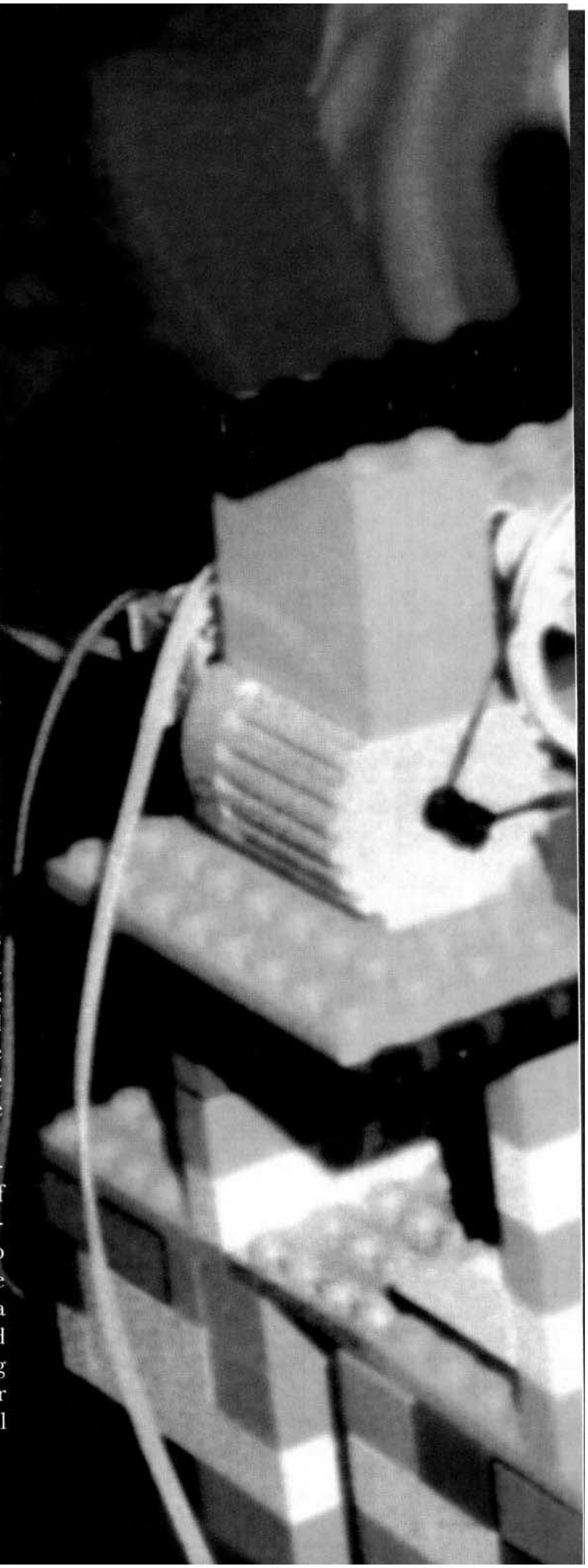
BEHAVIOR CONSTRUCTION KITS

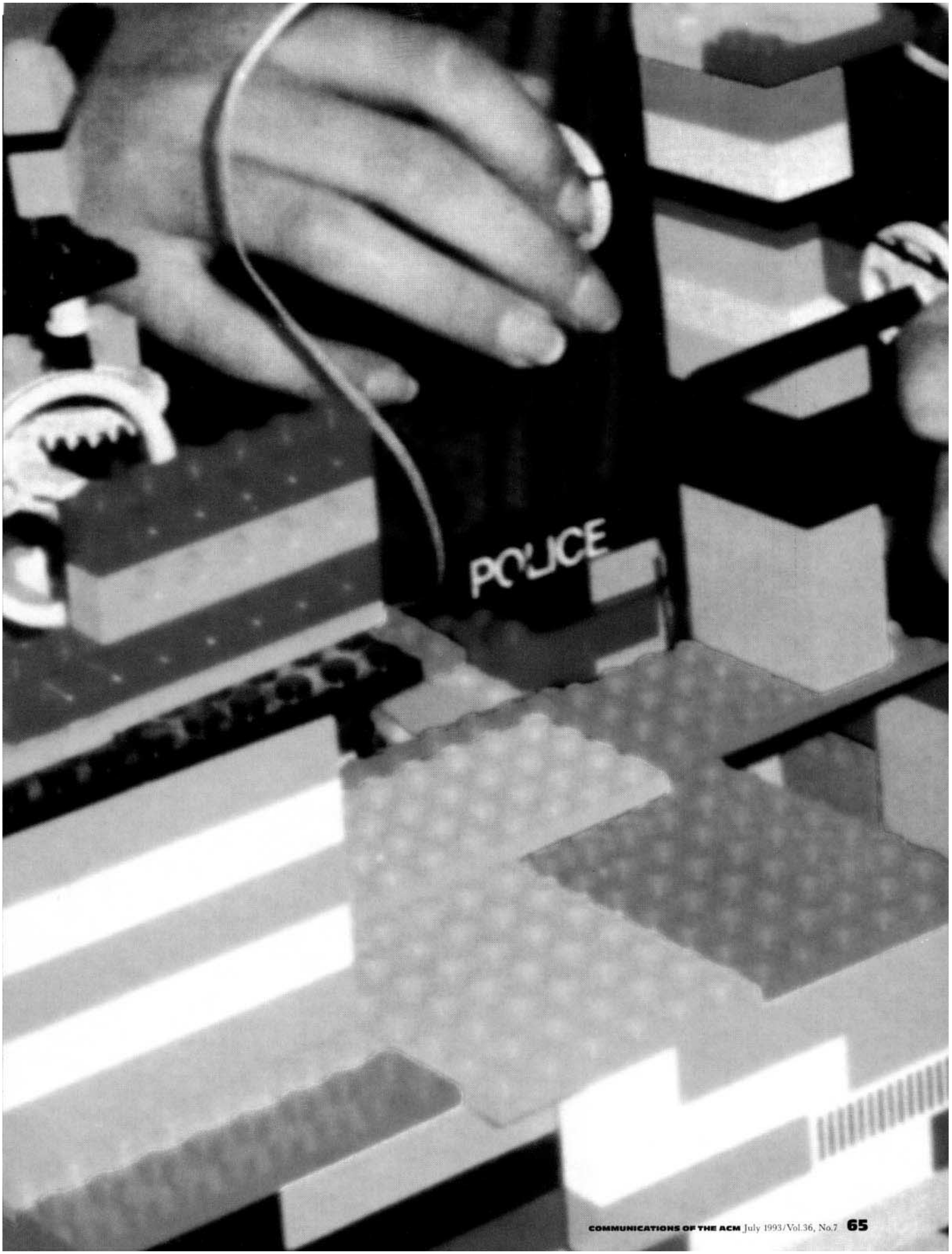
Mitchel Resnick

In the beginning, there were building blocks. Children used building blocks to construct houses and castles, towers and bridges—and, in so doing, began to play with ideas from the fields of architecture and structural engineering. Then came gears and axles, motors and battery packs. With these new parts (along with the original building blocks), children built mechanized constructions: cars and trucks, cranes and Ferris wheels. Ideas from the field of mechanical engineering entered the world of children.

During the past few years, a group of us[1] has been working on the development of a new generation of construction kits. Whereas first-generation construction kits allowed children to build *structures*, and second-generation kits allowed them to build *mechanisms*, our third-generation kits allow children to build *behaviors*. Children (and adults too!) have used our new construction kits to build a wide assortment of "behaving machines," such as a robotic creature that "wants" to move toward the light, a hamster cage that keeps track of the movements of its occupant, and a kinetic sculpture that responds to the movement of a person—or the movements of another machine.

Such projects are possible because our new construction kits include computation in the bin of building parts. Children can not only embed computation in the machines they build, they can also spread computation throughout their world, in the spirit of ubiquitous computing. Put a sensor and a few "electronic bricks" on the side of the door, and suddenly the door can "talk," issuing a greeting to each person who enters the room. Add another few bricks, and the door keeps track of the total number of people who enter each day.




POLICE

By working on projects such as these, children develop very new ways of thinking about computation, programming, and control. No longer do they see computers as boxes that sit on desktops, controlling images on video monitors. No longer do they see programming as something for experts only. Children play the roles of computer scientists and electrical engineers—and of psychologists too. By constructing machines with behavior, children develop new images not only of machines and computers, but of themselves.

In this article, I discuss several behavior construction kits developed by our group. The kits are in different stages of development. One kit (called LEGO/Logo) has already become a commercial product and has been used by more than a million children in schools in the U.S. Other kits are early prototypes, just starting to be used in a few research settings. In all cases, I discuss how kits like these can change the ways children think and learn.

LEGO/Logo

Our work on behavior construction kits began with a project known as LEGO/Logo (see Photo 1).[2] LEGO/Logo links the popular LEGO construction kit with the Logo programming language. In using LEGO/Logo, children start by building machines out of LEGO pieces, using not only the traditional LEGO building bricks but newer pieces such as gears, motors, and sensors. Then they connect their machines to a computer and write computer programs (using a modified version of Logo) to control the machines. For example, a child might build a LEGO house with lights and program the lights to turn on and off at particular times. Then, the child might build a garage and program the garage door to open whenever a car approaches.

Logo itself was developed in the late 1960s as a programming language for children [6, 12]. In the early years, the most popular use of Logo involved a "floor turtle," a simple mechanical robot connected to the computer by a long "umbilical cord" (inspired, in part, by early cybernetics research [19]). Logo included commands such as `forward`, `back`, `left`, and `right` to control the floor turtle. For example, a child could type `forward 50` to make the turtle move forward by 50 "turtle steps" or `right 90` to make the turtle turn right through 90°. The turtle makes possible a new approach to thinking about geometry, contrasting sharply with the Euclidean methods traditionally taught in the classroom [1]. This new "turtle geometry" has proved to be much more intuitive for children. The turtle connects to children's experiences in the world—children can "play turtle," imagining themselves as the turtle. As a result, the turtle has helped many children form a new relationship with mathematical ideas.

With the proliferation of personal computers in the late 1970s, the Logo community shifted its focus to "screen turtles." Children still use commands such as `forward` and `right`, but these commands control graphic images of turtles on the computer screen, not actual mechanical robots. Screen turtles are much faster and more accurate than floor turtles, and thus allow children to create and investigate more complex geometric effects. About one-third of all elementary schools in the U.S. have used Logo in their classrooms.

In some ways, LEGO/Logo might seem like a throwback to the past, since it brings the turtle off the screen and back into the world. But LEGO/Logo differs from the early Logo floor turtles in several important ways. First of all, LEGO/Logo users are not given ready-made mechanical objects; they build their own machines before programming them. Second, children are not restricted to turtles. Elementary school students have used LEGO/Logo to build and program a wide assortment of creative machines, including a programmable pop-up toaster, a "chocolate-carob factory" (inspired by the Willy Wonka children's stories), and a machine that sorts LEGO bricks according to their lengths. The LEGO company now sells a commercial version of LEGO/Logo. It is used in more than 7,000 elementary and middle schools in the U.S.

LEGO/Logo includes new types of LEGO blocks (e.g., lights and sensors) for building machines, and new types of "Logo blocks" for building programs. The language includes new commands like `on` and `off` for controlling LEGO motors and lights, and new "reporter procedures" like `sensor?` for getting information from LEGO sensors. Just as students can build increasingly complex structures and machines by snapping together LEGO bricks, they can build increasingly complex computer programs by "snapping together" Logo commands. Imagine, for example, a LEGO car with a touch sensor on the front. A student can write a Logo program called `go-until-bump` that turns the car motor on, waits until the car bumps into something, then turns the car motor off. The program would look like this:

```
to go-until-bump
on
waituntil [sensor?]
off
end
```

Probably the best way to capture the spirit of LEGO/Logo is to focus on a few specific projects. The following are brief descriptions of two LEGO/Logo projects: one project by a fifth-grade student, and one project by a teacher at a summer workshop [14].

The Alarm-Clock Bed

John, a fifth-grader at the Hennigan Elementary School in Boston, had an alarm clock next to his bed at home. But the alarm clock was not very effective. Often, when the alarm went off, John simply shut off the alarm and went back to sleep. John was determined to invent a better solution. His goal: to design an alarm clock that could not be ignored. John started by playing with the LEGO optosensor. He placed the optosensor by the window, so the computer

[1]The primary working group has included Seymour Papert, Stephen Ocko, Fred Martin, Randy Sargent, Edith Ackermann, and me at the MIT Media Laboratory, and Brian Silverman at Logo Computer Systems Inc. In much of this work, we have collaborated with the LEGO company, particularly with Lars Bo Jensen, Robert Rasmussen, and Allan Toft.

[2]LEGO/Logo was developed primarily by Stephen Ocko, Seymour Papert, Brian Silverman, and myself, in collaboration with the LEGO company [17, 18].

could "know" when the sun came up. But what should happen at sunrise? John had an idea. He built a small LEGO bed, with a small LEGO person on top. Underneath the bed, he placed a hinged platform, so the bed could tilt from side to side. Alongside the bed, he built a conveyor belt. Then he wrote a Logo program. When the optosensor detected light coming through the window, the program turned on two motors. One motor made the LEGO bed tilt to the side, making the LEGO person slide off onto the conveyor belt. The other motor turned the conveyor belt, carrying the LEGO person out the door. Would John want a full-size version of his alarm-clock ejection bed for his home? Not really, he said. But he certainly enjoyed watching the little LEGO person fly out the door.

The "Smart" Hamster Cage

For a long time, Julie Fine had wondered: What did her pet hamster do at night? Did it sleep? Or did it spend the night running on its exercise wheel? So when Julie, a teacher at the Agassiz School in Boston, came to a LEGO/Logo teacher workshop at MIT, she decided to use LEGO/Logo to study her hamster's activity. She brought her hamster to the workshop and attached a LEGO optosensor to the exercise wheel in the hamster's cage. She wrote a Logo program to keep track of how far (how many revolutions) the hamster ran every 10 minutes. The first day, the hamster slept all day. The program reported a long list of zeros. She was pretty frustrated. But when she came back to the lab the next morning, the computer showed that there had been lots of activity overnight. Her hamster ran more than 1,000 revolutions between 9:00 PM and midnight. She did a quick calculation and found that the hamster had run nearly a quarter mile! After

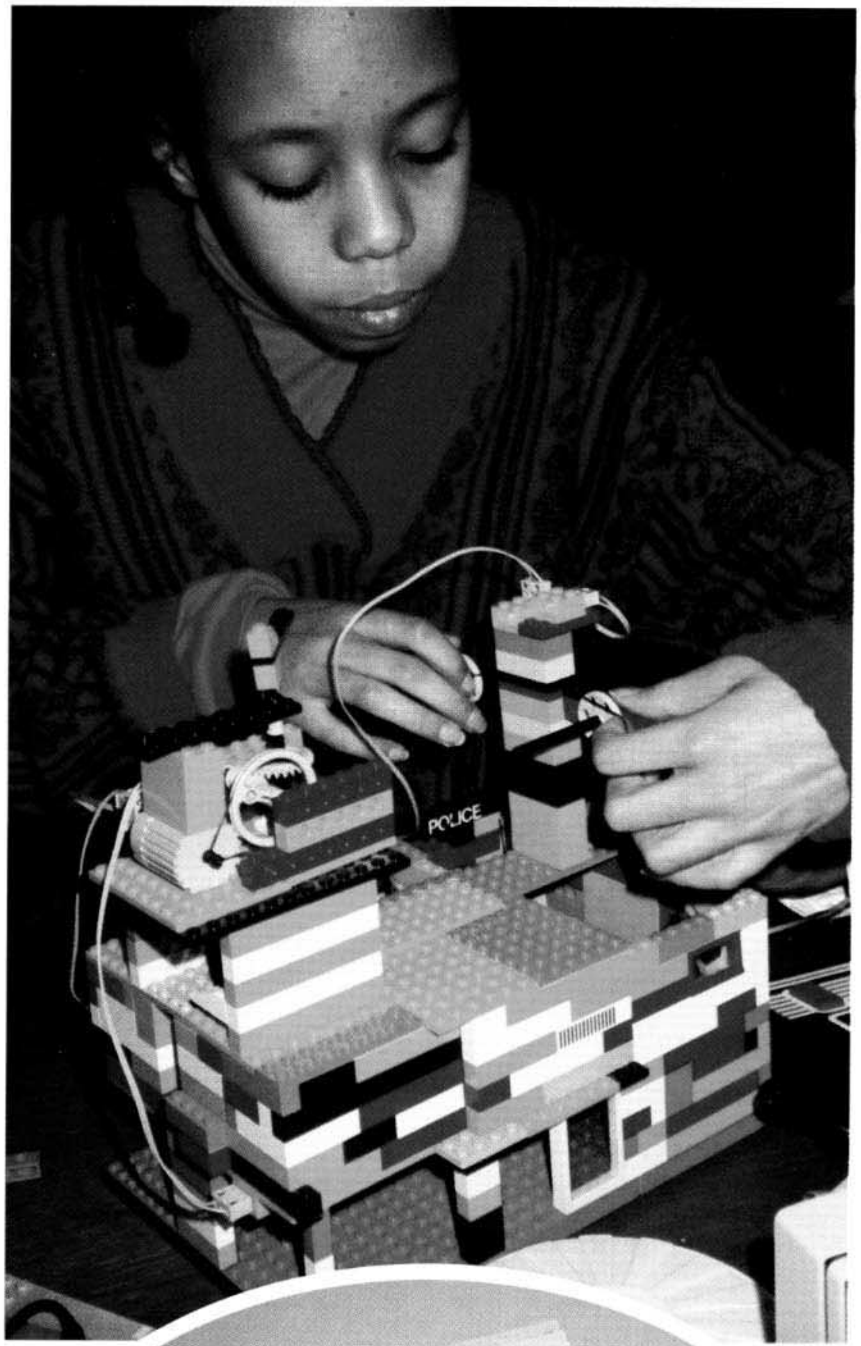


Photo 1.
Behavior construction kit

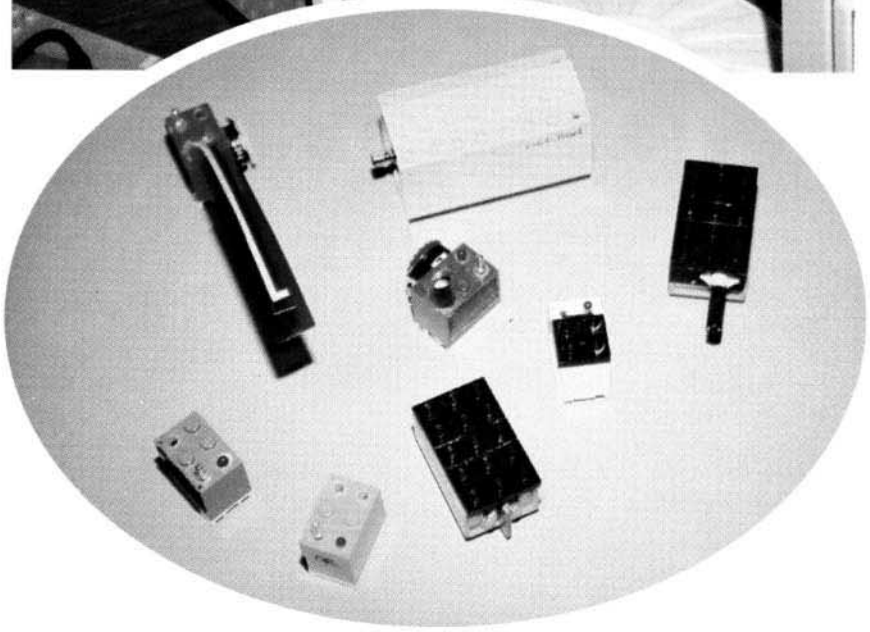


Photo 2. Electronic Bricks

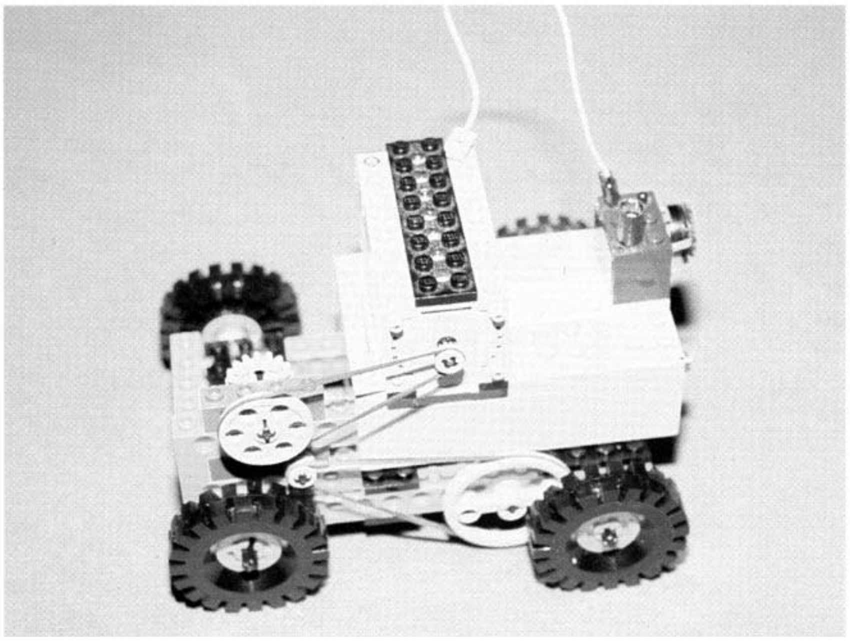


Photo 3. Motor Brick

midnight, the hamster apparently went back to sleep. Julie continued to monitor the hamster's activity for the next week. The hamster ran exclusively at night, usually in two-hour bursts of activity. But her results were not entirely reliable: on several occasions, the hamster gnawed through the LEGO wires.

Electronic Bricks

LEGO/Logo is a rich construction environment, but it does have some limitations. One problem is the sequential nature of Logo. It is very natural for children to want to control several LEGO machines at the same time or several parts of the same machine at the same time. But since Logo (like most programming languages) forces programmers to describe actions sequentially, it is very difficult to control several things at the same time with LEGO/Logo.

Another problem is that LEGO/Logo machines must be connected to a desktop computer with wires. Wires are a practical nuisance, particularly when children use LEGO/Logo to create mobile "creatures." Wires get tangled with other objects in the environment; they get twisted in knots as the creature rotates; and they restrict the overall range of the creature. Wires are also a conceptual nuisance. It is difficult to think of a LEGO/Logo machine as an autonomous creature as long as it is attached by a cord to a computer.

We tried to solve these problems in several ways. We wrote a multiprocessing version of Logo [15], so that users could control multiple machines more easily. And we experimented with various technologies for wireless communication, to get around the problem of wires. But none of these approaches satisfied us. So we decided to make a more serious modification: we began to build electronics *inside* the LEGO bricks. With these "Electronic Bricks" (developed primarily by Fred Martin, a graduate student at the MIT Media Lab), students no longer need to connect their LEGO constructions to a personal computer. Rather, they can build computational power directly into their LEGO machines and creatures. No more umbilical cord. And it becomes easy to control several machines at the same time: just build computational power into each.

We have created about a dozen types of Electronic Bricks (Photo 2), falling into three categories. There are Action Bricks (motors, lights), Sensor Bricks (light sensors, sound sensors), Logic Bricks (and-gates, flip-flops, timers). To create different behaviors, you simply wire Electronic Bricks together in different ways. Electronic Bricks are somewhat reminiscent of *Rocky's Boots* and *Robot Odyssey,* two early educational software products (sadly, no longer sold) that enabled children to construct behaviors from logic gates. But in *Rocky's Boots* and *Robot Odyssey,* all of the wires and parts were on the computer screen. With Electronic Bricks, everything is in the real world.

Children have used Electronic Bricks primarily for controlling robotic "creatures," much in the spirit of Valentino Braitenberg's *Vehicles* [3]. Even simple combinations of Electronic Bricks can lead to creatures with surprising behaviors. Consider a simple LEGO creature with a Light Sensor Brick (pointing upward) and a Motor Brick (Photo 3). The output from the Light Sensor Brick is connected to the Direction input of the Motor Brick (Figure 1). When the creature "sees" light, it moves in one direction; when it is in the dark, it moves in the other direction. Sounds simple enough—but the creature has a somewhat surprising behavior. When the creature enters a shadow, it changes directions—which causes the creature to leave the shadow and change directions again. So the creature will get stuck on the edge of a shadow, oscillating back and forth indecisively. We dubbed this creature Indecisive.

It is easy to create other behaviors. If you switch the wire to the Speed input of the Motor Brick, the creature becomes Wary: it suddenly stops whenever it enters the dark (Figure 2). If you then replace the Light Sensor with a Sound Sensor and a Flip-Flop Brick, the creature becomes Obedient: it stops moving when you clap your hands, then starts again when you clap again (Figure 3).

Students and teachers have worked with Electronic Bricks in several ways. Some have observed prebuilt creatures, trying to figure out how the creatures work—much as ethologists observe animals [5]. Others have built creatures of their own, trying to create new behaviors [2, 10]. In our research, we have found that Electronic Bricks enable young children to explore certain domains of knowledge that were previously inaccessible. The concept of feedback, for example, is typically not taught until college engineering courses. But by building (and playing) with Electronic Bricks, children can gain a meaningful understanding of feedback as early as elementary school.

We are particularly interested in how children think about the artificial creatures they build [16]. Do they see them more as machines or as creatures? To what extent do they attribute intentionality to the creatures/machines? It seems that people tend to view creatures on many different *levels.* Sometimes they view the creatures on a *mechanistic* level, examining how one LEGO piece makes another move. Then, they might shift to the *information* level, exploring how information flows from one Electronic Brick to another. At other times, people view the creatures on a *psychological* level, attributing intentionality or personality to the creatures. One creature "wants" to get to the light. Another creature "likes" the dark. A third is "scared" of loud noises.

Sometimes, children will shift rapidly between levels of description. Consider, for example, the comments of Sara, a fifth-grader. Sara was considering whether her creature would sound a signal when its touch sensor was pushed:

It depends on whether the machine wants to tell . . . if we *want the machine to tell us . . . if we* tell *the machine to tell us.*

Within a span of 10 seconds, Sara described the situation in three different ways. First she viewed the machine on a psychological level, focusing on what the machine "wants." Then she shifted intentionality to the programmer and viewed the programmer on a psychological level. Finally, she shifted to a mechanistic explanation, in which the programmer explicitly told the machine what to do.

Which is the correct level? That is a natural, but misleading, question. Complex systems can be meaningfully described at many different levels. Which level is "best" depends on the context: on what you already understand and on what you hope to learn. In certain situations, for certain questions, the mechanistic level is the best. In other situations, for other questions, the psychological

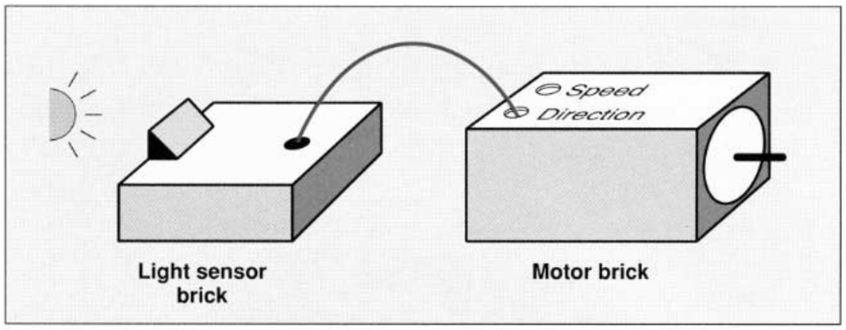


Figure 1. Indecisive

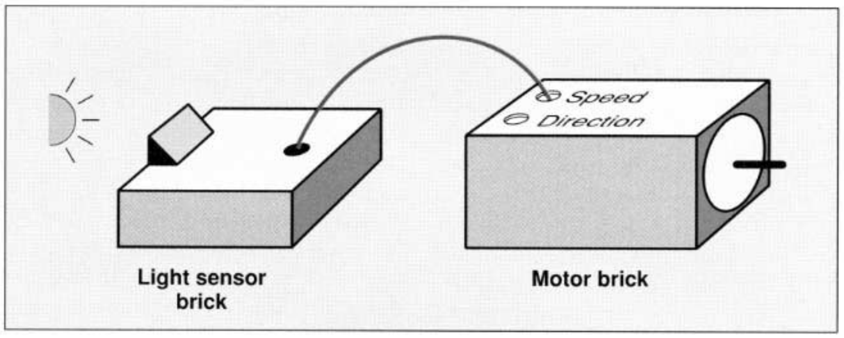


Figure 2. Wary

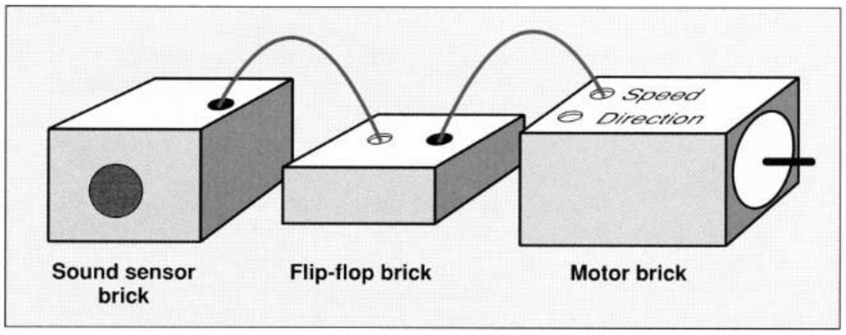


Figure 3. Obedient

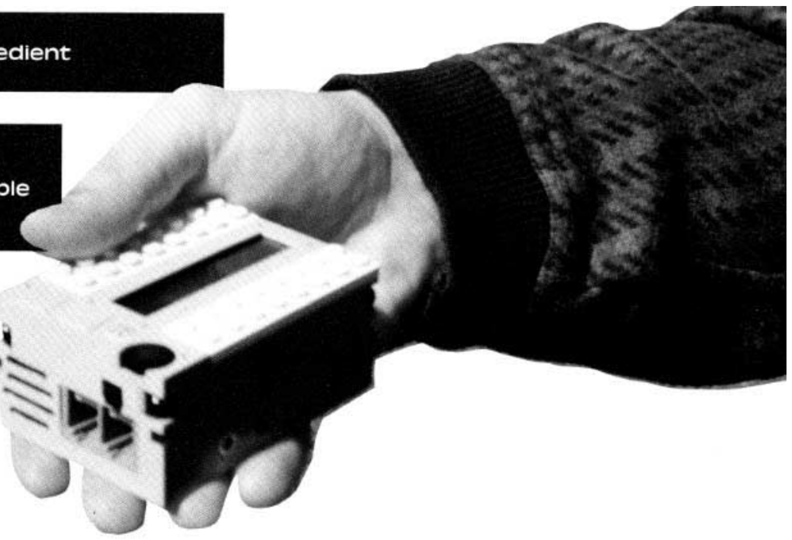


Photo 4. Programmable Brick

With Programmable Bricks, children can spread computation throughout their world, making familiar objects responsive and interactive. We view Programmable Bricks as a way of making ubiquitous computing accessible to children.

level is best. By playing with Electronic Bricks, students can learn to shift between levels, learning which levels are best for which situations.

Programmable Bricks

All of the Electronic Bricks discussed in the previous section had dedicated functions. The Flip-Flop Brick, for instance, has a very specialized function: it holds one bit of state, and it changes that state whenever it receives a sharp transition in its input. But why should we be restricted to dedicated bricks? Why not put a full computer in a LEGO brick?

Indeed, that is precisely what we are doing in our "Programmable Brick" project (Photo 4). The Programmable Brick (designed primarily by Randy Sargent, a graduate student at the MIT Media Lab) looks like a large LEGO brick, about the size of a deck of cards. (To be precise, it is 10 LEGO units long and 7 LEGO units wide.) In some ways, the Programmable Brick is like a handheld computer. It is based on a Motorola 68332 processor, with 256K of nonvolatile RAM, and a two-line liquid-crystal display on top.

But unlike most handheld computers, the Programmable Brick is specially designed for interacting with the world. It can control four motors or lights at a time, and it can receive inputs from eight sensors (eight bits of resolution each). A microphone and speaker are built into the brick for sampling and emitting sounds (at 20KHz). Around the sides of the brick are six infrared transmitters and receivers, so the brick can communicate with other Programmable Bricks (and other electronic devices). To program the Programmable Brick, you download programs from a personal computer. Then you can disconnect the brick and take it with you.

Programmable Bricks offer a new and more powerful means for controlling robotic creatures. In fact, an early version of the Programmable Brick has already been used in robot design classes and competitions at MIT [9]. But we envision many non-robot applications as well. With Programmable Bricks, children can spread computation throughout their world, making familiar objects responsive and interactive. We view Programmable Bricks as a way of making ubiquitous computing accessible to children.

Our work with Programmable Bricks is just beginning. We are just starting to introduce Programmable Bricks to children in a few schools and museums. But it is not too early to start speculating. More than 20 years ago, as researchers and educators were just beginning to explore the possibilities of computers in education, Seymour Papert and Cynthia Solomon wrote a memo called "Twenty Things to do with a Computer" [13]. The memo described a wonderful collection of activities, pushing computers in directions that few other people had imagined. Some of the activities on their list eventually became commonplace; others are still visionary today. A few years later, Danny Hillis (then an undergraduate at MIT) wrote a memo entitled "Ten Things to Do with a Better Computer" [8], describing a new set of activities that would be possible if computers could execute instructions in parallel. Hillis later realized some of these ideas in his massively parallel Connection Machine computer [7].

In the same spirit, Randy Sargent and I have compiled a new list entitled "Twenty Things to Do with a Programmable Brick":

1. Create a "haunted house." Attach a Programmable Brick to the door to make creaking sounds whenever the door is opened. Program another Brick to drop spiders on people when they walk through the door. Build a LEGO platform for a pumpkin, and program a Brick to drive the pumpkin around the room.
2. Connect sensors to various parts of your body. Then program a Programmable Brick to monitor your heartbeat and breathing as you walk and run. Or: program the Brick to play different sounds when you move different parts of your body.
3. Take a Programmable Brick with you to measure the pH level of the water in local streams or the noise levels at a local construction site.
4. Create LEGO musical instruments. The instrument might have buttons like a flute, or a sliding part like a trombone, or a completely new interface that you invent. Start by writing a simple program so the Programmable Brick plays different notes (or melodies) when you move different parts of the instrument. Then enhance the program so the Brick improvises on your notes. Or program the Brick to play "rounds" (by playing a second copy of your notes with a delay).
5. Put a Programmable Brick and light sensor on the door to keep track of the number of people that enter the room. Then program the Brick to greet people as they enter the room (with music or digitized speech).
6. Set up a weather station on the roof of the building.
7. Use a Programmable Brick to find out if the light really does go off when you shut the refrigerator door.
8. Attach a Programmable Brick to an ashtray, and program it to play a coughing sound whenever anyone uses the ashtray.
9. Build a remote-control LEGO car. Use a standard television remote control to communicate (via infrared) with a Programmable Brick in the car.
10. Create an "intelligent room" that automatically turns on the lights when someone walks in the room. (Here is one approach. Build a LEGO machine that turns on the light switch, and connect it to a Programmable Brick. Use another Programmable Brick to detect when

anyone enters the room. Use infrared to communicate between the two Bricks.)

11. Use a Programmable Brick to control a videocamera (via infrared). Program the Brick to make a time-lapse video of a plant growing (taking a few frames every hour or day).

12. Use a Programmable Brick to program your VCR.

13. Send secret messages across the room to someone else who also has a Programmable Brick.

14. Put a Brick on your dog's collar and collect data about your dog's behavior. How much time does your dog spend running around? Discuss whether experimenting on your dog is ethical.

15. Use a Brick to record your dog barking. Then put the Brick in a remote-control LEGO car. Play the barking sound when the LEGO car gets near a cat. How does the cat react?

16. Build a LEGO creature you can interact with. Program the creature to act in different ways when you clap once, or clap twice, or shine a light in its "eyes."

17. Build a LEGO creature that explores its environment. Program the creature to find the part of the room with the most light or the highest temperature. Next, put a plant on your LEGO creature, so that the plant will always move to the part of the room with the most light (or the highest temperature). Use other sensors to monitor the growth of the plant.

18. Build a LEGO machine that can water your plants. Then program a Brick to make the machine water the plants every few days.

19. Create a game where each player carries a Programmable Brick. Program the Bricks so they give instructions to the players, and send messages from one player to another.

20. Think up 20 more things to do with a Programmable Brick.

Constructing a Computerized Reality

On the day the great physicist Richard Feynman died, the following message was found on his office blackboard: "What I cannot create, I do not understand" [4]. What was true for Feynman is true for the rest of us. One of the best ways to gain a deeper understanding of something is to create it, to construct it, to build it [11].

This idea is important to keep in mind as more and more elements of our world become computerized. The only way people will understand—and feel comfortable with—this emerging computerized reality is if they can participate in constructing it, modifying it, and extending it. Computerized reality should not be built only by experts, with everyone else merely interacting with it. Rather, we need to continue developing new types of construction kits, so everyone can participate in the construction of new computerized realities. C

References

1. Abelson, H. and diSessa, A. *Turtle Geometry: The Computer as a Medium for Exploring Mathematics.* MIT Press, Cambridge, Mass., 1980.
2. Bourgoin, M. Children using LEGO robots to explore dynamics. In *Constructionist Learning.* MIT Media Laboratory, Cambridge, Mass., 1990.
3. Braitenberg, V. *Vehicles.* MIT Press, Cambridge, Mass., 1984.
4. Gleick, J. *Genius: The Life and Science of Richard Feynman.* Pantheon, New York, 1992.
5. Granott, N. Puzzled minds and weird creatures: Spontaneous inquiry and phases in knowledge construction. In *Constructionism.* Ablex, Norwood, N.J., 1991.
6. Harvey, B. *Computer Science Logo Style.* MIT Press, Cambridge, Mass., 1985.
7. Hillis, W.D. *The Connection Machine.* MIT Press, Cambridge, Mass., 1985.
8. Hillis, W.D. Ten things to do with a better computer. Unpublished memo, MIT Artificial Intelligence Lab, Cambridge, Mass., 1975.
9. Martin, F. Building robots to learn design and engineering. In *Proceedings of the Frontiers in Education Conference* (Nashville, Tenn.), 1992.
10. Martin, F. Children, cybernetics, and programmable turtles. Master's Thesis, MIT Media Laboratory, Cambridge, Mass., 1988.
11. Papert, S. Situation constructionism. In *Constructionism.* Ablex, Norwood, N.J., 1991.
12. Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas.* Basic Books, New York, 1980.
13. Papert, S. and Solomon, C. Twenty things to do with a computer. Artificial Intelligence Memo 248, MIT AI Laboratory, Cambridge, Mass., 1971.
14. Resnick, M. Xylophones, hamsters, and fireworks: The role of diversity in constructionist activities. In *Constructionism.* Ablex, Norwood, N.J., 1991.
15. Resnick, M. MultiLogo: A study of children and concurrent programming. *Interactive Learn. Envir. 1,* 3 (1990), 153–170.
16. Resnick, M. LEGO, Logo, and life. In *Artificial Life.* Addison-Wesley, Reading, Mass., 1989.
17. Resnick, M. and Ocko, S. LEGO/Logo: Learning through and about design. In *Constructionism.* Ablex Publishing, Norwood, N.J., 1991.
18. Resnick, M., Ocko, S. and Papert, S. LEGO, Logo, and Design. *Children's Envir. Q. 5,* 4 (1988).
19. Walter, W.G. An imitation of life. *Sci. Am. 182,* 5 (May 1950), 42–45.

CR Categories and Subject Descriptors: J.7 [**Computer Applications**]: Computers in Other Systems; K.3 [**Computers and Education**]: Computer Uses in Education

General Terms: Experimentation

Additional Key Words and Phrases: Education

About the Author:

MITCHEL RESNICK is an assistant professor of media arts and sciences at MIT, and head of the Learning and Common Sense section of the MIT Media Laboratory. His current research focuses on how new technological tools can support and encourage new ways of thinking. He is particularly interested in how people think about self-organizing systems and emergent phenomena. **Author's Present Address:** MIT Media Laboratory, 20 Ames Street, Cambridge, MA 02139; email: mres@media.mit.edu