Pergamon

Journal of Mathematical Behavior
19 (2000) 9–47

Mathematical Behavior

From exercises and tasks to problems and projects Unique contributions of computers to innovative mathematics education☆

Douglas H. Clements*

Department of Learning and Instruction, State University of New York at Buffalo, 593 Baldy Hall, Buffalo, NY 14260, USA

Abstract

Those interested in using computers in mathematics education stand at a crossroads. Will computers be used to reinforce existing educational practices or can they catalyze creative innovations? Educators face these turning points with directions from classroom-based research. In this article, this research corpus is reviewed. Implications are drawn from it, centering on the need to move computer use in mathematics education from the domains of exercises and tasks to engagement with problems and projects. Unique contributions of computers to problem and project-oriented pedagogical approaches are described and unique challenges that must be faced when implementing these approaches discussed. © 2000 Elsevier Science Inc. All rights reserved.

Keywords: Mathematics; Software; Computers; Educational reform

Those interested in using computers in mathematics education stand at a crossroads. Will computers be used to reinforce existing educational practices or can they catalyze creative innovations, following NCTM standards (National Council of Teachers of Mathematics (NCTM), 2000)? Fortunately, educators face these turning points with directions from

☆ An early version of this paper was presented as a general session at the 72nd Annual Meeting of the National Council of Teachers of Mathematics, Indianapolis, IN, April 14, 1994. Time to prepare this material was partially provided by National Science Foundation Research Grant NSF MDR-8954664, "An Investigation of the Development of Elementary Children's Geometric Thinking in Computer and Noncomputer Environments." Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

* Corresponding author. Tel.: +1-716-689-3788; fax: +1-716-645-3161.
E-mail address: clements@acsu.buffalo.edu (D.H. Clements)

0732-3123/00/$ – see front matter © 2000 Elsevier Science Inc. All rights reserved.
PII: S0732-3123(00)00036-5

classroom-based research. To begin, we might look at what students in mathematics classrooms are doing now with computers.

1. Students using computers: the picture today

Picture students using computers in their classrooms. What are they doing? Research indicates that they are using computers only occasionally, and usually only to provide "variety," "rewards," "enrichment," or "something for students to do" (Becker, 1990). In one study, 12% of teachers reported that *only* those students "who have finished their seat work" actually got to use the computer. Often, slower students never get to use the computer (Hickey, 1993). In the elementary grades, they use mostly drill-and-practice software; their teachers state that their goal for using computers is to increase basic skills rather than solve problems. At the junior and high school levels, students spend at least one-half of their time on computers learning computer-specific skills. They spend less than one-quarter of their time working productively in all academic subjects combined. A minority of their mathematics teachers use computers for instruction. Thus, students spend much of their present-day computer time with drill-and-practice exercises and tasks—structured work assigned or done as part of one's classroom duties.

Computer Contributions

Standard		Drill
1.	Mathematics as Problem-solving	°
2.	Mathematics as Communication	°
3.	Mathematics as Reasoning	°
4.	Mathematical Connections	-
5+	Achievement (content)	❍
Beliefs about mathematics		-
Affect, motivation, autonomy		°
Social construction of mathematics		°
Mathematical creativity		-

Key: Level of contribution

-	little or negative
°	small
❍	medium
●	large

Fig. 1. Computer contributions: drill and tutorial software. Although I drew from research in creating this chart, there was also considerable personal interpretation.

When students use such computer drills or tutorials consistently, they do lead to moderate but statistically significant learning gains, especially in mathematics (Clements & Nastasi, 1992; McCoy, 1996; Roblyer, Castine, & King, 1988). This does not mean, of course, that using any software in any way guarantees such gains.

Also, research can't tell you *what your goals are*. Use of only drill-and-practice or tutorial software is inconsistent with NCTM's standards. While such software can develop skills in various topics, it is a weak contributor to the first four curriculum standards and other NCTM goals (Fig. 1).

Unfortunately, from this point of view, many schools are investing heavily in highly structured "integrated learning systems" (ILS's). These systems automatically load one of an extensive sequence of lessons into each student's computer. Evaluations of these systems also show a moderate effect on basic skills (Becker, 1992; Kelman, 1990). We must, however, question other aspects of ILS's, especially diminished teacher and student control. In too many cases, ILS's represent a triumph of bureaucratic efficiency over students' full mathematical development.

The bottom line is this: *What we as a mathematics education community are doing the most is what research and the NCTM's standards say that we should be doing the least* (not none, necessarily, just the least).

2. Paths to promising problems

Although exercises and tasks predominate in the software students experience most frequently, many teachers say that computers *should* be used differently. In the last few years, the proportion of teachers advocating using computers from the "tool" perspective has increased, though they are still in the minority (and they more often teach language arts and English education, not mathematics) (Becker, 1991).

These teachers want to move away from drill exercises and tutorial tasks. They, along with other teachers and researchers (Holt, 1982; Scardamalia, Bereiter, & Lamon, 1994), find that students do develop and use strategies, but that these strategies are adapted to a different goal. Students use "procedure copying" and "knowledge telling" (just say everything you know) strategies. They "do well," but do not become *engaged* with the subject matter. We all know students who are not lazy, but are coping with a long line of tasks. They are rewarded for completing these tasks with opportunities to do more enjoyable activities—sometimes, a computer game. They develop strategies for efficiently completing these tasks; they see that reflecting on and extending ideas do not help them.

But what *do* we as teachers do beyond exercises and tasks? We can move to problems and projects. Both are difficult to do well. Both can be easier to do well with computers.

2.1. Good mathematical problems

Let's look at problems first. What are the characteristics of investigations that can lead to good mathematics problems for students? Good problems (adapted from Russell, Magdalene, & Rubin, 1989; Wheatley, 1991):

- are meaningful to the students;
- stimulate curiosity about a mathematical or nonmathematical domain, not just an answer;
- engage knowledge that students already have, about mathematics or about the world, but challenges them to think harder or differently about what they know;
- encourage students to devise solutions;
- invite students to make decisions;
- lead to mathematical theories about (a) how the real world works or (b) how mathematical relationships work;
- open discussion to multiple ideas and participants; there is not a single correct response or only one thing to say;
- are amenable to continuing investigation, and generation of new problems and questions.

2.2. How computers help with a problem-centered approach

Computers, especially with their visual displays, can be a source of meaningful problems with a variety of solution strategies and solutions. Students can explore these various solution paths individually and in small groups, making decisions and receiving feedback about their ideas and strategies. Access to computers helps implement this less-structured problem-centered approach, which positively affects the number and kinds of decisions students have

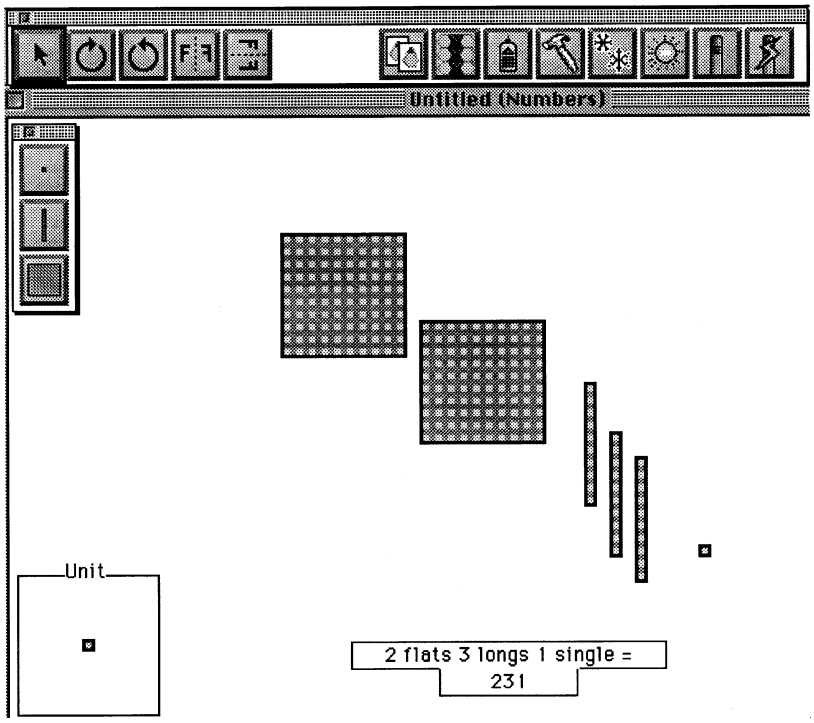


Fig. 2. A Number Blocks screen; the "odometer" at the bottom shows the symbolic representation of the number of blocks.

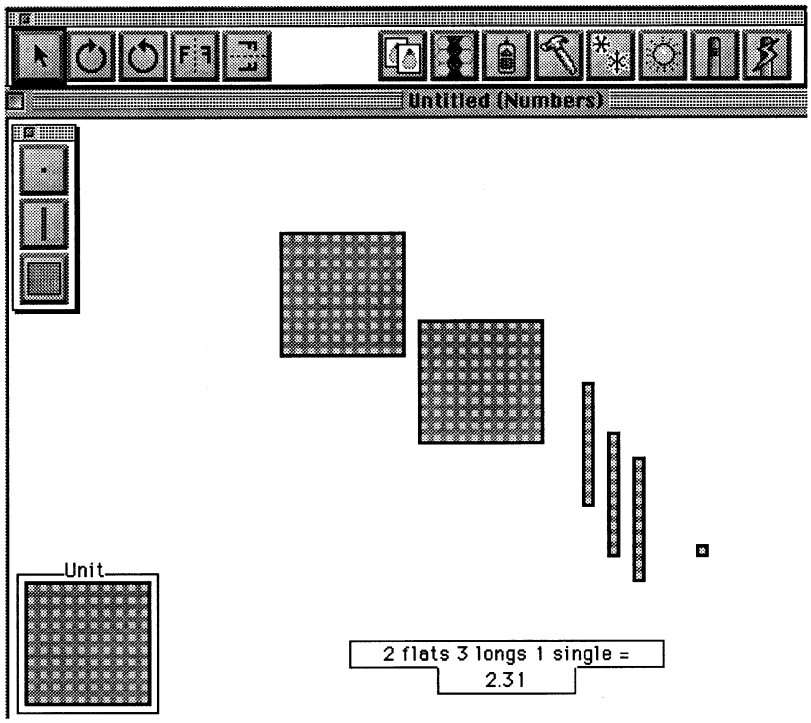


Fig. 3. Any block can be set to "one."

to make and results in more active and highly motivated learners (Fisher, 1990, 1991). The social nature of educational computing facilitates discussion about the problems (Clements & Nastasi, 1992; Nastasi, Clements, & Battista, 1990). Most straightforward, computers allow a de-emphasis on those aspects of mathematical work that can be (and, in the world outside of school, usually is) done by machines and an increased emphasis on conceptual thinking and planning (Fey, 1989; McCoy, 1996).

2.3. Building a better manipulative

One type of program is the computer manipulative. In *Shapes*[1], for example, students can choose to manipulate base-ten blocks. Students' concrete actions are immediately represented symbolically in two ways (see Fig. 2). Adding blocks to the block set changes the display so that it always reflects the number represented by the blocks. Students can change any block to be "one" (Fig. 3). Students can easily model operations (Fig. 4).

These computer blocks are not physically concrete. However, no base-ten blocks "contain" place value ideas; rather, students construct these ideas while thinking about their actions on the blocks (Clements & McMillen, 1996; Kamii, 1986). Actual base-ten blocks

[1] *Shapes*™, copyright 1997 Douglas H. Clements and Julie Sarama. All rights reserved.

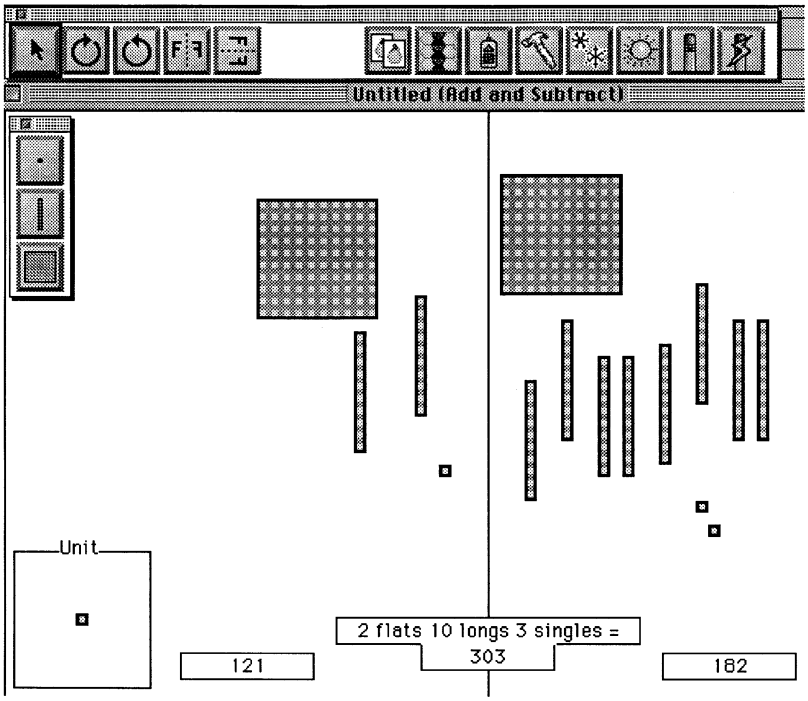
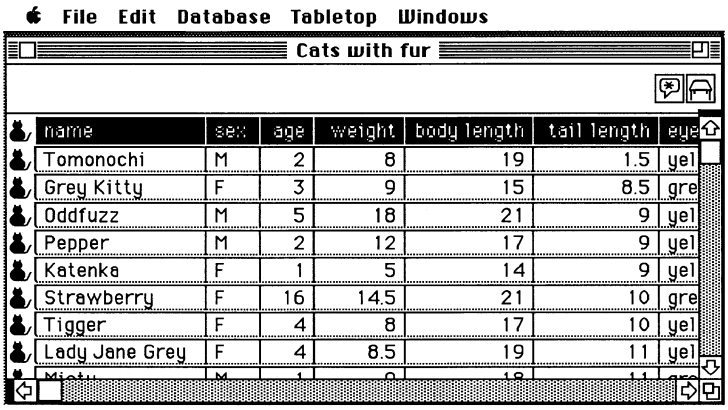


Fig. 4. Addition is facilitated by separate groups of number blocks, each with their own odometer. If the separation is erased, one odometer shows the number in the combined group. For multiplication, students create an array and then fill it with blocks.

can be so clumsy that the actions become disconnected. The computer blocks can be more *mentally* manageable (Thompson, 1992; Thompson & Thomp- son, 1990).

Research supports the notion that manipulatives—on or off computer—are concrete and meaningful if they make sense to the student (Clements & McMillen, 1996). Sidewalk



name	sex	age	weight	body length	tail length	eye
Tomonochi	M	2	8	19	1.5	yel
Grey Kitty	F	3	9	15	8.5	gre
Oddfuzz	M	5	18	21	9	yel
Pepper	M	2	12	17	9	yel
Katenka	F	1	5	14	9	yel
Strawberry	F	16	14.5	21	10	gre
Tigger	F	4	8	17	10	yel
Lady Jane Grey	F	4	8.5	19	11	yel

Fig. 5. The "cats" database in the *Tabletop* program.

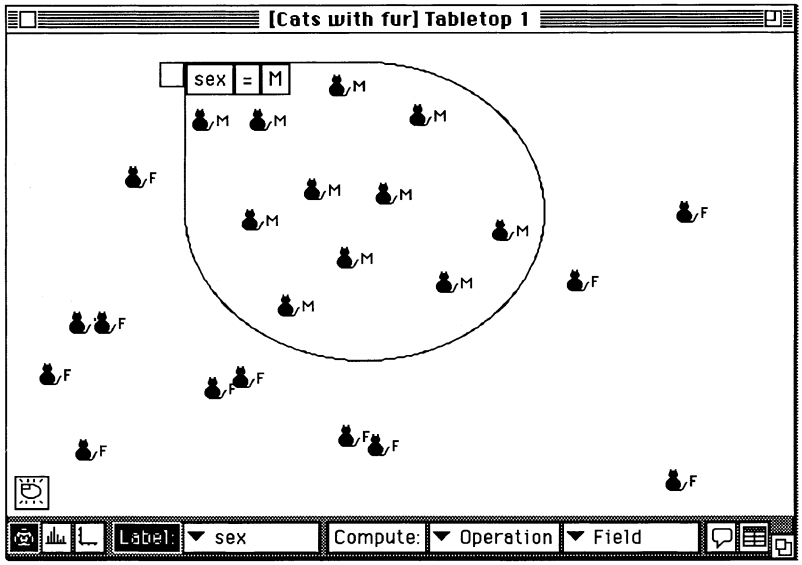


Fig. 6. After moving to the tabletop, students have selected a loop and specified that it include all those cats whose sex is male. The cats move automatically into an appropriate position.

concrete is strong because it combines many separate particles in a connected physical structure (concrete means "to grow together"). A concrete idea is strong because it combines many ideas in a connected knowledge structure—we call this "integrated-concrete" knowledge (Clements & McMillen, 1996). The computer environment helps links the blocks to the symbols, connecting these ideas and helping students develop integrated-concrete knowledge.

In addition, students can break computer base-ten blocks into ones, or glue ones together to form tens. Such actions are more in consistent with the *mental actions* that we want students to learn (Clements & McMillen, 1996).

2.4. Graphing: changing, rearranging, and connecting representations

Another aspect of the flexibility provided by many computer manipulatives is the ability to change and rearrange information. Most spreadsheet and data base programs will sort and reorder the data in different ways. Such flexibility can help open up new possibilities for exploring problems.

For example, Fig. 5 shows a "cats" database in the *Tabletop*[2] program. Students can add, change, or sort the data in tabular form. When students move to the tabletop, they can graphically represent these data in a variety of ways. For example, in a Venn diagram form, they can show male cats. The cats move dynamically to their proper place (Fig. 6). They may wish to inquire whether more female or more male cats are heavy. Students often make another

[2] *Tabletop*™, copyright 1994 TERC. All rights reserved. The cats data is from the *Used Numbers* curriculum, published by Dale Seymour.

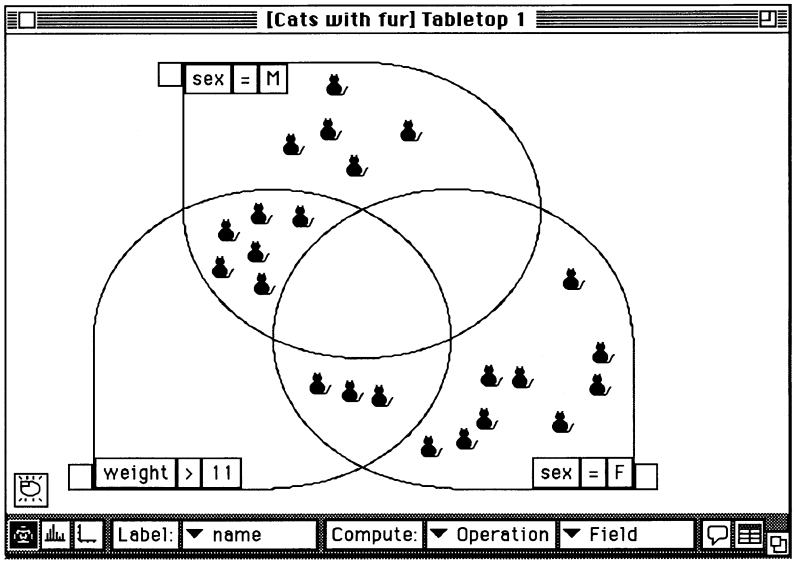


Fig. 7. Adding two more loops, students can determine whether there are more heavy male or female cats.

loop for female cats, then another for cats that weigh more than 11 pounds (Fig. 7). Again, the cats move dynamically into their correct position. Other representations are possible (Fig. 8).

Research shows that *Tabletop* offers a flexible, exploratory, approach to data analysis that encourages students to make sense of data in their own way (Hancock, Kaput, & Goldsmith,

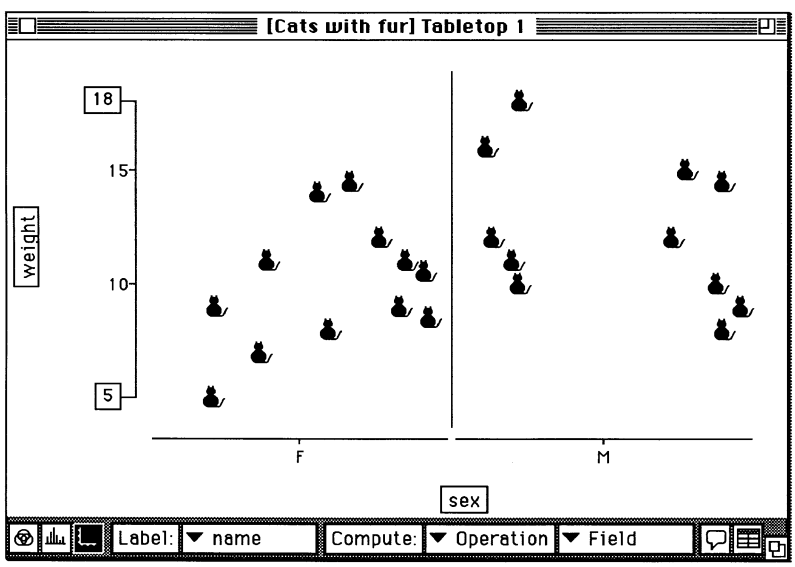


Fig. 8. Another way to view these cats is to select an "axes" representation. The cats can now be moved left and right, but not up or down, as their vertical position indicates their weight.

1992). A graph is not a result, but one stage in a continuing process of exploration and analysis. Students can construct many representations and comparisons in a short time, making qualitative differences in their educational experience. *Tabletop* extends students' range; for example, they can study large composite data sets from multiple classrooms. Finally, programs such as *Tabletop* help students connect different types of tabular and graphic representations—again helping students build integrated-concrete knowledge.

2.5. Geometric construction programs: building dynamic connections

Another class of programs that dynamically link multiple representations consists of geometric construction programs. One of the first, the *Geometric Supposer* software series (Schwartz & Yerushalmy, 1986), was designed to facilitate students making and testing conjectures. The *Supposer* programs allow students to choose a primitive shape, such as a triangle or quadrilateral (depending on the specific program), and to perform measurement operations and geometric constructions on it. The programs record the sequence of constructions and can automatically perform it again on other triangles or quadrilaterals. In one study, *Supposer* students performed as well as or better than their non-*Supposer* counterparts on geometry exams (Yerushalmy, Chazan, & Gordon, 1987). In addition, students' learning went beyond standard geometry content, for example, reinventing definitions, making conjectures, posing and solving significant problems, and devising original proofs. *Supposer* activities engendered a movement away from considering measurement evidence as proof (Chazan, 1989; Wiske & Houde, 1988), although some students still thought that there might be counterexamples to deductively proven results. Unlike textbook theorems, students believe *Supposer*-generated theorems need to be proved before they could be accepted as true, leading to ownership of the theorems. To implement successfully the

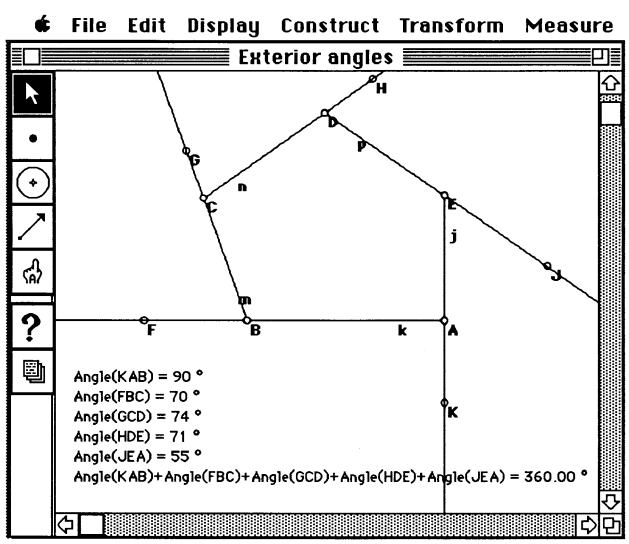


Fig. 9. A *Sketchpad* pentagon shows the measures of the exterior angles and the sum of these angles.

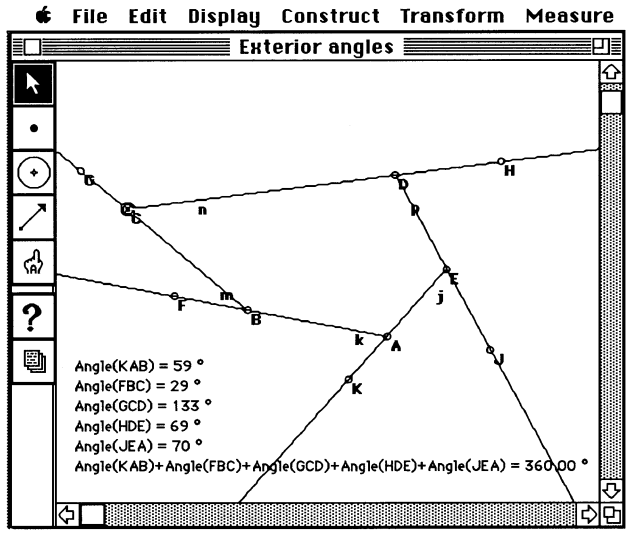


Fig. 10. The points can be moved with the mouse, and the measures are changed automatically.

Supposer's guided inquiry approach requires teaching strategies that connect students' inquiry with the curriculum and encourage inquiry as a way to learn successfully what needs to be known. While use of the *Supposer* was found to demand hard work from, and cause some frustration in, both teachers and students, benefits were evident (Lampert, 1988; Wiske & Houde, 1988; Yerushalmy et al., 1987).

Other programs emphasize dynamic geometry via dragging, including *Cabri-Geometry* (Baulac, Bellemain, & Laborde, 1988) and *Geometer's Sketchpad* (Jackiw, Klotz et al., 1991). Explorations encourage students to make conjectures and provide insight into the reasons those conjectures might be true or false. For instance, students can construct a polygon with rays, and measure the exterior angles (Fig. 9). Then they can move parts of the polygon and watch what changes dynamically (Fig. 10). Some measures change, but the sum does not. The demonstration is convincing because the size and shape of the polygon can be changed at will. Students can repeat this experiment with other polygons, and make a conjecture. To explore *why* their conjecture might be true, they can use the dilate tool to "shrink" their polygon. As it approaches a point, the sum of the exterior angles can be considered in a new light (Fig. 11).

Feedback helps students in many ways. Betty Kantrowitz, a teacher who received the Presidential Award for Excellence in Mathematics Teaching, uses a similar geometric construction program. She states, "Kids believe the machine. I can tell them a thousand times that they can't construct a particular circle because they don't have the correct properties, and they won't believe me. They'll insist that they can. But as soon as Geo-Explorer displays the message 'Can't Construct This Object,' then they realize, 'Aha, I must be missing something'."[3] This may be promising, especially if students do not just believe the machine; do they search for reasons? The answer is tentatively positive, as we see from research.

[3] From "Visualizing math with Geo-Explorer," Macintosh Resource Guide. *Geo-Explorer*™ is published by Scott Foresman.

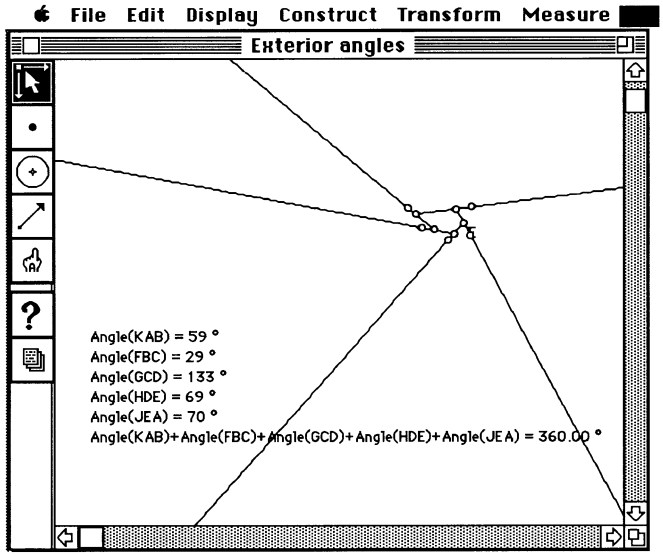


Fig. 11. Using the dilation tool to shrink the pentagon begins to reveal the relationship among the exterior angles.

Dynamic geometry software may change the nature of the relations between what Laborde (1996) calls "diagrams" (physical, spatial properties) and "theory" (geometrical properties). Students often believe it is possible to abstract the properties of geometric objects from diagrams directly and thus deduce a property empirically. Dynamic geometry software introduces a new type of diagram whose behavior is controlled by theory (Cabri tends to focus on interdependence of objects' properties). Students' actions require construction of an interpretation in which visualization plays a critical role but geometrical properties constrain such action. To encourage students to make connections between representations, teacher might ask them to interpret a diagram in geometric terms, give geometrical reasons for the behavior of the diagram, or give geometrical reasons for the simultaneity of two diagram relations while "dragging." The facility of relaxing or modifying conditions (Cabri's "redefine an object") is one means of asking students to make predictions. Research shows that use of Cabri can help students move from purely visual to geometric strategies (Laborde, 1995). For example, the drag mode disqualified the purely visual strategies and it enabled the students to notice spatial invariants. Students' belief that the software "knew geometry" motivated students to search for a geometrical solution.

Dynamic geometry programs offer a viable way to avoid beginning with proof. According to de Villiers (1995), proof is more about explanation and discovery—and sometimes systematization and communication—than about convincing. Software such as Cabri can convince people, but can they then *understand?* Students who use dynamic geometry programs in a curriculum specifically designed on the van Hiele theory do increase their levels of geometric thinking and their achievement (Choi koh, 1999; Dixon, 1997), although such positive results are not guaranteed (Roberts & Stephens, 1999).

In general, research indicates that students using such tools score higher on higher-level and application questions (McCoy, 1996). The increasing availability of dynamic geometry

programs (e.g., Cabri on TI calculators) indicates that achieving such benefits should be within the reach of many. An advantage of geometric construction programs that has been inadequately studied is storing more than static configurations. Once we finish a series of actions, it is often difficult to reflect on them. Computers, in contrast, have the power to record and replay *sequences* of our actions on manipulatives. We can record our commands and later replay, change, and view them. This encourages mathematical exploration.

2.6. Computer programming: Logo

Recording and replaying students' actions is most completely realized in computer programming. Programming in languages such as Logo also focuses on the connections between the concrete and the symbolic.

An episode in one fifth-grade class serves as an example. Students had written Logo procedures to draw rectangles, then had abstracted these to write a "general" rectangle procedure—one that takes two inputs for the lengths of opposite sides. Their teacher gave them a group of figures (Fig. 12) and asked them to decide whether they could draw each using their procedure and an initial turn if needed.

In response to the parallelogram, #7, Jonathan said, "Maybe" (he could draw it with the rectangle procedure). He estimated the initial turn, then the side lengths. After typing in his commands, he got a rectangle. He held the sheet up right next to the screen. "No."

Teacher: Could you use different inputs, or is it just impossible?
Jonathan: Maybe if you used different inputs. (Note in the next section evidence of a transition moment.)

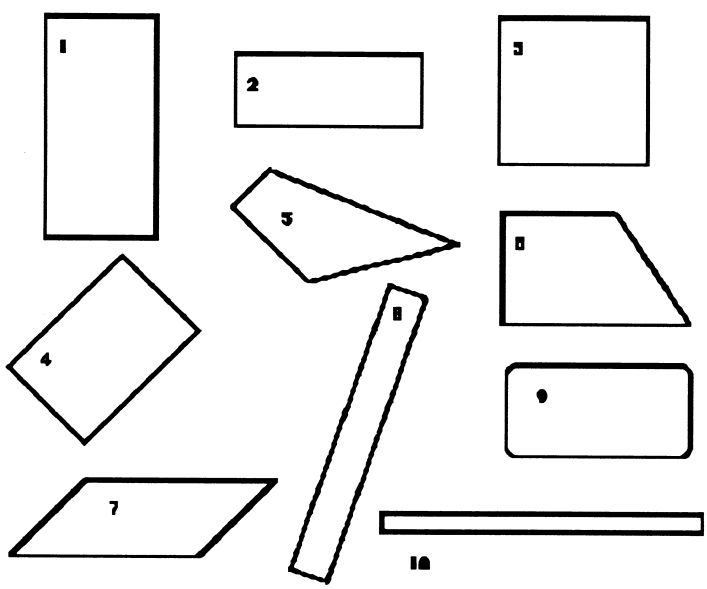


Fig. 12. Am I a rectangle? Students were asked to determine if they could draw each figure with their Logo rectangle procedure.

Jonathan: (Types in the initial turn and stares at the picture of the parallelogram; pauses.): No, you can't. (Pause.)
Jonathan: Because the lines are slanted, instead of a rectangle going like that. (traces).
Teacher: Yes, but this one's slanted (indicating #4, the oblique rectangle that Jonathan had successfully drawn).
Jonathan: Yeah, but the lines are slanted. This one's still in the size (shape) of a rectangle. This one (parallelogram) the things slanted. This thing ain't slanted. It looks slanted, but if you put it back it wouldn't be slanted. Anyway you move this, it wouldn't be a rectangle.
Jonathan: (Shaking his head): So, there's no way.

The Logo environment was important to Jonathan in figuring this out. He didn't even complete his second attempt. Rather, after making the initial turn and trying to decide on the inputs, he recognized somehow that the relationship between adjacent sides was not consonant with the implicit definition of a rectangle in the Logo procedure. The first attempt with Logo and his "running through the procedure in his head" contributed to his emerging sense of certainty (Clements & Battista, in press).

2.7. Misunderstandings of programming

When I first came to my present university, I asked the person running the educational computer lab, "What versions of Logo do you have?"

"We don't have Logo," he replied, "I just threw it out." Retaining my proper academic composure, I ask why.

"Now that we have 'Dazzle Draw,' what would you want to use Logo for?" But, of course, the point is not the drawing, it's *thinking about doing the drawing* (Clements & Battista, 1992b). A lot more thought has to go into deciding what should be "easy" and what should remain a struggle, in the positive sense of the word. Logo can be difficult, but worth it. As one third-grade boy put it, "Logo is very hard. . .but it had to be done. I liked doing it" (Carmichael, Burnett, Higginson, Moore, & Pollard, 1985).

Others believe, conversely, that Logo is for babies. In their Connected Geometry project, however, the Educational Development Center is using Logo and *The Geometer's SketchPad* to present deep and interesting problems to high school students.

Finally, some say that we don't need programming anymore. The author of another geometric construction program demanded a direct manipulation interface only. . .there was no need for programming! In later versions, however, the program included recorded actions. . .then scripts. . .then modifiable scripts. Unfortunately, the scripting language is not now as well thought out as formal computer programming languages. It is ironic that at the same time that many in computer education are downplaying programming, programming languages are showing up everywhere, in programs from word processors to the computer operating systems. The "acquisition of computer programming skills" remains important (President's Committee of Advisors on Science and Technology—Panel on Educational Technology, 1997).

So, we need programming. The alternatives are less viable. The latest version of one popular program, *SimLife,* has 350 menu options and control options and more than 200

pages of documentation. Users want to do many things, but few want to do everything that is available (Eisenberg, 1993). If, instead, students could extend an application with programming, there would be little need for all these extras. If students have a set of mathematical primitives as building blocks to investigate a variety of topics, they do not need the latest update. Eisenberg states that the development of complexity within the application should grow from the students' expanding ideas, realized through programming, rather than from often profit-driven additions of features.

When you program, you are constructing mathematical processes and objects. That programming uses a formal language should be welcomed, not eschewed. It offers generality and power. This is why Bulgarian educators Filimonov, Kreith, and Sendov (Filimonov & Kreith, 1990; Filimonov & Sendov, 1990) chose to use an enhanced Logo to study secondary geometry. The tool is, in their eyes, more powerful, extensible, and creative. Students can build primitives (e.g., for the area of any quadrilateral) that are often "black boxes" in construction programs. They can also ask questions (e.g., will the same phenomenon be true of pentagons or octagons) that can not be raised within more constrained construction programs. Students build a toolbox of algorithms and geometric knowledge simultaneously.

The educators argue that the use of a language is central. "At the heart of this system is the philosophy that in order to do mathematics students must have LANGUAGE to express their mathematical ideas and that the notion of DEFINITION is so central to mathematics that it cannot be ignored in mathematical education" (Filimonov & Sendov, 1990, p. 1). Also, "the most important and fundamental mathematical activity is dealing with notions—mainly composing and decomposing of notions—which definitely needs a language" (Filimonov & Sendov, 1990, p. 1). When mathematical objects and processes are so described, they can be saved, studied, revised, generalized, utilized, and communicated. For example, if you have procedures to do greatest command divisor for integers, you can just change the MOD and the QUOTIENT procedures and...it then works with polynomials.

This is not to argue against direct manipulation. In contrast, such manipulation and programming languages complement each other. This—along with the fact that programming environments themselves are developing and changing—can be illustrated with a new version of Logo.

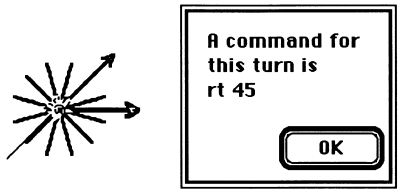


Fig. 13. *Turtle Math*'s on-screen protractor. One arrowhead shows the turtle's heading. The other follows the cursor, which students move with the mouse. When they click the mouse, this arrowhead "freezes" and the computer displays a turn command.

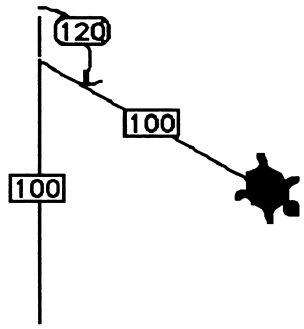


Fig. 14. When you click on *Turtle Math*'s Label Lines and Label Turns tools, the turtle shows the measures on the drawn figure.

2.8. New versions of Logo

New versions of Logo are being created. Brian Harvey's emphasizes computer science concepts, and is free. Our own effort, *Turtle Math*[4] was born from research on Logo. We believed that this research would provide useful guidelines for designing a Logo environment fine-tuned for the learning of geometry and other mathematical topics. We abstracted five principles and designed *Turtle Math* based on these principles (Clements & Sarama, 1995). For example, research supports the idea that computers can help students build mathematical knowledge out of their personal, visual knowledge, such as knowledge of walking and moving. However, it also warns that children have much difficulty with certain facets of turtle geometry work, such as measurements.

To help, *Turtle Math* (Clements & Meredith, 1994) features a turtle that turns slowly and, optionally, a ray turns with it, showing the change in heading throughout the turn. *Turtle Math* also provides measurement tools; for example, an on-screen protractor measures turns and angles (Fig. 13). In addition, students can use tools to label the lengths of all line segments and the measure of all turns (Fig. 14).

Another way *Turtle Math* supports the growth of mathematics from the visual lies in its overall structure, which helps connect the symbolic to the graphic figure. Students enter commands in "immediate mode" in a Command window (Fig. 15). This window is long enough that students can view the steps through all the commands that correspond to a figure. Any change to these commands is reflected automatically in the drawing. For example, if you change the fd 40 to fd 60, the drawing *automatically* shows that change (Fig. 16). This makes *Turtle Math* easy to use and it helps students make important connections between the mathematical commands and the turtle's drawing. One student said, "You should change those three commands [gesturing at 'rt 90 rt 20 rt 10'] to just rt 120. Because 90 + 20 + 10 is 120." A tool (the first icon on the left) copies these into the Teach window, applies a student-supplied name, and so defines the procedure.

[4] Our version of Logo exists in two versions. It is called *Geo-Logo*™, as it is used in the new curriculum, "Investigations in Number, Data, and Space," published by Dale Seymour. It is called *Turtle Math*™, a stand-alone expanded version environment with activities; *Turtle Math* is a trademark of Logo Computer Systems.

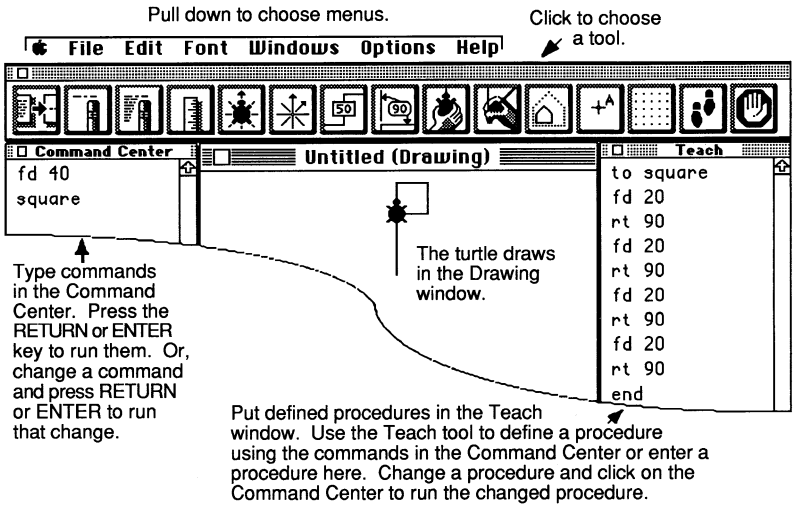


Fig. 15. *Turtle Math*'s main windows.

One limitation of traditional versions of Logo has been in the lack of *two-way* connection between visual and symbolic representations. Students create or modify symbolic code to produce visual drawings, but not the reverse. *Turtle Math* provides a "Draw Commands" tool that allows the student to use the mouse to turn and move the turtle, with corresponding Logo commands created automatically (Fig. 17).

Another tool that allows students to build a two-way connection is the Change Shape tool. An activity sheet on parallelograms that introduces students to that tool is shown in Fig. 18. A similar tool allows similarity transformations, with the commands dynamically changing; students quickly notice that the inputs to the turn commands do not change, while the inputs to the forward commands do change, and, of course, in a certain way.

Turtle Math provides other tools, to facilitate editing and reflection (the erasing tools and the step tool, which walks through a series of commands one-by-one), use of geometric motions, and work with coordinates and grids (Fig. 19).

Other new versions of Logo innovate in different ways. With Lego-Logo, students create Lego structures, including lights, sensors, motors, gears, and pulleys, as well as Logo programs that control these structures. For example, fourth-grader Kevin started, as many

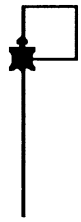


Fig. 16. Any change that is made to the commands, such as changing the fd 40 to fd 60, is automatically reflected in the drawing.

The cursor changes to a cross hairs, the turtle continually turns to face it, and the corresponding turn command is dynamically updated in the Command Center.

The turtle is then dragged forward or back and the corresponding fd or bk command is dynamically updated in the Command window.

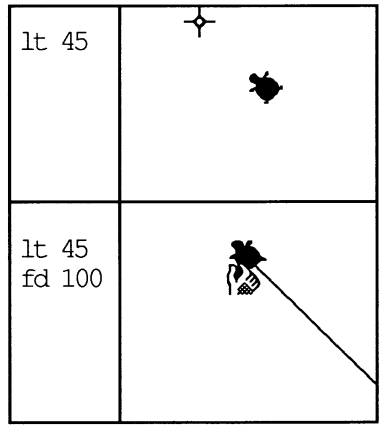


Fig. 17. *Turtle Math* provides a "Draw Commands" tool that allows students to "draw" with the mouse and automatically create corresponding Logo commands.

other students do, by building a car out of Lego (Resnick, 1988). The car moved forward a bit. . .and then the motor fell off and vibrated across the table. The movement interested Kevin. He wondered if he could use the vibrations to power the vehicle. He mounted a motor on a Lego base and learned that he could control the walker—it turned right when the motor rotated in one direction, left when it rotated in the other. Another group of children built an entire chocolate factory.

There are but a few studies on Lego-Logo, but they indicate that such experiences can positively affect mathematical achievement, especially concepts such as angle, processes such as reversibility, and competencies in higher-order thinking skills (Browning, 1991; Enkenberg, 1994; Flake, 1990; Weir, 1992), though one control group showed higher gains in computation (Flake, 1990). Lego-Logo appears to provide authentic learning tasks (Lafer & Markert, 1994), motivate and empower students as well, and possibly develop self-esteem (Silverman, 1990; Weir, 1992). This may be because Lego-Logo provides an academic setting in which students can develop their own goals.

In regular, sequential, Logo, there is a single process that runs instructions one step at a time. One of the many recent innovations in various versions of Logo is the addition of parallel programming. Also called "concurrent" or "multiprocessing," this feature allows programmers to control multiple, interacting processes. In LCSI's Microworlds, for example, students might change the turtle's shape to a bird and paint a cage around it. They create a button and click on it to make the bird move around. They teach the bird to flap its wings. Then they teach it to change direction whenever it hits the side of its cage.

Resnick extended this work to investigate people's thinking about decentralized systems in the context of *StarLogo*, a version of Logo that simultaneously controls hundreds or thousands of turtles. He worked with high school students on projects ranging from simulations of slime mold (which, when food is scarce, stop reproducing and move toward one another, forming a cluster with tens of thousands of cells that act as a whole), ants, traffic jams, and geometry. Students' design of *StarLogo* programs often was based

TURTLE MATH

Topic: Shapes

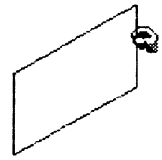
Activity: Parallelograms

Parallelograms

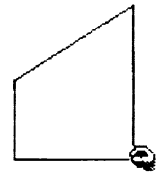
1. Draw a rectangle with these commands.

```
fd 50
rt 90
fd 75
rt 90
fd 50
rt 90
fd 75
rt 90
```

2. Click on the Change Shape tool.

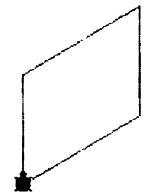


3. Drag this side. What is this shape?
   What do you notice about the turn commands?
   What relationships do you see?



4. Drag this corner. What is this shape now?
   What do you notice about the turn commands?
   What do you notice about the forward commands?

5. Choose Hide Command from the Windows menu.
   Change the shape, then predict what the commands will be.
   Choose Show Command to check your predictions.

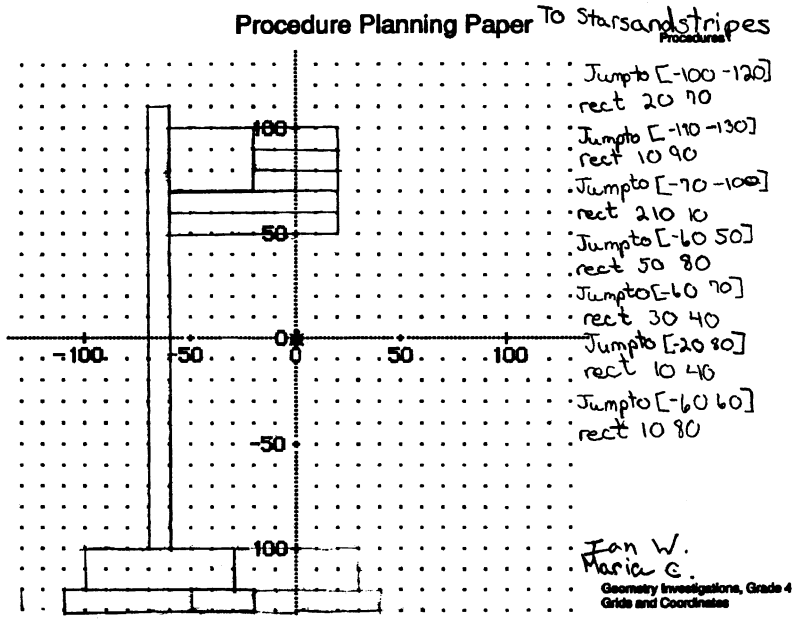


6. Enter the commands **fd 80 rt 45 fd 100**.
   Estimate how much the turtle needs to turn so that the next line—the third—will be parallel to the first.
   Finish drawing this parallelogram, including the final turn.

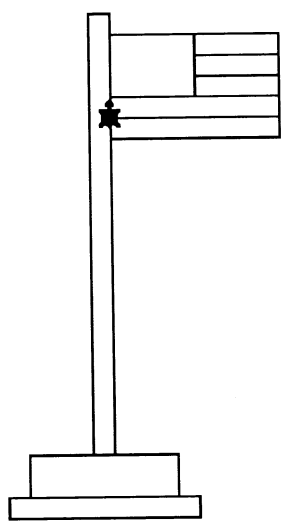
7. Write your own procedure to draw a parallelogram. Use what you just learned.
   Write another procedure to draw a different parallelogram.

Fig. 18. The parallelograms activity uses the Change Shape tool to promote students' reflection.

on an unquestioned assumption of a "leader" or an "outside force" or "seed" for change. For instance, people would quickly assume that a radar trap might cause traffic jams, or an accident. They predicted that without such outside forces, "there was nothing," so traffic would proceed smoothly. However, traffic slowdowns and jams



a.



b.

Fig. 19. To plan their flag (a), Ian, a child labeled as hyperactive, and his partner Maria had to analyze the relationships between intrinsic turtle geometry (e.g., the rectangle command, with two inputs that represent two different measures) and extrinsic geometry (e.g., the coordinate jump to commands, which have two numbers in a list that represent a single entity, a point). Their program ran perfectly the first time (b).

emerge even with no seed and no leader. Across a variety of areas, the simplest and most accurate programs did *not* use "leads or seeds." Instead, they used myriad interactions among objects or beings following a few simple rules. Centralized thinking seems a

strong bias in our thinking. Resnick's research indicates that people can successfully struggle to understand decentralized systems with *StarLogo*. As Resnick puts it, "The flock is organized without an organizer, coordinated without a coordinator" (Resnick, 1994, p. 3).

A final project is Boxer. "Boxer is the name for a multipurpose computational medium intended to be used by people who are no computer specialists. Boxer incorporates a broad spectrum of functions—from hypertext processing, to dynamic and interactive graphics, to databases and programming—all within a uniform and easily learned framework. Most significantly, Boxer is intended to be a new interactive educational medium through which students and teachers can carry out activities spanning a wide range of areas" (diSessa, Abelson, & Ploger, 1991, p. 3). Boxer is typified by concreteness, no distinction between the visual representation and the state of the system; inspectability, any piece of code that is seen can be executed just to see what happens; and hierarchical representation, Boxer has boxes inside boxes. Programmers store both data and programs in boxes. In this way, Boxer can control and present "nested complexity."

In all cases, Boxer has a consistent spatial metaphor. Boxer environments also illustrate the notion of "open tool sets," which involve a greater number of smaller units than conventional applications (diSessa, 1997). These units are to be modified, extended, and combined by teachers and students. Open tools can be copying into any document or work environment, they can interact with that environment, they can be "opened" to show users how they work, and they are modifiable.

2.9. Research on Logo

The following is a summary of the findings (reviews of numerous additional studies are available in Clements & Bautista, 1992b, in press; Clements & Meredith, 1993; Clements & Sarama, 1997; McCoy, 1996). Used appropriately, computer programming has been shown to help students:

- develop higher levels of mathematical, especially geometric, thinking (Clements & Battista, 1989, 1990, in press; Olive, 1991);
- learn geometric concepts and skills, including two-dimensional figures, angles, symmetry, congruence, and geometric motions (Clements & Battista, in press; Clements, Battista, Sarama, & Swaminathan, 1996; Clements, Battista, Sarama, Swaminathan, & McMillen, 1997; Edwards, 1991), although teacher guidance is important, for example, confusions between the angle of turn and the constructed angle can persist for years (Clements, 1987; Cope & Simmons, 1991; Hoyles & Sutherland, 1986; Kieran, 1986);
- gain "entry" to the use of the powerful tool of algebra (Noss & Hoyles, 1992);
- develop concepts of ratio and proportion (Hoyles & Noss, 1989);
- form more generalized and abstract views of mathematical objects (Clements & Battista, 1992a; Noss & Hoyles, 1992);
- develop problem-solving abilities, especially particular skills (e.g., problem decomposition, systematic trial and error) and higher-level metacognitive abilities (Clements, 1990; Delclos & Burns, 1993; Kromhout & Butzin, 1993);

- enhance the social interaction patterns (Clements & Nastasi, 1988; Nastasi & Clements, 1992; Nastasi et al., 1990).

Many of the newer versions of Logo have not been extensively researched, but there are some positive signs. Indications from the first field test were that the research-based features of *Turtle Math* support and encourage students' mathematical investigations (Clements & Sarama, 1995). In a follow-up study, *Turtle Math* was tested over yearlong use in two fourth-grade classrooms (Sarama, 1995). Support was found for most design principles. For example, *Turtle Math* facilitated students' construction of the abstract from the visual conceptually; the level of students' abstraction increased with the amount of time they worked with the concepts in *Turtle Math*. As a second example, students used the turtle metaphor to think about the situation on computer even when the situations went beyond typical turtle geometry. They also, then, made connections between math topics usually compartmentalized by the school curriculum when working on *Turtle Math* projects that required the use of several different concepts such as coordinates, turn measure, length estimation, fractions, and decimals. The turtle metaphor was also applied to situations outside the computer environment, without teacher mediation, in situations that evoked the students' experience on computer. In brief, *Turtle Math* was effective in transforming these fourth-grade classrooms into the type of mathematical learning environment recommended by the NCTM's *Standards*.

In other studies, we investigated the development of specific mathematics knowledge and competencies in *Turtle Math* environments. One study examined students' development of linear measure concepts (Clements et al., 1997). We observed three levels of strategies for solving our different length problems and showed how *Turtle Math's* features aided students progression from less to more sophisticated strategies. The second study investigated the development of turn and turn measurement concepts (Clements et al., 1996). Students gained experience with physical rotations, especially rotations of their own bodies. In parallel, they gained limited knowledge of assigning numbers to certain turns, initially by establishing benchmarks. A synthesis of these two domains—turn-as-body-motion and turn-as-number—constituted a critical juncture in learning about turns for some students. Again, the environment's measurement tools and structure aided this learning. The third study focused on the development of concepts of geometric figures (Clements, Sarama, & Battista, 1998). Work with *Turtle Math* facilitated students' conceptualization of the properties of geometric shapes and their connection between these properties, measurements, and number conceptions.

Boxes has shown potential to be an exploratory environment and expressive medium for a wider range of mathematical concepts (diSessa et al., 1991a; note that this and the next issue contain several articles on Boxer and its application in mathematics education). For example, sixth graders used it to solve complex problems in the physics of motion by manipulating part of the display of the microworld that also functioned as a working part of the program (Adams & diSessa, 1991). They also "reinvented" graphing as a means of representing motion (diSessa, Hammer, Sherin, & Kolpakowski, 1991). Secondary students with little prior computing experience learned concepts of sampling and binomial distributions by using, modifying, and creating Boxer tools (Picciotto & Ploger, 1991). The addition of open tools can extend the power of mathematical ideas. In one instance, a

Computer Contributions

Standard	Problem	Drill	Solving
1.	Mathematics as Problem-solving	°	●
2.	Mathematics as Communication	°	❍
3.	Mathematics as Reasoning	°	●
4.	Mathematical Connections	-	❍
5+	Achievement (content)	❍	●
Beliefs about mathematics		-	●
Affect, motivation, autonomy		°	●
Social construction of mathematics		°	❍
Mathematical creativity		-	❍

Key: Level of contribution

- little or negative
° small
❍ medium
● large

Fig. 20. Computer contributions: problem-solving software.

vector tool set catalyzed instruction on motions and student investigation and appropriation of the tools and the ideas of vectors; for example, students used the tools in creating video games (diSessa, 1997). Anticipating the next section, there are reports of impressive projects that students constructed with Boxer. One student constructed an eight-level adventure game. He used a graduate student's graphing tool, pulling out only the essence of it for his game.

The newer versions of Logo have decided advantages and potential. However, used wisely, research has shown benefits of even the least "sophisticated" versions. Such use of computers can make substantially greater contributions to important goals of mathematics education (see Fig. 20).

3. Projects

Two boys were supposed to build on what they had learned about drawing rectangles with Logo. They started drawing a man. They made a rectangle or two, argued a bit, and frittered away their time. One looked at a large rectangle. "Hey! That's Bruce Smith!" "Yes!" answered his partner. For the next week they continued making what was to *me* the same

drawing—but with an affective and cognitive engagement indicating that it was not so for *them*. It was *their* work, *their* idea, *their* project.

A project is an extensive undertaking requiring concerted effort. Projects can be engaging, but, as teachers, we can find it difficult to balance children's interests with our educational goals. Also daunting are demands regarding materials, orchestration, and management. Computers ameliorate these difficulties in project work.

3.1. Shared knowledge building

Conscious, cooperative development of shared knowledge is the focus of the Computer Supported Intentional Learning Environments (CSILE) project. Active building of knowledge, or constructivism, is becoming increasingly accepted by the educational community. Few classrooms, however, directly encourage students to view what they do in school as the construction of knowledge (Scardamalia & Bereiter, 1992). CSILE's goal is to do so.

CSILE is a networked hypermedia system in which students create the entire database. They produce notes—text, graphics, or both—and enter them into a communal database. These notes range from answering a specific question ("What did you find out in the heating and cooling experiment?") to making a creative drawing unrelated to any class assignment. Students store their notes with key words, and search for others' notes according to these words, to complete projects or just to browse.

Students share information and strategies. One student invented a "making animals" routine that consisted of choosing geometric shapes and forming them into an animal shape, then deleting unnecessary parts of the shapes. The communal database is a particularly powerful medium for the dissemination of such inventions (Scardamalia et al., 1992).

Teachers from first grade to high school are trying out CSILE. The developers' research (Scardamalia et al., 1992) indicates that such use results in increases in amounts of writing, depth of explanation, knowledge quality, question asking, solving complex mathematical word problems, and standardized achievement test scores. Also increased is student collaboration. Control groups possessed "shallow" conceptions of learning, seeing it as a matter of paying attention, doing assigned work, and memorizing. CSILE classrooms exhibited a "deep" conception, seeing learning as dependent on thinking and understanding.

Bringing scientific processes into the classroom usually has implied a focus on individual students; CSILE focuses on the classroom functioning like a scientific community (Scardamalia et al., 1994). One class was working on the inheritance of characteristics. A boy entered a telling one-line note: "Mendel worked on Karen's question." This puts the usual top-down delivery on its head. Students are part of an ongoing process of building knowledge.

What is the unique role of the computer in these classes? Perhaps that it opens up a new channel for communication, one that is not mediated through the teacher (Scardamalia et al., 1994). Research comparing CSILE to classrooms with similar goals but without CSILE shows that students in the computer classrooms talk more about high-level goals. Control students get lost in the lower-level problems and lose sight of their goal. These groups, working face-to-face, varied widely in the extent to which members contributed; some hardly contributed. Competitiveness and other social factors led to rejection of good ideas. With CSILE, students

participated at virtually equal levels. Finally, CSILE offers record of past goal states and solution ideas. Students review their previous work and move on from there, where control students are more likely to flounder or start over.

3.2. Projects for knowledge design

"What is a fraction?" Fourth grader Debbie drew a circle divided into two, shaded the right half, and stated, "That's a fraction." When the investigator asked her about the other half, she replied, "No. It's not a fraction. It's nothing." Debbie's fraction knowledge was limited, and disconnected from her everyday world (Papert, 1993).

Fortunately, that was pretest time. With the rest of her class, Debbie designed software to teach fractions to third graders. They had the freedom and the responsibility to create their own designs and to teach themselves about fractions and representing fractions to other children. Debbie initially was not emotionally involved in the project, but over several months, in designing her Logo software, she came to appreciate that there are many ways to represent fractions. It happened when she figured out that she could use the animated decorations for her poems to make representations of fractions more aesthetic. Other children asked her about the effects. She began to see fractions everywhere and to appreciate that people "impose" fractions on anything they wish. When asked at the end, "What is a fraction?" Debbie insightfully replied, "Fractions can be put on anything!" She used that as a theme in her software (see Fig. 21, Harel & Papert, 1990).

Harel compared three groups. The first comparison group received the same amount of exposure to Logo programming. Their work was integrated with various curriculum topics, but the problems were short and assigned by the teacher. The second comparison group received Logo once a week in a computer literacy course.

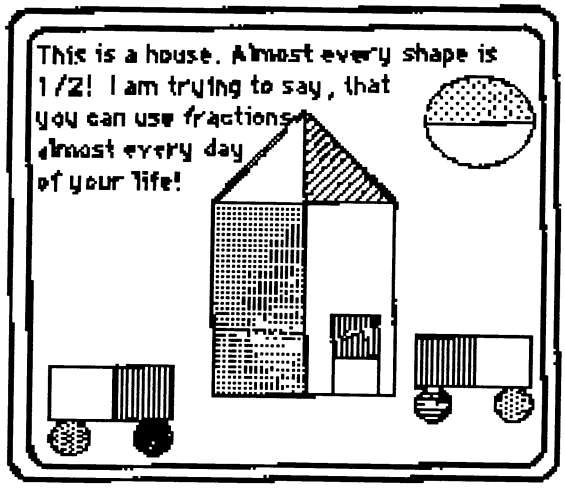


Fig. 21. Debbie illustrates that fractions can be "put on anything!".

The design group showed greater mastery of both Logo and fractions than the two comparison groups. Why? The design children's projects revealed a rich variety of ways to represent fractions. They divided a circle into four regions and flashed two on and off to show two-fourths with the written text "two-fourths"; showed an animated clock; showed a one-dollar bill with four quarters underneath, two of which moved and stopped near the written words "two-fourths of one dollar" (Fig. 22a). These led to ostensibly simple, but phenomenologically deep insights for the students. Sharifa represented

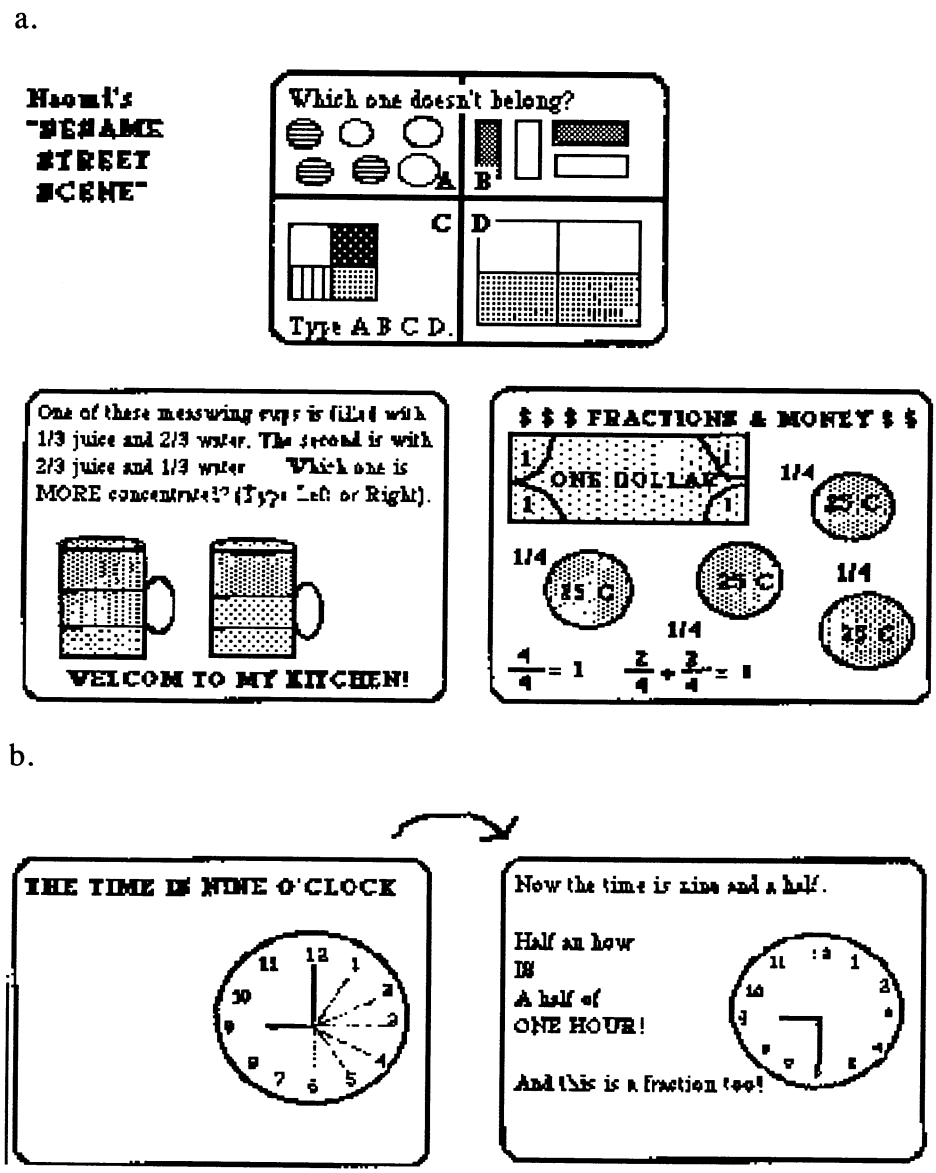


Fig. 22. Students created a variety of ways to represent fractions.

fractions by using a clock, teaching users that, "Half an hour is a half of one hour!" (Fig. 22b). Sherifa's ability to see the analogy between the clock and the turtle is one example of direct contributions of the computer. One might argue that this could be accomplished with other media, but the research showed that computers are critical in allowing students to create different representations of ideas, modify them over months, and share procedures and representations with each other.

Another group of fourth graders designed computer games to teach third graders about fractions (Kafai, 1993). Compared to other groups, students who designed games improved significantly in their knowledge of Logo programming and fractions, although fraction knowledge did not increase as much as in the Harel study.

Thus, designing extensive projects in Logo appears to hold real potential for learning. Designing software to teach concepts seems to be an effective means of learning mathematical content, of creating a personal perspective on that content, and on connecting the content to everyday life. Designing Logo games appears to be an effective means of learning design and programming, and a different, if somewhat less effective, means of learning mathematical content. A good Logo environment—one with features that support the designing—offers real advantages to this type of design activity. A powerful aspect of Logo appears to be giving students control over their own representations of mathematical ideas. Various pictures, animations, and texts can be composed, selected, and combined. Logo programming may contribute to general mathematical ability. These two projects have shown what intensive four months of meaningful Logo programming can do. We have yet to see an investigation of several year's use of Logo.

Repeatedly we hear similar stories. In one fourth-grade class, immediate access to computers supported implementation of larger, more complex, and partially specified projects. This increased students' engagement, motivation, autonomy, and empowerment. Benefits varied insofar as students controlled or influenced the content, process, product, and evaluation of their own learning. Lego-Logo was especially high in offering this type of control (Fisher, 1990, 1991).

The computers' role was helping minimize problems managing more complex tasks. High access to computers helped "absorb" more student variation before the classroom management problems became overwhelming.

3.3. Creativity

I have heard that the physicist Richard Feynmann was asked "How do you learn problem solving?" He said that the first thing you have to do is listen to what the problem is trying to tell you, before you tell the problem all the mathematics you know. Technology allows us to listen to problems, play with them, and listen to them in different ways. And play is of the essence. Jerome Bruner has shown that students encouraged to play with materials first were far more creative in solving problems with those materials. He suggests that play loosens the coupling between ends and means and allows for exploration of different combinations. In work, we hold the end steady and vary the means until we achieve our end, but in play, we can also do the opposite. For

Bruner, unprompted metacognition is nothing more or less than internalized play (Bruner, 1985). Bruner's research indicates that play is richest when the material provided has a clear-cut variable means–end structure, has some constraints, and yields feedback that students can interpret on their own. Interestingly, these descriptions—originally about puzzles and building blocks—fit certain computer environments that we have discussed quite well.

Logo may make it possible to play with certain mathematical ideas creatively earlier than is currently believed (Carmichael et al., 1985; Papert et al., 1979). Such activity engenders both cognitive and affective involvement with mathematics. In the words of one student, "I've thought about circles in ways I've never considered before" (Carmichael et al., 1985, p. 285).

In such environments, pupils experience, probably for the first time, the power to create and be in control of solving their own mathematical problems. Most of their teachers believed that this was enough reason to justify Logo programming as an activity in the mathematics classroom. Students agree: "If we didn't have the computer, what could we use to say that the electricity should flow and then it should stop? Where would we put our knowledge? We can't just leave it in our heads. We know it, we think it, but our programs would stay in our heads" (Sasha, p. 7) (Winer & Trudel, 1991, p. 51).

Computer Contributions

Standard	Problem	Drill	Solving	Projects
1.	Mathematics as Problem-solving	°	●	○
2.	Mathematics as Communication	°	○	●
3.	Mathematics as Reasoning	°	●	●
4.	Mathematical Connections	-	○	●
5+	Achievement (content)	○	●	○
Beliefs about mathematics		-	●	●
Affect, motivation, autonomy		°	●	●
Social construction of mathematics		°	○	●
Mathematical creativity		-	○	●

Key: Level of contribution

- little or negative
° small
○ medium
● large

Fig. 23. Computer contributions: project software.

This stands in contrast to structured CAI activities that, due to their restrictions on solution strategies, extrinsic rewards, built-in evaluations, and lack of playfulness, may negatively affect mathematical creativity (Clements, in press). Fig. 23 summarizes the computer's contribution to mathematics projects.

4. Unique contributions of computers

4.1. Provide environment to test ideas and feedback

Computer programs such as those suggested here provide an environment in which students can try out their ideas *and simultaneously* receive feedback on those ideas. As another example, Hoyles and Noss (1989) found that on a geometric proportion task, students used additive strategies on paper-and-pencil tasks, but none adopted such strategies on the related Logo tasks. The authors trace this catalytic effect of Logo to the interaction between students' formalization and computer feedback. Students formalize proportional relationships algebraically in the form of Logo programs. They receive feedback regarding their mathematical intuitions through the geometric effects. On pencil-and-paper, the first of these, while present, is less salient; the second is absent. For example, students transformed a strategy of addition with adjusted increments to a strategy of adding a fraction of a variable to that variable when they moved to the computer.

4.2. Provide mirrors to mathematical thinking

These features combine to make the computer environments mirrors of students' geometric thinking. Researchers and teachers consistently report that in such contexts students cannot "hide" what they do not understand. Difficulties and misconceptions that are easily hidden by traditional approaches emerge. This leads to some frustration for both teachers and students, but also to greater development of mathematics abilities (Clements & Battista, 1989; Yerushalmy et al., 1987). For teachers willing to work with and listen to students, such environments provide a fruitful setting for learning to take the student's perspective on analyzing mathematical situations and for discovering previously unsuspected abilities for students to construct sophisticated ideas if given the proper tools, time, and teaching.

4.3. Encourage autonomy

Because students may test the ideas for themselves and receive feedback, computers can aid students in moving from naive to empirical to logical thinking and encourage students to make and test conjectures, rather than relying strictly on authority. In addition, the environments appear conducive not only to posing problems, but to wondering and to playing with ideas. Research suggests that computer environments such as those considered here can enable "teaching children to be mathematicians vs. teaching about mathematics" (Papert, 1980, p. 177) insofar as making and testing conjectures, posing problems, and playful engagement with ideas is considered the role of the mathematician.

4.4. Link the general and the specific

Computer environments encourage the manipulation of specific screen objects in ways that assist students in viewing them as mathematical objects and as representatives of a class of objects. Such activities develop students' ability to reflect on the properties of the class of objects and to think in a more general and abstract manner. Returning to Hoyles and Noss' example, students using computers abandoned additive thinking because the computer provided a way to think about the general within the specific. The paper-and-pencil mode activated a fixed answer to a fixed question. The computer allowed exploration as an antidote to mental "blocks" and activated a dynamic answer. Posing the task of writing a super-procedure that would handle all cases promoted additional development. Computer environments encourage more generalized and abstract views of mathematical objects (Clements & Battista, 1994).

Numerical computation provided by spreadsheets, Logo, and other programs can help students build intuitive understanding of variables, relationships among variables, and functions. This helps students make the transition from arithmetic to algebraic reasoning (Ainley, 1997; Confrey, in press; Fey, 1989; Hoyles & Noss, 1987b; Sarama & Clements, 1998).

4.5. Link the symbolic to the visual

Computers can promote the connection of formal representations with *dynamic* visual representations, supporting the construction of mathematical strategies and ideas out of initial intuitions and visual approaches. The integration of ideas from algebra and geometry is particularly important and computer tools play a critical role in that integration (NCTM, 1989, p. 125; President's Committee of Advisors on Science and Technology—Panel on Educational Technology, 1997).

4.6. Catalyze natural and mathematical language

Fortunately, students are not surprised that the computer does not understand natural language, so they have to formalize their ideas to communicate them. Students formalize about fives times as often using computers as they do using paper (Hoyles, Healy, & Sutherland, 1991). As example from that study, students struggled to express the number pattern that they explored on spreadsheets. They used phrases such as "this cell equals the next one plus 2; and then that result plus this cell plus 3 equals this." Their use of the structure of the spreadsheet's rows and columns, and their incorporation of formulas in the cells of the spreadsheet, helped them more formally express the generalized pattern they invented.

The need for complete and abstract explication in many computer environments accounts in part for students' creation of richer geometric ideas (Johnson-Gentile, Clements, & Battista, 1994). That is, in Logo students have to specify steps to a noninterpretive agent. In contrast, when "intuition is translated into a program it becomes more obtrusive and more accessible to reflection" (Papert, 1980, p. 145). In one study, we attempted to help a group of students using noncomputer manipulatives become aware of these motions. However, students described these physical motions to other students who understood the task. In contrast, students using the

computer specified motions to the computer, which does not "already understand." The specification had to be thorough and detailed. Students observed the results of, reflected on, and corrected commands. This led to more discussion of the motions themselves, rather than just the shapes (Johnson-Gentile et al., 1994).

4.7. Emphasize concepts and problem solving

Computers help de-emphasize routine aspects of mathematical work and emphasize conceptual thinking and problem solving. This can lead students to view mathematics as a source of models for real-world phenomena, and so to joint teachers in making sense of mathematics and mathematical models collaboratively.

4.8. Facilitate problem solving

Computer environments may facilitate problem solving by allowing students to build on their initial intuitive visual approaches and construct more analytic approaches. In this way, early ideas and strategies may be precursors of more sophisticated mathematics. One boy wrote a procedure to draw a rectangle (Clements & Battista, in press). He created a different variable for the length of each of the four sides. He gradually saw that he only needed two variables, as the lengths of the opposite sides are equal. In this way, he recognized that the variables could represent values rather than specific sides of the rectangle. There was no teacher intervention during this time; Logo provided the structure by requiring a symbolic representation and by allowing the boy to link the symbols to the figure (c.f. Hoyles & Noss, 1987a).

4.9. Serve as effective manipulatives

Besides the advantages already discussed, computer manipulatives can be uniquely effective in their roles of:

- offering flexibility,
- changing arrangement or representation,
- storing and later retrieving configurations,
- recording and replaying students' actions, and
- changing the nature of the manipulative.

4.10. Support projects

Computer environments can also facilitate larger, longer, mathematical investigations in that they help students:

- communicate with one another, sometimes without teacher mediation, and with remote data bases and people;

- focus on high-level goals;
- participate at equal levels;
- view, copy, modify, and use others' work; and
- actively explore and "play" with more activities that have scientific and mathematical content (Papert, 1993).

4.11. Encourage more positive social interaction

One of the surprising research results is that the largest, most consistent benefits of the computer may be in the social and emotional domain. For example:

- Students cooperate more in computer environments and they cooperate on *learning* (Clements & Nastasi, 1992).
- They also disagree more—but they disagree about ideas, and they are more likely to successfully resolve these disagreements, synthesizing their ideas (Clements & Nastasi, 1992; Nastasi et al., 1990).
- The reasons for such facilitation of interaction include all the contributions already discussed, as well as such simple features as the shared display and single keyboard.

In conclusion, the *more* we move to problems and projects, the *more* computers make a unique contribution! And the more we see that education is revitalized.

5. Unique *challenges* of computers: steering around the potholes

An early concern was that computers would replace teachers. Now most of us see that technology doesn't take work off our shoulders; if anything, it adds to the effort we must expend. Several evaluation projects, while showing that computers can be effective educational aids overall, also show they can impose stressful workloads, especially upon first-year teachers and principals (Kromhout & Butzin, 1993).

5.1. Software

Of course, one needs hardware, but content and pedagogy of the software should receive primary emphasis. Availability is requisite, but most attention should be given to achieving goals of educational reform through computer integration (President's Committee of Advisors on Science and Technology—Panel on Educational Technology, 1997).

There are three additional general research implications for the selection and use of software. First, use an appropriate combination of off- and on-computer activities. Each has its own advantages; helping students seeing the connection between the two is perhaps the most significant. Become familiar with all software ahead of time.

Second, consider technology less as a pedagogical tool and more as a mathematical tool (Fey, 1991). Graphing tools and spreadsheets should encourage us to reconsider the algebra

curriculum just as calculators and computers have made us reconsider the arithmetic curriculum. If we see algebra as primarily the study of functions and their representations, we might use function plotters, curve fitters, and symbol manipulators as in the Computer-Intensive Algebra project. Our students too would see algebra as a source of mathematical models and ask "What if...?" questions: What if problem conditions change? What if the goal changes?

Third, consider technology as a thinking tool. Only a few computer-using teachers have students use *any* one kind of program more than five times during the year. For example, only 14% of mathematics teachers who use computer had students use tutorial CAI programs more than five times (Becker, 1991). The programs students use should become tools for thinking mathematically. We need to be wary of providing a potpourri of applications with no internal coherence. Research comparing Logo to a set of utility and problem-solving programs demonstrated that stronger feelings of control and mastery emerged with Logo (Clements & Nastasi, 1992). Use extensible programs for long periods of time, across topics when possible, to engage students in meaningful problems and projects.

5.2. Navigating with students

Computers do not stand alone. The curriculum in which computers are embedded, and the teacher who *chooses*, *uses*, and *infuses* these programs, are essential elements in realizing the full potential of technology (Hancock et al., 1992; Henry, 1995; Littlefield et al., 1988; Olson, 1988; Schofield, Evans-Rhodes, & Huber, 1990; Swan, Mitrani, Cheung, Guerrero, & Schoener, 1991; Webb & Lewis, 1988). This section draws implications for teaching from this research.

Computers allow students to build their mathematical ideas. As teachers, we play a critical role in *mediating students' computer work*, but we must remember: The student's motor is always running; our job as teachers is to build roads, place signs, direct traffic, and teach good driving, but not to drive the car. What is good mediation? We can do the following:

- bring the mathematics in computer work to a conscious level of awareness and systematically extend the ideas encountered;
- focus attention on critical mathematics aspects of activities, emphasizing the need for consistency and mathematical language;
- facilitate disequilibrium using computer feedback as a catalyst in strengthening concepts;
- construct links, or mappings, between computer and noncomputer work; and
- provoke reflection and prediction.

Students do not automatically transfer knowledge gained in one situation to another. Repetition is not sufficient. Questions that cause students to reflect on what they were doing are instrumental. One research recorded the following: "Ingrid rotated a rectangle [one degree at a time] 540 times and produced a circle that was filled in. When I asked her if it had gone

over part of the circle again she said 'yes,' thought for a minute and then said, 'I bet 360 degrees would do it. I'm going to try it'." (Carmichael et al., 1985, p. 316).

5.3. Computers in two classrooms

Olson's research tells a tale of several sites, only one of which is a success story (Olson, 1988). In this case, what distinguishes them was *not* number of computers or background of the students. All had moderate computer access;[5] all were from working-class backgrounds.

At Maple Hills, boys wishing to play games dominated the informal use of computers, precluding others from gaining expertise. The teacher in charge of computers, Jack, took a teacher-centered approach, and exhausted himself trying to give explanations in an inadequate period. Another teacher let students "experiment" but didn't know how to integrate this activity with the rest of his instruction. Teachers received little support. "It's one more thing added on, and they never take things off." So, a huge burden fell on Jack, to teach everyone. He did minimal basics and then "enrichment" for a few. Others played games and had "fun." For example, they played "Lemonade." Bankruptcy was frequent and mastery or sense of self-accomplishment rare.

In Spruce Grove, children consistently made "money" selling lemonade and the best players scored high. Their teachers viewed Lemonade not as a game, but an opportunity to teach mathematical principles and the power of group organization. They would record the "best price" for a temperature and humidity, compiling these into grand lists. They complained that the game was "too easy" because it had no "randomization factor"; interesting language for sixth graders from a working-class background. This was not uncommon; later they learned to program an algorithm (a term they also used) for determining whether a number was prime or not. These students learned that knowledge is valuable and that if you cooperate to acquire it, it works for you.

The computer demanded much time...at *all* the sites. But only in Spruce Grove did teachers see it not as one more thing to do, but as an auxiliary tool to be integrated into a total program. The computer makes life more complicated, but also more interesting and rich. When teachers utilize their own skills and judgement, teaching and learning in computer-based classrooms is dramatically more student-centered and individualized than teaching and learning in traditional classroom settings (Becker & Ravitz, 1999; Swan et al., 1991). The computer is *not* a panacea, but it *should not* be merely an adjunct.

To accomplish this, teachers need computer access, support for teaching reform, technical support, validation, collegial support, and extra time (Dexter, Anderson, & Becker, 1999; Sarama, Clements, & Henry, 1998). In addition, reformers should recognize the threat of unrealized potential and missed opportunities due to divergent beliefs of the various social groups involved, from teachers to students to administrators to parents to curriculum and software developers (Sarama et al., 1998). For example, everyone in one school believed that

[5] This, then, does not contradict the importance of having adequate numbers of computers. Computer access is a necessary, but not sufficient, condition for educational success.

computer access was important. However, whereas the administrators and parents believed that the lab and its schedule were adequate, the teachers believed that it was unworkable to implement the reform.

Thus, more emphasis must be given to professional development and support. Teachers need available help with technical problems, but also with pedagogic challenges of choosing software, teaching with technology, and organizing projects (Hancock et al., 1992; McDougall, 1997; President's Committee of Advisors on Science and Technology—Panel on Educational Technology, 1997). This help must address the challenges of technological reform as seen *through their eyes* (Sarama et al., 1998).

6. Conclusions

Thinking about computers in mathematics education should not mean thinking computers. It should mean rethinking mathematics education. When I talked to teachers about computers, real problems, and projects, I used to be cautious. . .even a bit defensive. I think I'm going to turn that around. Now I want to ask those I talk to how they can justify a year of mathematics in school *without* that kind of activity.

Research offers directions at today's crossroads. An important decision is among three paths that differ in the goals and types of computer applications. Those traveling on the first path use simple computer games for "rewards" or occasionally use drill software but do not integrate it into their wider educational program. Those traveling on the second path integrate drill exercises and other structured software tasks into their programs. Those traveling on the third path use problem-solving software, geometric construction programs, function and graphing software, and Logo to engage their students in substantive problems and long-range projects.

Research suggests that the first path leads nowhere educationally useful. Teachers might better invest efforts and resources elsewhere. The second path is educationally plausible. Well-planned, integrated computer activities can increase achievement in cost-effective ways.

The third path is more challenging—in time, in effort, in commitment, and in vision. This path alone, however, offers the potential for substantive educational innovation consonant with NCTM Standards. Technology actuated the world of tomorrow. Technology—used thoughtfully and creatively rather than as a teaching machine—can engender and support educational environments that will help students to flourish in this intensively mathematical world.

Finally, research shows that decisions about these paths are not final. Many teachers take the second path, return one or more years later to the crossroads, and turn to the third path. These teachers see the potential of computers, and extend their vision. I leave you, and my path metaphor, with the vision of the poet, Frost:

> Two roads diverged in a wood, and I—I took the one less traveled by, And that has made all the difference.[6]

[6] "The road not taken," by Robert Frost.

References

Adams, S. T., & diSessa, A. A. (1991). Learning by "cheating": students' inventive ways of using a Boxer motion microworld. *Journal of Mathematical Behavior, 10* (1), 79–89.

Ainley, J. (1997). Purposeful contexts for formal notation in a spreadsheet environment. *Journal of Mathematical Behavior, 15* (4), 405–422.

Baulac, Y., Bellemain, F., & Laborde, J.-M. (1988). *Cabri Géomètrie, un logiciel d'aide à l'apprentissage de la geometrie, logiciel et manuel d'utlizisation*. Paris: Cedic-Nathan.

Becker, H. J. (1991). How computers are used in United States schools: basic data from the 1989 I.E.A. computers in education survey. *Journal of Educational Computing Research*, *7*, 385–406.

Becker, H. J. (1992). Computer-based integrated learning systems in the elementary and middle grades: a critical review and synthesis of evaluation reports. *Journal of Educational Computing Research*, *8*, 1–41.

Becker, H. J., & Ravitz, J. (1999). The influence of computer and Internet use on teachers' pedagogical practices and perceptions. *Journal of Research on Computing in Education, 31* (4), 356–384.

Browning, C. A. (1991). Reflections on using Lego®TC Logo in an elementary classroom. In E. Calabrese (Ed.), *Proceedings of the third european Logo conference* (pp. 173–185). Parma, Italy: Associazione Scuola e Informatica.

Bruner, J. (1985). On teaching thinking: an afterthought. In S. F. Chipman, J. W. Segal, & R. Glaser (Eds.), *Thinking and learning skills*. *Research and open questions*, (vol. 2). Hillsdale, NJ: Erlbaum.

Carmichael, H. W., Burnett, J. D., Higginson, W. C., Moore, B. G., & Pollard, P. J. (1985). *Computers, children and classrooms: a multisite evaluation of the creative use of microcomputers by elementary school children*. Toronto, Ontario, Canada: Ministry of Education.

Chazan, D. (1989). Instructional implications of a research project on students' understandings of the differences between empirical verification and mathematical proof. In D. Hergert (Ed.), *Proceedings of the first international conference on the History and Philosophy of Science in Science Teaching* (pp. 52–60). Tallahassee, FL: Florida State University Science Education and Philosophy Department.

Choi koh, S. S. (1999). A student's learning of geometry using the computer. *Journal of Educational Research, 92* (5), 301–311.

Clements, D. H. (1987). Longitudinal study of the effects of Logo programming on cognitive abilities and achievement. *Journal of Educational Computing Research*, *3*, 73–94.

Clements, D. H. (1990). Metacomponential development in a Logo programming environment. *Journal of Educational Psychology*, *82*, 141–149.

Clements, D. H., & Battista, M. T. (1989). Learning of geometric concepts in a Logo environment. *Journal for Research in Mathematics Education*, *20*, 450–467.

Clements, D. H., & Battista, M. T. (1990). The effects of Logo on children's conceptualizations of angle and polygons. *Journal for Research in Mathematics Education*, *21*, 356–371.

Clements, D. H., & Battista, M. T. (1992a). *The development of a Logo-based elementary school geometry curriculum*. Final Report: NSF Grant No.: MDR-8651668. Buffalo, NY: State University of New York at Buffalo; Kent, OH: Kent State University.

Clements, D. H., & Battista, M. T. (1992b). Geometry and spatial reasoning. In D. A. Grouws (Ed.), *Handbook of research on mathematics teaching and learning* (pp. 420–464). New York: Macmillan.

Clements, D. H., & Battista, M. T. (1994). Computer environments for learning geometry. *Journal of Educational Computing Research*, *10* (2), 173–197.

Clements, D. H., & Battista, M. T. (in press). *Logo and geometry.*

Clements, D. H., Battista, M. T., Sarama, J., & Swaminathan, S. (1996). Development of turn and turn measurement concepts in a computer-based instructional unit. *Educational Studies in Mathematics*, *30*, 313–337.

Clements, D. H., Battista, M. T., Sarama, J., Swaminathan, S., & McMillen, S. (1997). Students' development of length measurement concepts in a Logo-based unit on geometric paths. *Journal for Research in Mathematics Education*, *28* (1), 70–95.

Clements, D. H., & McMillen, S. (1996). Rethinking "concrete" manipulatives. *Teaching Children Mathematics*, *2* (5), 270–279.

Clements, D. H., & Meredith, J. S. (1993). Research on Logo: effects and efficacy. *Journal of Computing in Childhood Education*, *4*, 263–290.

Clements, D. H., & Meredith, J. S. (1994). *Turtle math [Computer program]*. Montreal, Quebec: Logo Computer Systems Inc. (LCSI).

Clements, D. H., & Nastasi, B. K. (1988). Social and cognitive interactions in educational computer environments. *American Educational Research Journal*, *25*, 87–106.

Clements, D. H., & Nastasi, B. K. (1992). Computers and early childhood education. In M. Gettinger, S. N. Kratochwill, & T. R. Kratochwill (Eds.), *Advances in school psychology: preschool and early childhood treatment directions* (pp. 187–246). Hillsdale, NJ: Erlbaum.

Clements, D. H., & Sarama, J. (1995). Design of a Logo environment for elementary geometry. *Journal of Mathematical Behavior*, *14*, 381–398.

Clements, D. H., & Sarama, J. (1997). Children's mathematical reasoning with the turtle metaphor. In L. D. English (Ed.), *Mathematical reasoning: analogies, metaphors, and images* (pp. 313–337). Mahway, NJ: Erlbaum.

Clements, D. H., Sarama, J., & Battista, M. T. (1998). Development of concepts of geometric figures in a specially-designed Logo computer environment, *20*, 47–64.

Confrey, J. (in press). Designing mathematics education: the role of new technologies. In: C. Fisher (Ed.), *Education and technology: reflections on a decade of experience in classrooms*. San Francisco: Jossey-Bass and Apple Corp.

Cope, P., & Simmons, M. (1991). Children's exploration of rotation and angle in limited Logo microworlds. *Computers in Education*, *16*, 133–141.

de Villiers, M. D. (1995). An alternative introduction to proof in dynamic geometry. *MicroMath*, *11* (12), 14–19.

Delclos, V. R., & Burns, S. (1993). Mediational elements in computer programming instruction: an exploratory study. *Journal of Computing in Childhood Education*, *4*, 137–152.

Dexter, S. L., Anderson, R. E., & Becker, H. J. (1999). Teachers' views of computers as catalysts for changes in their teaching practice, *31*, 221–239.

diSessa, A. A. (1997). Open toolsets: new ends and new means in learning mathematics and science with computers. In E. Pehkonen (Ed.), *Proceedings of the 21st conference of the international group for the psychology of mathematics education* (pp. 47–62). Lahti, Finland: University of Helsinki.

diSessa, A. A., Abelson, H., & Ploger, D. (1991). An overview of Boxer, *10*, 3–15.

diSessa, A. A., Hammer, D., Sherin, B., & Kolpakowski, T. (1991). Inventing graphing: meta-representational expertise in children, *10*, 117–160.

Dixon, J. K. (1997). Computer use and visualization in students' construction of reflection and rotation concepts, *97*, 352–358.

Edwards, L. D. (1991). Children's learning in a computer microworld for transformation geometry. *Journal for Research in Mathematics Education*, *22*, 122–137.

Eisenberg, M. (1993). *Creating software applications for children: some thoughts about design*. University of Colorado.

Enkenberg, J. (1994). Situated programming in a LEGOLogo environment. *Computers and Education*, *22* (1–2), 119–128.

Fey, J. T. (1989). Technology and mathematics education: a survey of recent developments and important problems. *Educational Studies in Mathematics*, *20*, 237–272.

Fey, J. (1991). Calculators, computers, and algebra in secondary school mathematics. In J. P. Becker, & T. Miwa (Eds.), *Proceedings of the U.S.–Japan seminar on computer use in school mathematics* (pp. 103–119). Honolulu, Hawaii: The East–West Center.

Filimonov, R., & Kreith, K. (1990). *Euclid turns to programming*. Davis: University of California.

Filimonov, R., & Sendov, B. (1990). *A brief design rationale for the plane geometry system*. Bulgaria: Sofia University.

Fisher, C. W. (1990/91). Some influences of classroom computers on academic tasks. *Journal of Computing in Childhood Education*, *2*, 3–15.

Flake, J. L. (1990). An exploratory study of Lego Logo. *Journal of Computing in Childhood Education*, *1* (3), 15–22.

Hancock, C., Kaput, J. J., & Goldsmith, L. T. (1992). Authentic inquiry with data: critical barriers to classroom implementation. *Educational Psychologist*, *27*, 337–364.

Harel, I., & Papert, S. (1990). Software design as a learning environment. *Interactive Learning Environments*, *1* (1), 1–32.

Henry, J. J. (1995). *The role of teachers' beliefs in the implementation of a mathematics curriculum innovation*. Unpublished doctoral dissertation, State University of New York at Buffalo.

Hickey, M. G. (1993). Computer use in elementary classrooms: an ethnographic study. *Journal of Computing in Childhood Education*, *4*, 219–228.

Holt, J. (1982). *How children fail*. New York: Dell.

Hoyles, C., Healy, L., & Sutherland, R. (1991). Patterns of discussion between pupil pairs in computer and non-computer environments. *Journal of Computer Assisted Learning*, *7*, 210–228.

Hoyles, C., & Noss, R. (1987a). Children working in a structured Logo environment: from doing to understanding. *Recherches en Didactique des Mathématiques*, *8*, 131–174.

Hoyles, C., & Noss, R. (1987b). Synthesizing mathematical conceptions and their formalization through the construction of a Logo-based school mathematics curriculum. *Int. J. Math. Educ. Sci. Technol.*, *18*, 581–595.

Hoyles, C., & Noss, R. (1989). The computer as a catalyst in children's proportion strategies. *Journal of Mathematical Behavior*, *8*, 53–75.

Hoyles, C., & Sutherland, R. (1986). *When 45 equals 60*. London England: University of London Institute of Education Microworlds Project.

Jackiw, N. (1995). *The geometer's sketchpad [Computer program]*. Berkeley, CA: Key Curriculum Press.

Johnson-Gentile, K., Clements, D. H., & Bautista, M. T. (1994). The effects of computer and noncomputer environments on students' conceptualizations of geometric motions. *Journal of Educational Computing Research*, *11* (2), 121–140.

Kafai, Y. B. (1993). *Minds in play: computer game design as a context for children's learning*. Unpublished doctoral dissertation, Harvard University.

Kamii, C. (1986). Place value: an explanation of its difficulty and educational implications for the primary grades. *Journal of Research in Childhood Education*, *1*, 75–86.

Kelman, P. (1990). *Alternatives to integrated instructional systems*. Paper presented at the meeting of the National Educational Computing Conference, Nashville, TN (June).

Kieran, C. (1986). Logo and the notion of angle among fourth and sixth grade children. In C. Hoyles, & L. Burton (Eds.), *Proceedings of psychology in mathematics education*, (Vol. 10, pp. 99–104). London, England: City University.

Kromhout, O. M., & Butzin, S. M. (1993). Integrating computers into the elementary school curriculum: an evaluation of nine Project CHILD model schools. *Journal of Research on Computing in Education*, *26* (1), 55–69.

Laborde, C. (1995). Designing tasks for learning geometry in a computer-based environment. In L. Burton, & B. Jaworski (Eds.), *Technology in mathematics teaching*. Bromley, England: Chartwell-Bratt.

Laborde, C. (1996). A new generation of diagrams in dynamic geometry software. In E. Jakubowski, D. Biske, & H. Biske (Eds.), *Proceedings of the eighteenth annual meeting of the North America chapter of the international group for the psychology of mathematics education.* Columbus, OH.

Lafer, S., & Markert, A. (1994). Authentic learning situations and the potential of Lego TC Logo. *Computers in the Schools*, *11* (1), 79–94.

Lampert, M. (1988). *Teachers' thinking about students' thinking about geometry: The effects of new teaching tools*. Technical Report. Cambridge, MA: Educational Technology Center, Harvard Graduate School of Education.

Littlefield, J., Delclos, V. R., Lever, S., Clayton, K. N., Bransford, J. D., & Franks, J. J. (1988). Learning Logo: method of teaching, transfer of general skills, and attitudes toward school and

computers. In R. Mayer (Ed.), *Teaching and learning computer programming: multiple research perspectives* (pp. 111–135). Hillsdale, NJ: Erlbaum.

McCoy, L. P. (1996). Computer-based mathematics learning. *Journal of Research on Computing in Education*, *28*, 438–460.

McDougall, D.E. (1997). *Mathematics teachers' needs in dynamic geometric computer environments: in search of control*. Unpublished doctoral dissertation, University of Toronto, Toronto, Canada

Nastasi, B. K., & Clements, D. H. (1992). Social–cognitive behaviors and higher-order thinking in educational computer environments. *Learning and Instruction*, *2*, 215–238.

Nastasi, B. K., Clements, D. H., & Battista, M. T. (1990). Social–cognitive interactions, motivation, and cognitive growth in Logo programming and CAI problem-solving environments. *Journal of Educational Psychology*, *82*, 150–158.

National Council of Teachers of Mathematics. (1989). *Curriculum and evaluation standards for school mathematics*. Reston, VA: The Council.

National Council of Teachers of Mathematics. (2000). *Principles and standards for school mathematics*. Reston, VA: National Council of Teachers of Mathematics.

Noss, R., & Hoyles, C. (1992). Afterword: looking back and looking forward. In C. Hoyles, & R. Noss (Eds.), *Learning mathematics and Logo* (pp. 427–468). Cambridge, MA: MIT Press.

Olive, J. (1991). Logo programming and geometric understanding: an in-depth study. *Journal for Research in Mathematics Education*, *22*, 90–111.

Olson, C. P. (1988). Computing environments in elementary classrooms. *Children's Environments Quarterly*, *5* (4), 39–50.

Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. New York: Basic Books.

Papert, S. (1993). *The children's machine. Rethinking school in the age of the computer*. New York: Basic Books.

Papert, S., Watt, D., diSessa, A., & Weir, S. (1979). *Final report of the Brookline Logo Project. Part II: Project summary and data analysis* (Logo Memo No. 53). Cambridge, MA: Massachusetts Institute of Technology, Artificial Intelligence Laboratory.

Picciotto, H., & Ploger, D. (1991). Learning about sampling with Boxer. *Journal of Mathematical Behavior*, *10* (1), 91–113.

President's Committee of Advisors on Science and Technology — Panel on Educational Technology. (1997). *Report to the president on the use of technology to strengthen K-12 education in the United States*. Washington, DC: President's Committee of Advisors on Science and Technology — Panel on Educational Technology.

Resnick, M. (1988). LEGO, Logo, and design. *Children's Environments Quarterly*, *5* (4), 14–18.

Resnick, M. (1994). *Turtles, termites, and traffic jams: explorations in massively parallel microworlds*. Cambridge, MA: MIT Press.

Roberts, D. L., & Stephens, L. J. (1999). The effect of the frequency of usage of computer software in high school geometry. *Journal of Computers in Mathematics and Science Teaching, 18* (1), 23–30.

Robyler, M. D., Castine, W. H., & King, F. J. (1988). *Assessing the impact of computer-based instruction: a review of recent research*. New York: Haworth Press.

Russell, S. J., Magdalene, L., & Rubin, A. (1989). *What's real problem?* Paper presented at the meeting of the American Educational Research Association, San Francisco (March).

Sarama, J. (1995). *Redesigning Logo: the turtle metaphor in mathematics education*. Unpublished doctoral dissertation, State University of New York at Buffalo.

Sarama, J., & Clements, D. (1998). Using computers for algebraic thinking. *Teaching Children Mathematics*, *5* (3), 186–190.

Sarama, J., Clements, D.H., & Henry, J.J. (1998). Network of influences in an implementation of a mathematics curriculum innovation. *International Journal of Computers for Mathematical Learning, 3*, 113–148.

Scardamalia, M., & Bereiter, C. (1992). An architecture for collaborative knowledge building. In E. De Corte, M. C. Linn, H. Mandl, & L. Verschaffel (Eds.), *Computer-based learning environments and problem solving* (pp. 41–66). Berlin: Springer-Verlag.

Scardamalia, M., Bereiter, C., Brett, C., Burtis, P. J., Calhoun, C., & Lea, N. S. (1992). Educational applications of a networked communal database. *Interactive Learning Environments*, *2, 45–71.*

Scardamalia, M., Bereiter, C., & Lamon, M. (1994). *The CSILE project: trying to bring the classroom into world 3*. Ontario, Canada, Centre for Applied Cognitive Science Ontario Institute for Studies in Education (OISE).

Schofield, J. W., Evans-Rhodes, D., & Huber, B. R. (1990). Artificial intelligence in the classroom: the impact of a computer-based tutor on teachers and students. *Social Science Computer Review*, *8*, 24–41.

Schwartz, J. L., & Yerushalmy, M. (1986). *The Geometric Supposer series [computer-based courseware] [Computer program]*. Pleasantville, NY: Sunburst Communications.

Silverman, N.S. (1990). *Logo and underachievers*. Unpublished masters thesis. University of the Virgin Islands.

Swan, K., Mitrani, M., Cheung, M., Guerrero, F., & Schoener, J. (1991). *The changing nature of teaching and learning in computer-based classrooms*. Paper presented at the meeting of the American Educational Research Association, Chicago, IL (April).

Thompson, P. W. (1992). Notations, conventions, and constraints: contributions to effective use of concrete materials in elementary mathematics. *Journal for Research in Mathematics Education*, *23*, 123–147.

Thompson, P. W., & Thompson, A. G. (1990). *Salient aspects of experience with concrete manipulatives*. Mexico City: International Group for the Psychology of Mathematics Education.

Webb, N. M., & Lewis, S. (1988). The social context of learning computer programming. In R. Mayer (Ed.), *Teaching and learning computer programming: multiple research perspectives* (pp. 179–206). Hillsdale, NJ: Erlbaum.

Weir, S. (1992). LEGO-Logo: a vehicle for learning. In C. Hoyles, & R. Noss (Eds.), *Learning mathematics and Logo* (pp. 165–190). Cambridge, MA: MIT Press.

Wheatley, G. (1991). Constructivist perspectives on science and mathematics learning. *Science Education*, *75*, 9–21.

Winer, L. R., & Trudel, H. (1991). Children in an educational robotics environment: experiencing discovery. *Journal of Computing in Childhood Education*, *2*, 41–64.

Wiske, M. S., & Houde, R. (1988). *From recitation to construction: teaching change with new technologies*. Technical Report. Cambridge, MA: Educational Technology Center, Harvard Graduate School of Education.

Yerushalmy, M., Chazan, D., & Gordon, M. (1987). *Guided inquiry and technology: a year long study of children and teaching using the Geometric Supposer*. ETC Final Report. Newton, MA: Educational Development Center.