J. Webster (ed.), *Wiley Encyclopedia of Electrical and Electronics Engineering*

LOGO

Logo is a computer programming language whose goal is to facilitate learning by facilitating student programming. The idea behind Logo was to place programming in the hands of children, with the firm belief that, through programming, students would come to explore and learn a great deal. Logo was meant not only to support learning content areas (such as science, mathematics, and linguistics) but also metaknowledge, such as how to plan and how to solve problems.

Early History of Logo

Logo was invented in the late 1960s by Wally Feurzeig, Danny Bobrow, and Seymour Papert at Bolt, Beranek, and Newman. It is a direct descendant of Lisp. Logo has been referred to as "Lisp without parentheses." The early uses of Logo were on teletype machines, and students wrote programs that emphasized natural-language exploration.

Later, Logo was used to control a robot turtle. The turtle could be told to go forward or backward a specified number of "turtle steps" and could be told to turn right or left a specified number of degrees. The turtle carried a pen, which could be carried "up" (not touching the ground or paper) or "down" (drawing a path as the turtle moved). Using a turtle, students could explore a new kind of geometry that Papert called "turtle geometry." A later book by Abelson and diSessa (1) showed that turtle geometry was very rich and could be used, for a wide range of mathematical exploration.

A Logo program for drawing a square might look like this:

```
TO SQUARE :SIZE
REPEAT 4 [FD :SIZE RT 90]
END
```

This procedure defines the word SQUARE, which could be used for example, by executing the command SQUARE 10. The procedure would then execute the list [FD :SIZE RT 90] four times. This list instructs the turtle to move forward (FD) the size passed in as input to SQUARE, then to turn right (RT) 90°. Doing this four times creates a square of the specified size.

As Logo implementations moved to microcomputers (originally the Apple II and Texas Instruments' personal computer), the turtle changed from a robot to an on-screen representation. The focus of Logo use shifted from language exploration to science and mathematics. More recent Logo implementations, from manufacturers LCSI and Terrapin, support student manipulation of a wide variety of media (e.g., movies and sounds, in addition to text and graphics), which has helped make Logo useful across the curriculum. The multimedia production tool, HyperStudio, has adopted a form of Logo for its scripting language.

Logo In Education. Whereas Logo was born at BBN, it grew up at MIT, under the direction of Professor Seymour Papert. The MIT Logo group advanced the unusual prospect of students as programmers, and even as creators, of knowledge. The MIT Logo technical report series included articles like "Teaching children to be mathematicians versus teaching about mathematics" (2).

Papert's views on Logo first received wide attention in 1980 with his book *Mindstorms* (3), where he talked about children learning "powerful ideas" that would change how they would approach knowledge. Papert used his own experience as a child with gears as an analogy. He felt that his play with gears as a child made him a better mathematician later in life. Through programming, Papert believed that students would learn to think about knowledge and learning differently. By viewing programming as "teaching the computer," students could be given the opportunity to think about representations of knowledge and their own learning and knowing.

After the publication of *Mindstorms* and the first implementations of Logo on microcomputers, the popularity of Logo soared. Many teachers brought Logo into their classroom, and books and curricular units on Logo came out in droves. International Logo conferences were held at MIT during the mid-1980s where teachers and researchers from around the world talked about how they might use Logo.

Logo in Educational Research

The broad claims of *Mindstorms* were not supported by empirical research. The most famous of the studies on Logo was the work of Roy Pea and Midian Kurland, which showed that, under the curricula they studied, many students were not learning Logo well (4) and showed few signs of applying their knowledge of Logo in new contexts (5). As cognitive science was learning in many situations, transferring knowledge from one situation to another is very hard, and many studies showed that it happened in programming only rarely (6). The form and methodology of the earlier studies of Logo have come under scrutiny (7). Later studies of Logo did show transfer of skill from Logo to other tasks through careful design of the curriculum to inculcate transferrable knowledge (e.g., how to plan) from programming to other domains (e.g., Ref. 8).

In the late 1980s, the emphasis of research on Logo in education shifted. Rather than viewing programming in Logo as an activity to lead to metaknowledge, Logo was viewed as a rich medium in which students could construct and design. With a theoretical perspective from David Perkins on viewing design as a learning activity (9), Idit Harel used Logo as a design medium in which students were creating software to teach mathematics to younger students (10). Harel found that, through programming, students came to a deeper understanding of mathematics than a control group. Her work led to others exploring Logo as a design medium and exploring design as an opportunity for learning. The work has been continued and expanded by Yasmin Kafai, who has been exploring the use of Logo by students to build video games (11).

Microworlds in Logo

An important direction for educational technology that Logo initiated is the development of open-ended exploratory *microworlds*. A microworld is a restricted simulation in which a student can learn by experimentation and construction. A microworld can provide access to a set of concepts for students in a playful space. Turtle geometry in Logo is one form of microworld.

Many microworlds have been created in Logo, especially in mathematics (12), and their success has led to microworlds being created in other languages as well. Popular simulation games like SimCity have been influenced by the Logo microworlds work. The programming language Boxer (13), developed by Andrea diSessa, is a more powerful descendant of Logo that is explicitly aimed at supporting microworld creation and exploration.

Logo Today

Logo implementations are still available today, and Logo research continues. Today, the Logo Foundation (http://el.www.media.mit.edu/logo-foundation/) serves as the clearinghouse for Logo information around the world. Research at MIT has taken Logo away from simple, individual turtles and toward the use of Logo for exploring thousands of turtles at once through parallel programming (14,15), for controlling external devices created with Lego (16), and for exploring a range of external media, such a text-based virtual realities (17,18). Logo still influences education and educational research.

Logo as a programming language has left a lasting legacy. Logo directly influenced Alan Kay and the Smalltalk programming language (19), and, as the first object-oriented programming language, Smalltalk has led to a whole new paradigm of programming. The research on Logo created a subfield of researchers studying novice and children programmers (e.g., Ref. 20), which continues today through conferences like the Empirical Studies of Programmers Workshops (e.g., Ref. 21).

BIBLIOGRAPHY

1. H. Abelson A. A. diSessa *Turtle Geometry: The Computer as a Medium for Exploring Mathematics*, Cambridge, MA: MIT Press, 1986.
2. S. Papert Teaching children to be mathematicians versus teaching about mathematics, AI Memo No. 249 and Logo Memo No. 4, Cambridge, MA: MIT, 1971.
3. S. Papert *Mindstorms: Children, Computers, and Powerful Ideas*, New York: Basic Books, 1980.
4. D. M. Kurland C. A. Clement R. Mawby R. D. Pea Mapping the cognitive demands of learning to program, in R. D. Pea and K. Sheingold (eds.), *Mirrors of Minds*, Norwood, NJ: Ablex, 1996, pp. 103–127.
5. R. D. Pea D. M. Kurland On the cognitive effects of learning computer programming, in R. D. Pea and K. Sheingold (eds.), *Mirrors of Minds*, Norwood, NJ: Ablex, 1986, pp. 147–177.
6. D. B. Palumbo Programming language/problem-solving research: A review of relevant issues, *Rev. Educ. Res.*, **60**(1): 65–89, 1990.
7. R. Noss C. Hoyles *Windows on Mathematical Meanings*, Norwell, MA: Kluwer, 1996.
8. D. Klahr S. M. Carver Cognitive objectives in a LOGO debugging curriculum: Instruction, learning, and transfer, *Cogn. Psychol.*, **20**: 362–404, 1988.
9. D. N. Perkins *Knowledge as Design*, Hillsdale, NJ: Lawrence Erlbaum Associates, 1986.
10. I. Harel *Children Designers: Interdisciplinary Constructions for Learning and Knowing Mathematics in a Computer-Rich School*, Norwood, NJ: Ablex, 1991.
11. Y. Kafai *Minds in Play: Computer Game Design as a Context for Children's Learning*, Hillsdale, NJ: Lawrence Erlbaum Associates, 1995.
12. R. Noss C. Hoyles *Windows on Mathematical Meanings: Learning Cultures and Computers*, Norwell, MA: Kluwer, 1996.
13. A. A. diSessa H. Abelson D. Ploger An overview of Boxer, *J. Math. Behav.*, **10**(1): 3–15. 1991.
14. U. Wilensky M. Resnick Thinking in levels: A dynamic systems perspective to making sense of the world, *J. Sci. Educ. Technol.*, **8**(1), 1999.
15. M. Resnick Beyond the centralized mindset, *Learning Sci.*, **5**(1): 1–22, 1996.
16. M. Resnick Lego Logo: Learning through and about design, in I. Harel (ed.), *Constructionist Learning: A 5th Anniversary Collection of Papers*, Cambridge, MA: MIT Media Lab., 1990.
17. A. Bruckman Situated support for learning: Storm's weekend with Rachael, *J. Learning Sci.*, **9**(3): 329–372, 2000.
18. M. Resnick A. Bruckman F. Martin Pianos not stereos: Creating computational construction kits, *Interactions*, **3**(5): 41–50, 1996.
19. A. C. Kay The early history of Smalltalk, in J. E. Sammet (ed.), *History of Programming Languages (HOPL-II)*, New York: ACM, 1993, pp. 69–95.
20. E. Soloway J. C. Spohrer, (eds.) *Studying the Novice Programmer*, Hillsdale, NJ: Lawrence Erlbaum Associates, 1988.
21. C. R. Cook J. C. Scholtz J. C. Spohrer (eds.) *Empirical Studies of Programmers: Fifth Workshop*, Norwood, NJ: Ablex, 1993.

MARK GUZDIAL
Georgia Tech