ED 287 710 SE 048 666

AUTHOR Cuneo, Diane O.
TITLE Young Children and Turtle Graphics Programming: Generating and Debugging Simple Turtle Programs.
PUB DATE 86
NOTE 12p.; Paper presented at the Annual Meeting of the American Educational Research Association (San Francisco, CA, April 16-20, 1986). For related documents, see ED 260 800 and SE 048 687.
PUB TYPE Reports - Research/Technical (143) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Computer Assisted Instruction; *Computer Graphics; *Computer Science; Computer Uses in Education; Elementary Education; *Elementary School Science; Preschool Education; *Problem Solving; *Programing; Programing Languages; Science Education; Science Instruction; Sequential Learning
IDENTIFIERS *Turtle Graphics

ABSTRACT

Turtle graphics is a popular vehicle for introducing children to computer programming. Children combine simple graphic commands to get a display screen cursor (called a turtle) to draw designs on the screen. The purpose of this study was to examine young children's abilities to function in a simple computer programming environment. Four and five-year-olds were asked to solve turtle graphics problems requiring two or three commands for a solution. There were two programming-type conditions. First, children were asked to give the complete sequence of commands in advance (i.e., generate a program). Second, if the sequence did not work, they were asked to determine what went wrong and modify it accordingly (i.e., debug their program). Results of the study indicate that young children had difficulty in generating two- or three-command programs. Their ability to debug their own incorrect programs was not as high as expected. Several implications for educators are discussed. Tables and figures are provided. (TW)

Meeting of the American Educational Research Association

San Francisco, CA, April 1986

U S DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

[✓] This document has been reproduced as received from the person or organization originating it

☐ Minor changes have been made to improve reproduction quality

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy

Young Children and Turtle Graphics Programming: Generating and Debugging Simple Turtle Programs

Turtle graphics is a popular vehicle for introducing children to computer programming. Children combine simple graphics commands to get a display screen cursor called a turtle to draw designs on the screen. Anecdotal evidence suggests that young (i.e., prechool-aged) children are able to jump right into creating their own turtle designs. It is not clear, however, whether they are actually capable of computer programming.

The existing literature on young children's problem-solving also offers little clue as to what their computer programming abilities might be. An exception is Klahr and Robinson's (1981) Tower of Hanoi study. Klahr and Robinson presented 4- and 5-year-olds with three-disk Tower of Hanoi problems requiring from one to seven moves for solution. The children were asked to plan (i.e., state but not execute) the required sequence of moves. Most 4-year-olds could give correct plans for up to two-move problems, and most 5-year-olds for up to three-move problems. Since generating a computer program demands an analogous kind of planning, these results suggest that young children are hardly capable of computer programming. It is not clear, however, whether the two problem-solving situations are similar enough to permit this analogy.

The purpose of this study was to examine young children's ability to function in a simple computer programming environment. Four- and 5-year-olds were asked to solve turtle graphics problems requiring two or three commands for solution. There were two programming-type conditions. First, children were asked to give the complete sequence of commands in advance (i.e., generate a program). Second, if the sequence did not work, they were asked to figure out what went wrong and modify it accordingly (i.e., debug their program). Young children's attempts at such programming-type activities would provide important preliminary information on what they can actually do in a computer programming environment, and the kinds of things they find difficult.

Method

Turtle graphics environment. A highly simplified turtle graphics environment was created for the study. There were four possible turtle orientations, facing 0-, 90-, 180-, and 270-degrees (see Figure 1), and four legal commands, FORWARD (F), BACK (B), RIGHT (R), AND LEFT (L). F and B moved the turtle a fixed distance (i.e., one "turtle step") forward and back, respectively, and R and L rotated it 90-degrees clockwise and anti-clockwise, respectively.

Problems. Children were asked to solve simple problems that involved getting the turtle to a goal. Set 1 consisted of eight two-command problems. The turtle appeared on the display screen in one of four orientations (see top panel of Figure 2). Then an object appeared one "turtle step" above, below, to the right of, or to the left of the turtle such that rotation and then displacement could get the turtle to the object (see bottom panel of Figure 2). Set 2 consisted of twelve two- and three-command problems (display screen version of problems used by Gregg (1978)). A row of three objects, spaced one "turtle step" apart, appeared on the display screen (see top panel of Figure 3). Then the turtle, oriented 0- or 180- degrees, appeared below one of the objects. Finally, one of the remaining two objects was designated as the goal object. Eight problems required rotation and then displacement of the turtle, and four required rotation and then two displacements of the turtle (see bottom panel of Figure 3).

A brief introduction preceded each problem set in order to present the problem scenario, familiarize the children with the distance covered by a "turtle step," and demonstrate alternative solutions to a problem (e.g., RF and LB). Children were asked to solve the problems in the "fastest way," that is, with the fewest possible commands.

Problem-solving conditions. Children solved the problems in Set 1 in each of three conditions, followed by the problems in Set 2. For each set, the problem-solving conditions were as follows. An immediate condition, administered first, provided on-line feedback from the screen. Children gave a command, the turtle executed it, and the next command was given and executed. An incorrect command could be immediately detected, and corrected or accommodated by the next command. The purpose of this condition was to ensure that children understood the task, and give them practice. A planning condition, administered second, provided no screen feedback. That is, commands were not executed by the turtle. Children were simply asked to give the command(s) they thought would solve the problem, and moved on to the next problem. Finally, a programming condition provided delayed feedback from the screen. This condition was comparable to a real programming environment. Commands were executed after the entire sequence was given. Children gave the command(s) they thought would solve the problem, and the turtle executed the given command(s). They were asked to modify an incorrect solution, using information from the screen to figure out where the "bug" was. The turtle then executed the modified solution.

Subjects. Thirty-two 4- and 5-year-olds (mean age 5-1) participated in the study (29 completed the entire sequence of tasks). None of the children had previous experience with turtle graphics or computer programming although many had used computers to play games.

Procedure. In an initial session, children were introduced to the turtle graphics environment. The introduction included demonstration and explanation of each command, with the turtle beginning in each orientation. External aids (e.g., a cardboard turtle, a toy turtle, and having children "play turtle") were used to facilitate understanding. Each subsequent session began with a brief reintroduction to the turtle and commands. Children solved the problems in Set 1, followed by those in Set 2.

Number of sessions varied from 3 to 6, with each session lasting about 30 minutes. Children typically spent about 10 minutes after a session playing around and drawing turtle designs.

Results

Immediate conditions. Although the immediate conditions were mainly for practice, children's performance in them provides a context for viewing their subsequent performance in the programming-type conditions. Performance was quite good in the two immediate conditions. Children gave "fastest way" solutions on 52% of the problems in Set 1, and 54% of the problems in Set 2 (chance level was about 6%). They eventually got the turtle to its goal in 97% and 99% of the problems in Sets 1 and 2, respectively, indicating that they understood the task at hand.

Planning conditions. Children's plans were classified as Correct (i.e., "fastest way"), Appropriate Form (e.g., RF instead of LF), Incomplete (e.g., L or F instead of LF), or Other (e.g., FL instead of LF). As shown in Table 1, children gave "fastest way" plans on 19% of the problems in Set 1, and 38% of the problems in Set 2. These percentages were well below those in the immediate conditions, indicating that children had difficulty giving a second (or third) command without the benefit of screen effects of the previous command(s). Indeed, using Klahr and Robinson's (1981) criterion of perfect performance, none of the children could give correct plans for even two-command problems.

For Set 1 plans, the most common error was to give only one of the two needed commands. This error raises doubts about young children's ability to give more than one turtle command in advance (i.e., generate a program). For the later Set 2 problems, however, the most common error was to give the wrong rotation-displacement command combination (e.g., RF instead of LF or RB). This error, presumably due to confusion of right and left, has less serious implications for computer programming.

Programming conditions. Classification of children's first programs (i.e., first-given solutions) is shown in Table 1. First programs for 68% and 60% of the problems in Sets 1 and 2, respectively, were incorrect and, thus, in need of debugging. Second programs (i.e., second-given solutions) are shown in Table 2. Children were able to debug 53% and 55% of their incorrect programs in Sets 1 and 2, respectively. Not surprisingly, programs than were already of appropriate form (e.g., a rotation-displacement command sequence for two-command problems) were easiest to fix. Programs that were incomplete were surprisingly difficult to fix.

Conclusions

The results provide a general picture of young children's computer programming-type abilities. Four- and 5-year-olds could not easily generate a two- or three-command program. Their ability to give the correct sequence, and at least the appropriate number and type of commands, improved in the course of the study. But even their final abilities were not very impressive. Similarly, their ability to debug their own incorrect programs was not as high as we might have expected, given the simplicity of the problems and the screen feedback and practice they had received in the immediate conditions. These young children could not easily go on to more complex turtle graphics programming.

The implication is that educators and parents need to develop realistic expectations and goals with regard to young children and computer programming. Young children's activities in turtle graphics may provide them with useful pre-programming experience and, perhaps, should be viewed and appreciated as just that. Indeed, pre-programming activities would seem to belong in a preschool curriculum along with pre-reading and pre-arithmetic activities.

References

Gregg, L. W. (1978). Spatial concepts, spatial names, and the development of exocentric representations. In R. S. Siegler (Ed.), Children's thinking: What develops? Hillsdale, NJ: Lawrence Erlbaum Associates.

Klahr, D., & Robinson, M. (1981). Formal assessment of problem-solving and planning processes in preschool children. Cognitive Psychology, 13, 113-148.

Table 1

Classification of Children's Solutions, Plans, and First Programs

Problem Set	Condition	No. of Subjects	Correct	Incorrect: Appropriate Form	Incorrect: Incomplete	Incorrect: Other
One	Immediate	32	52%			
	Planning	31	19%	23%	48%	9%
	Programming	30	32%	31%	31%	5%
Two	Immediate	30	54%			
	Planning	30	38%	36%	18%	7%
	Programming	29	40%	27%	17%	16%

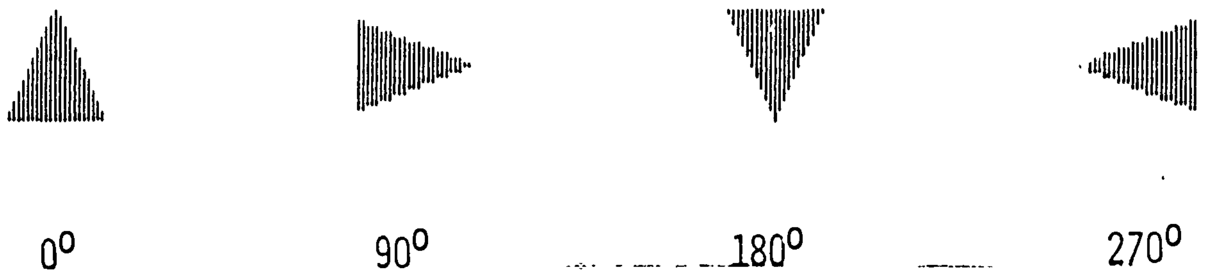

Table 2

Modification of Incorrect First Programs

Problem	First		Second Program			
Set	Program	Total	Correct	Appropriate Form	Incomplete	Other
One[a]	Appro. Form	75	64	9	1	1
	Incomplete	75	20	16	37	2
	Other	13	2	3	4	4
Two[b]	Appro. Form	93	70	12	4	7
	Incomplete	60	28	4	23	5
	Other	55	16	12	9	18

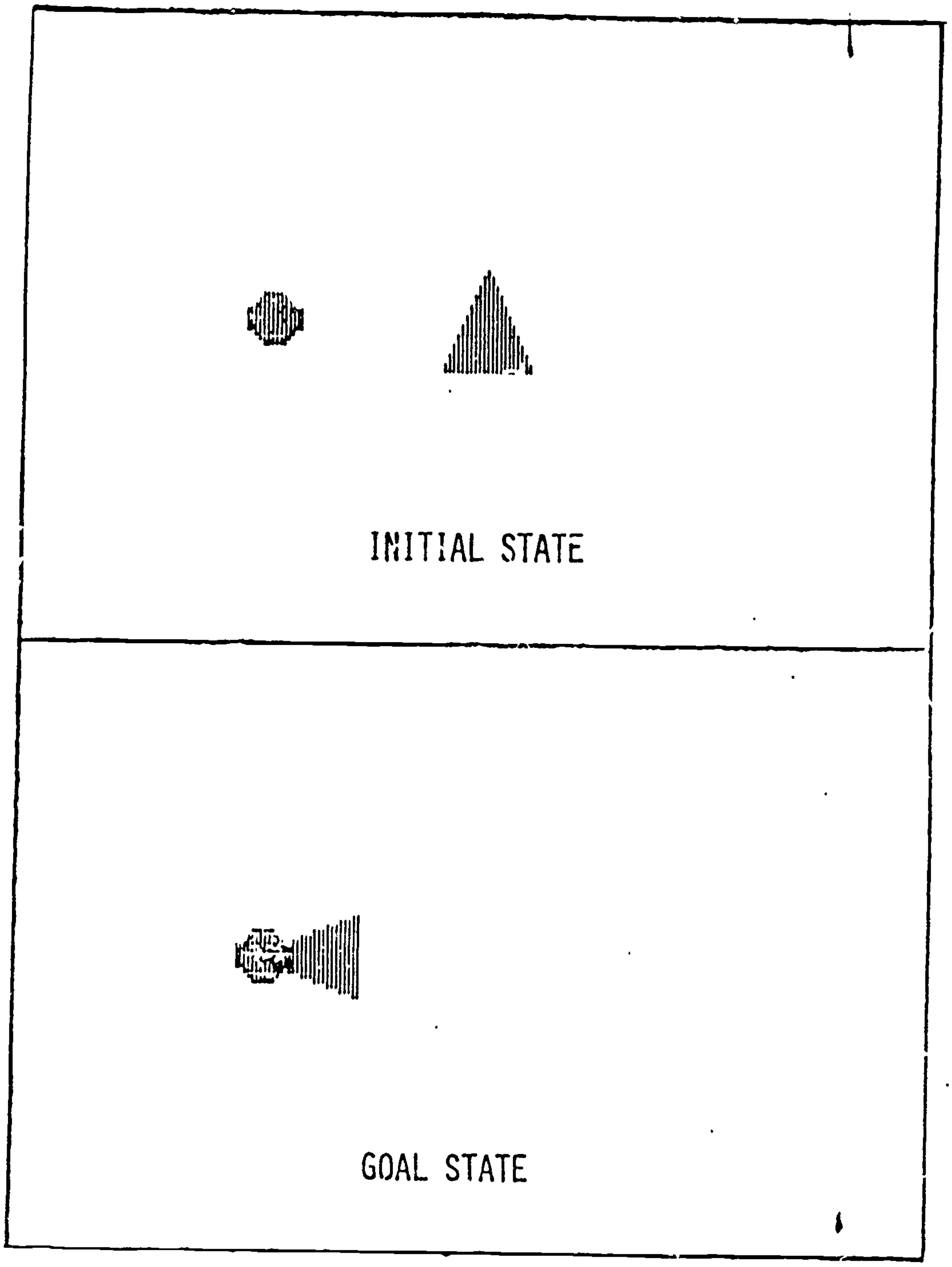

Figure 2

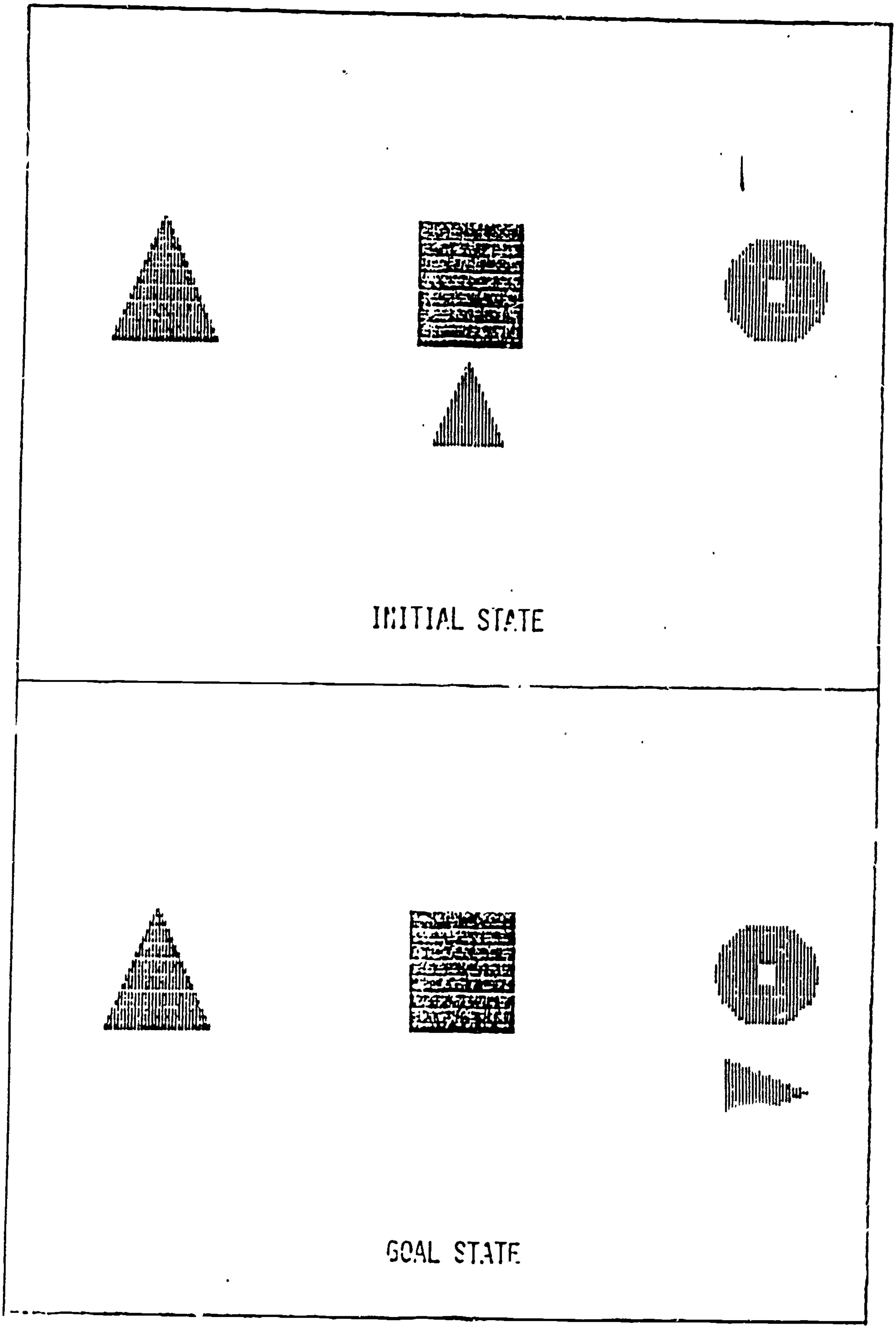

Figure 3