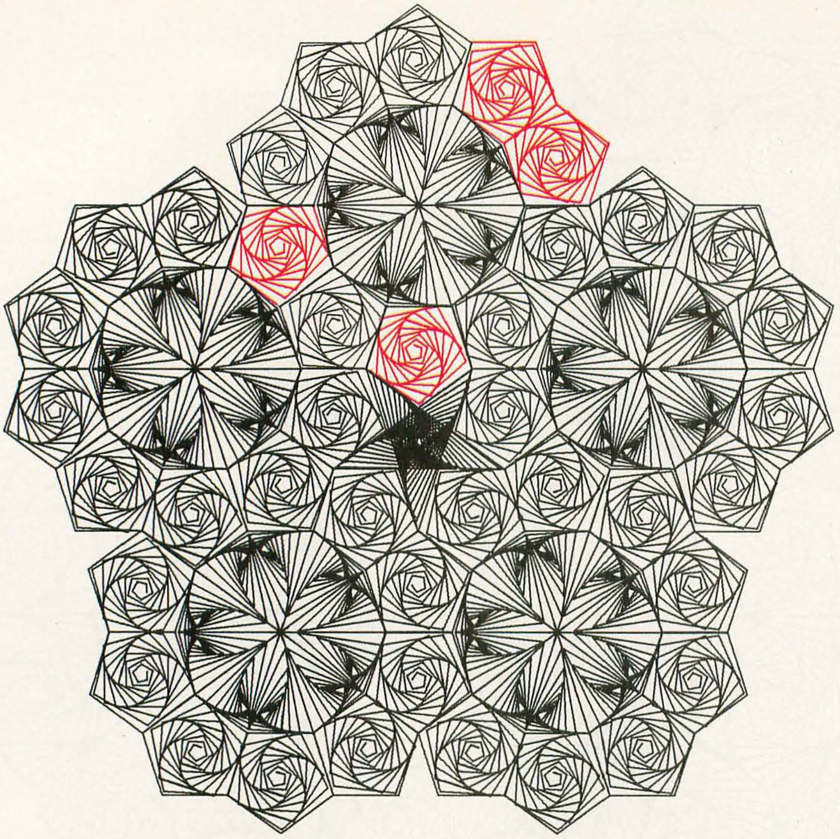


Figure 2: *The design is composed primarily of pentagonal spirals (shown in color), which can be drawn easily using a Logo* POLYSPI *procedure.*

pears 40 times. More accurately, it appears 20 times and its mirror image appears 20 times.

Figure 5 shows the entire design without most of the pentagonal spirals. Looking at it this way, you can see how the design may easily be broken into five identical parts that fit together like pieces in a jigsaw puzzle. One of them is outlined and appears by itself in figure 6. We called this piece a "wheel."

Note that two pentagons are missing from the wheel. This hole provides a space where one wheel can interlock with another. Now if we include the missing pentagons, we have a ring of ten pentagonal spirals surrounding a ten-sided figure (figure 6). What type of figure is this? And could this be broken down as well?

It seemed to Glenn that whatever shapes filled the wheel were arranged in mirrored pairs like the pentagons. So he began to draw lines along the likely edges of these alleged shapes. Figure 7 traces the progression of the lines he drew. As you can see, the

Listing 1: *The logo procedures* POLY *(left) and* POLYSPI *(right).*

```
TO POLY :SIDE :ANGLE
FD :SIDE
RT :ANGLE
POLY :SIDE :ANGLE
END
```

```
TO POLYSPI :SIDE :ANGLE
FD :SIDE
RT :ANGLE
POLYSPI :SIDE - 1 :ANGLE
END
```

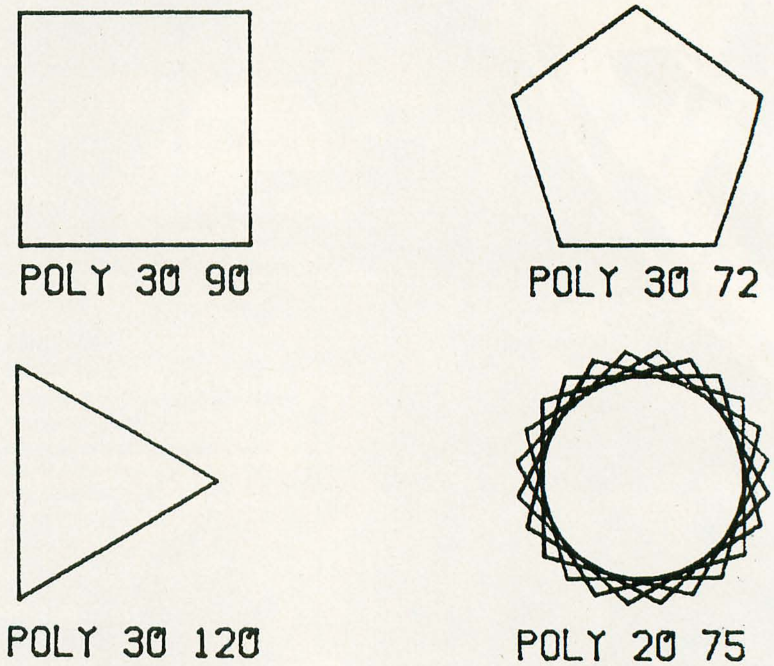


Figure 3: *The result of the* POLY *procedure of Logo. Many different shapes can be produced merely by changing the angle of the "turtle-turn."*

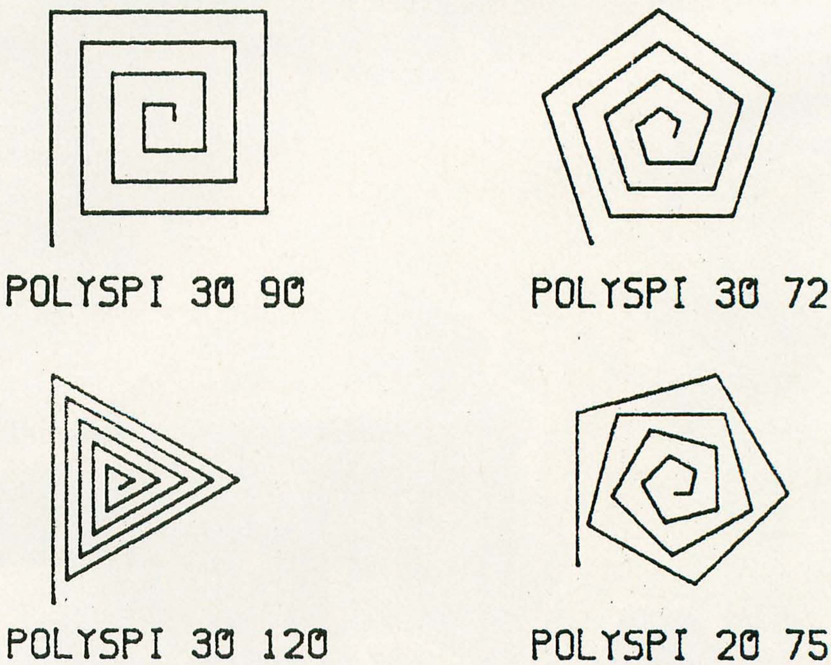


Figure 4: *In the* POLYSPI *procedure, the length of each successive side is shortened by one unit. The result is very similar to a spiral.*

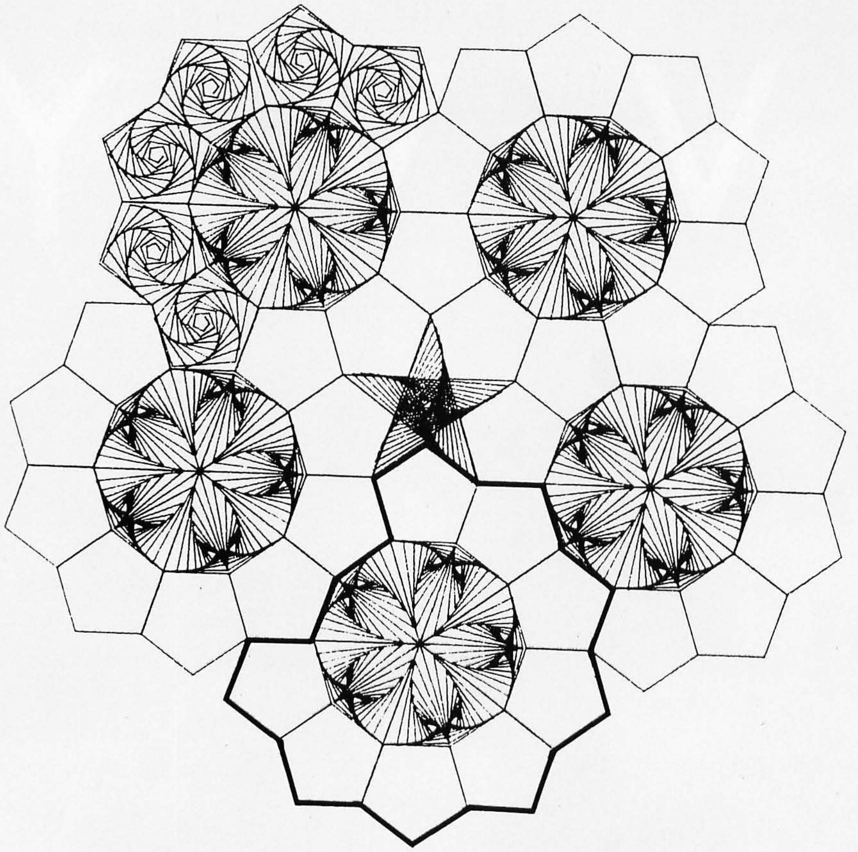


Figure 5: *With most of the spirals removed, you can see that the design can be broken down into five wheels that interlock with each other.*

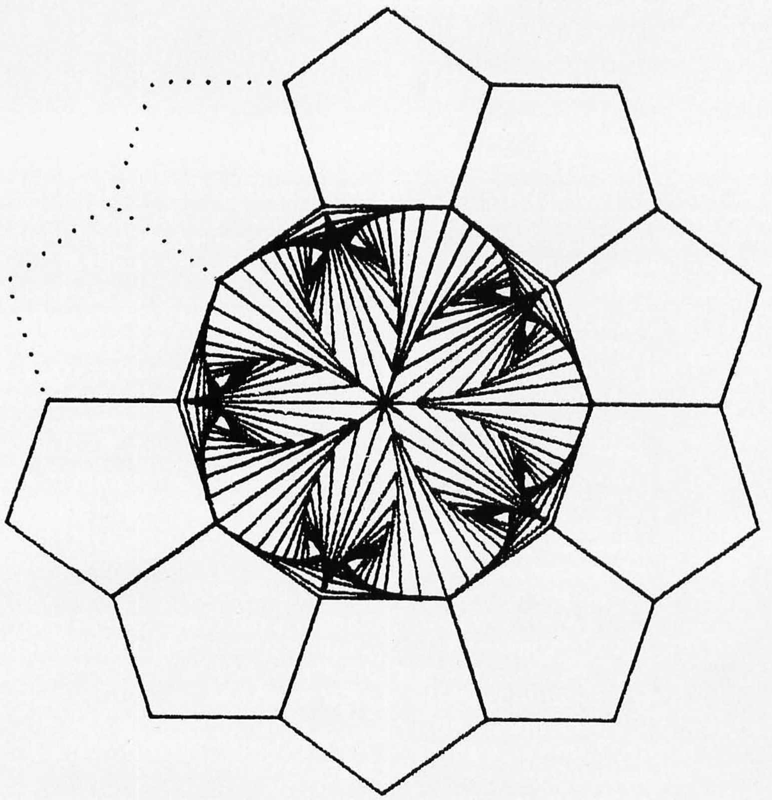


Figure 6: *A close-up of one of the wheels in figure 5. With two pentagonal spirals added, it becomes a complete wheel. The problem now: What is the design in the middle of the wheel?*

shapes revealed by these lines are triangles. They were much harder to recognize than their pentagonal neighbors; they emerged from the dissection of the design visually when lines were drawn.

It didn't occur to us at first that the figures inside the triangles might be spirals. However, the figure inside each triangle is a version of our friend POLYSPI, except that the triangles in the design are isosceles, not equilateral. The way a spiral based on an irregular polygon looks is different from what we were used to. The triangular spirals were also hard to identify because their sides get shorter very quickly. Figuring out how to draw these unusual spirals took Glenn a lot of time when we began to write the program. Figure 8 shows a triangular spiral, its mirror image, and both placed back to back. Five such pairs arranged in a circle make up the interior of each ring.

We arranged the five wheels more or less into a ring (figure 9). We were then left with a star-shaped hole in the center, which we filled with a star-shaped spiral (figure 10).

Success: Writing a New Program in Logo

Once we had divided the design into smaller pieces, it was relatively easy to write a program to produce it (see listing 2, page 132). First, we wrote five procedures, one for each version of the design's three basic figures. The procedures PENTR and PENTL draw pentagonal spirals, TRIPOLYR and TRIPOLYL triangular ones, and CENTERPIECE draws the central, star-shaped spiral.

PENTR draws a spiral that curves to the right. PENTL draws its mirror image. (Because this is the only difference between the two, I will limit my remarks to PENTR.) Note that PENTR differs from the procedure POLYSPI mentioned earlier. In PENTR the first instruction is a conditional statement; it stops the procedure when the value of SIDE gets smaller than 2. TRIPOLYR, the procedure that draws right-curving triangular spirals, differs from PENTR in three important ways. The lengths of the three sides in the underlying shape of this spiral are

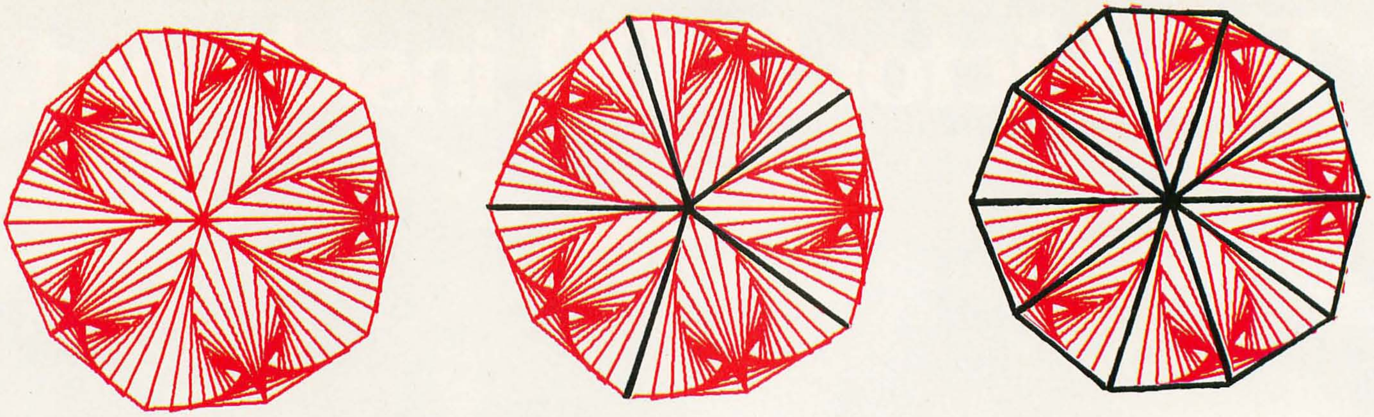


Figure 7: *The center of the wheel in figure 6 can be broken down into 10 isosceles triangles.*

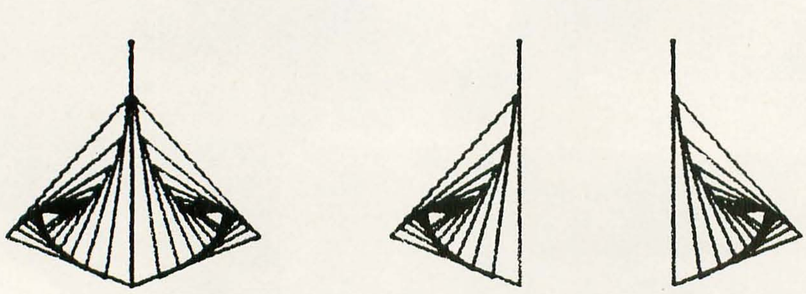


Figure 8: *We discovered that each of the isosceles triangles in figure 7 was a spiral of an irregular polygon.*

not equal. Also, triangles with sides of different lengths must have unequal angles. The "turtle-turn" at two vertices of this triangle is 111 degrees, while the third vertex has a turtle-turn of 146 degrees. The last difference between TRIPOLYR and PENTR is in how SIDE gets shorter each time a side is drawn. In PENTR, you reduce SIDE by subtracting .38 from each previous side. In TRIPOLYR, SIDE is *multiplied* by .75

 Circle 222 on inquiry card. Circle 223 on inquiry card.

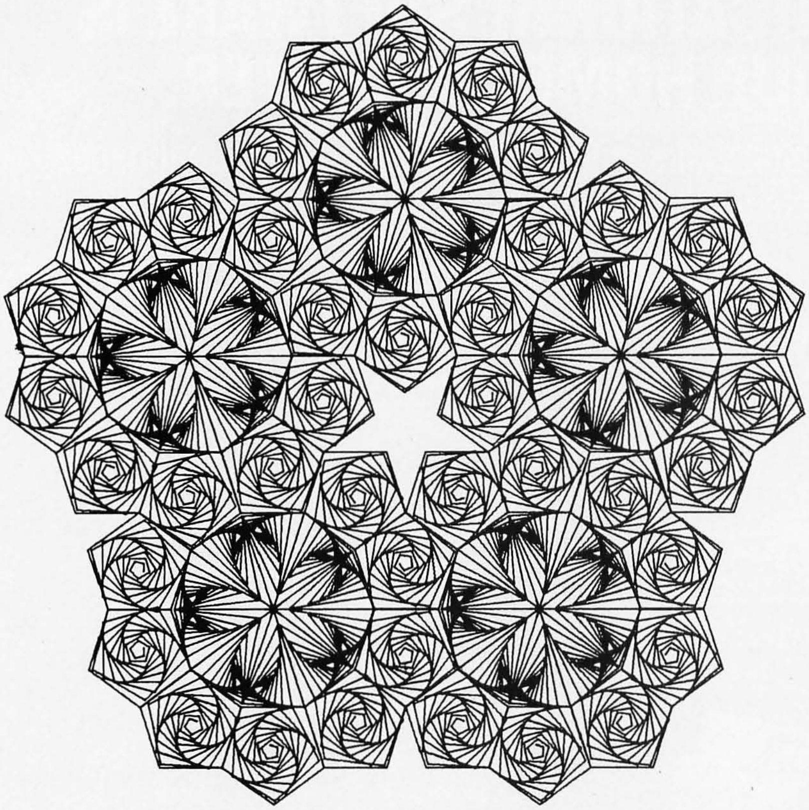


Figure 9: *Our design elements assembled together. The last remaining element is a star-shaped spiral in the center.*

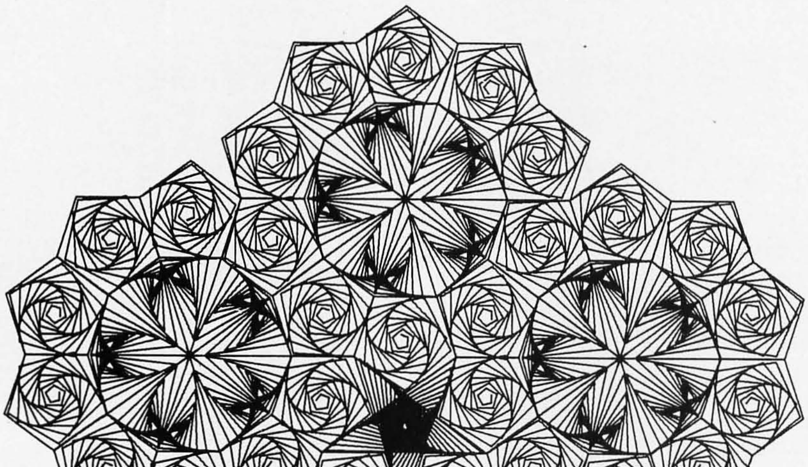


Figure 10: *Our completed Logo design. Using a simple Logo program we have made a good reproduction of a very complex design (figure 1).*

at the end of each three-side cycle; thus, SIDE gets shorter faster.

This precise combination of side lengths and angle measurements and the faster spiraling technique is required to produce the spiral that appears in figure 8. Glenn arrived at the final version of TRIPOLYR after much trial and error. The resulting triangular spiral is a very close approximation to the one in the original design.

Lastly, CENTERPIECE, which draws the central star-shaped spiral, is almost identical to the procedure POLYSPI.

Moving to a Higher Level

The twin procedures PENTRPIECE and TRIPIECE are the next level up in the program; they build bigger chunks of the design by supervising the work of PENTR, PENTL, TRIPOLYR, and TRIPOLYL.

PENTRPIECE's job is to supervise PENTR and PENTL. One PENTRPIECE procedure produces a pair of mirrored pentagonal spirals. Four consecutive PENTRPIECE s make a ring of eight pentagonal spirals. TRIPIECE performs a similar function by supervising TRIPOLYR and TRIPOLYL. Five consecutive calls to TRIPIECE make a ring of five pairs of mirrored triangular spirals.

WHEEL makes one of the five large, identical sections that fit together to form the entire design. The command REPEAT 4 [PENTRPIECE LT 72] makes the rim of one wheel. Then REPEAT 5 [TRIPIECE] fills the newly created wheel with pairs of mirrored, triangular spirals.

It is common to build one procedure in a Logo program that more or less gets the whole thing started. DESIGN supervises the work of WHEEL and makes sure that each of the five wheels interlock. It also has CENTERPIECE fill the central hole with a star-shaped spiral.

Listing 2 shows the complete program for an Apple II using Apple Logo. You can produce the design on the display screen of the Apple, but it will be rather dense. You can get a more interesting, but somewhat cropped, version by multiplying each

Listing 2: *Our Logo program, entitled* DESIGN, *will redraw the design in figure 1.*

```
TO DESIGN
CS HT PU WINDOW
REPEAT 5 [FD 64.65 PD WHEEL POS PU BK 64.65 RT 72]
PU HOME RT 36 FD 24.5 RT 198 PD
CENTERPIECE 46 143.4
END


TO WHEEL :INITPOS
RT 54 REPEAT 4 [PENTPIECE]
PD LT 36
REPEAT 5 [TRIPIECE]
LT 36
REPEAT 5 [PD RT 72 FD 28 PU BK 28]
LT 54
END


TO TRIPIECE
LOCAL "OLDH MAKE "OLDH HEADING
PD BK 2.5
TRIPOLYR 31.5
PU SETPOS :INITPOS SETH :OLDH
PD BK 2.5
TRIPOLYL 31.5
PU SETPOS :INITPOS SETH :OLDH
LT 72
END


TO PENTPIECE
LOCAL "OLDH MAKE "OLDH HEADING
PU FD 29 PD
REPEAT 5 [FD 18 RT 72]
PENTR 18 75
PU SETPOS :INITPOS SETH :OLDH
FD 29 PD
REPEAT 5 [FD 18 LT 72]
PENTL 18 75
PU SETPOS :INITPOS SETH :OLDH
LT 72
END


TO PENTL :SIDE :ANG
IF :SIDE < 2 [STOP]
FD :SIDE
LT :ANG
PENTL :SIDE - .38  :ANG
END


TO PENTR :SIDE :ANG
IF :SIDE < 2 [STOP]
```

Listing 2 continued on page 134

Listing 2 continued:

```
FD :SIDE
RT :ANG
PENTR :SIDE - .38 :ANG
END


TO TRIPOLYR :SIDE
IF :SIDE < 4 [STOP]
FD :SIDE
RT 111
FD :SIDE / 1.78
RT 111
FD :SIDE / 1.3
RT 146
TRIPOLYR :SIDE * .75
END


TO TRIPOLYL :SIDE
IF :SIDE < 4 [STOP]
FD :SIDE
LT 111
FD :SIDE / 1.78
LT 111
FD :SIDE / 1.3
LT 146
TRIPOLYL :SIDE * .75
END


TO CENTERPIECE :S :A
FD :S LT :A
IF :S < 7.5 [STOP]
CENTERPIECE :S - 1.2 :A
END
```

of the inputs to the FD and BK commands by two.

We produced our completed design on a Houston Instrument Hiplot plotter using a plotter-interface procedure written by Peter Cann. You can obtain these instructions from Logo Computer Systems (368 Congress St., Boston, MA 02210). Unfortunately, this interface procedure will work only with Apple Logo.

As you can see from figure 10, our design compares pretty well with the original. The important point of this exercise is that a complex design can be broken down into simpler components. And this approach, which is part of the basic philosophy behind Logo, can be used to solve practically any problem. ■