DOCUMENT RESUME

ED 279 311 IR 012 554

AUTHOR Kuschner, David
TITLE "This Computer Gives You a Hard Bargain": Is It Conflict or Frustration When Software Won't Let You Change Your Mind?
PUB DATE Apr 86
NOTE 17p.; Paper presented at the Annual Convention of the American Educational Research Association (San Francisco, CA, April 16-20, 1986).
PUB TYPE Reports - Evaluative/Feasibility (142) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Cognitive Development; *Computer Graphics; *Computer Software; *Creative Art; Kinetics; Media Research; Microcomputers; Preschool Education; *Responses; *Young Children
IDENTIFIERS Frustration; LOGO Programing Language

ABSTRACT

This study focused on the cognitive conflicts experienced by young children in using software programs that provided them with tools to create and/or combine individual graphic elements into larger structures. Six 5-year-old children, none with prior computer experience, were observed using three programs--Kids at Work, Picture Perfect, and Springboard. Through 12 sessions, each child spent 15 to 20 minutes per week interacting with the microcomputer and the same program. Observations revealed several types of conflict caused by software constraints when a child began to build a conceptual picture: only one of the programs (Kids at Work) allowed the children to rearrange what they had already done; none of the programs allowed the children to change the direction faced by individual graphics after they had been put in place; the children did not at first understand the "color fill" function of Picture Perfect and there was no way to correct their mistakes. These results suggest that the constraints of these particular software programs short-circuited the children's interest in and attention to that experience. It is concluded that if software for young children is to maximize their cognitive development, it should provide an opportunity for the child to exercise his intentions, reflect on the results of his actions, and subsequently revise these intentions/actions. The Logo programming language and software which emphasizes kinetic movement are cited as two types of software that might facilitate cognitive development. Fourteen references are provided. (BBM)

"This computer gives you a hard bargain": Is it conflict or frustration when software won't let you change your mind?

David Kuschner
Center for Teaching and Learning
University of North Dakota
Grand Forks, ND 58202

(Paper presented at the annual convention of the American Educational Research Association, San Francisco, April, 1986.)

Any medium designed for young children's activity carries with it certain inherent constraints on its use (Kuschner and Clark, 1976). It is not possible, for example, for a child to bend wooden blocks in order to form a curve in the road travelled by toy trucks. Playdoh, which is malleable, might work for making the curve in the road, but it is not a sturdy enough medium for building the bridge over which the trucks will ride. As a medium for young children, microcomputer software can also be described in terms of the inherent constraints on its use. One of these constraints, the degree to which children are able to change their minds and revise their own actions, is the focus of this paper.

The potential for revising one's actions is directly related to the experience of cognitive conflict. There are two types of cognitive conflict, both of which relate to a child's original efforts at understanding and acting on some aspect of his experience. The first type occurs when the child expects a particular result from his activity and then experiences the unexpected, or the discrepant event. In other words, he made a prediction which did not come true. The second type of cognitive conflict occurs when the child experiences two or more competing and contradictory explanations or understandings for the same event. In either case, the discrepant event or felt contradiction, by its very essence, suggests to the child that there is an alternative way of understanding his present experience. This alternative understanding can then lead to additional "testing" of ideas. In effect, previous actions are revised in an attempt to reconcile original ideas with the results of action based on those ideas. The ultimate importance of cognitive conflict is, of course, the revision of internal, mental structures and subsequent mental activity, but the ability to overtly act on ideas and their alternatives is vitally important in terms of the feedback such activity provides.

If computer software is to be considered a viable medium for fostering the cognitive development of young children, then the issue of the child's control of the software must be examined (Cuffaro, 1984). Revision of actions is certainly an important form of control. In an earlier study (Kuschner, 1985) I analyzed nine educational software programs created for young children in order to describe the variety of ways software does and does not facilitate the revision process. All of these programs were constructive in nature. In other words, children could use the tools provided by the software to create and/or combine individual graphic elements into larger structures. The results of that study suggested that there was no one software program which provided optimal control for a child in terms of revision. Much like the variations of block media that exist (e.g., Legos and Bristle Blocks), the software represented a variety of ways for revising actions. The purpose of the present study was to take a sample of these programs and observe children using them, with an eye towards documenting what happens when a child wants to revise an action--wants to change his mind--but is not able to do so because of software constraints. Because of these constraints, there may be a fine line between an event producing the experience of cognitive conflict and that same event producing an experience of frustration. That fine line, at least in part, may be directly related to the possibility of a child revising his own actions.

Procedures of the Study

Six children, each five years of age and attending a university day care center, were the subjects of this study. Three software programs were used, two children assigned to each program. On a weekly basis, each child was taken to a small room adjacent to the main space of the center where he or she spent 15 to 20 minutes interacting with the computer and software. In order to allow time for familiarity and competence with the software program to develop, each child worked with the same program for the duration of the study. Although illness and family moves reduced the number of weekly sessions for three of the children, there was an average of 12 sessions for each of the six children.

After providing initial instruction and guidance on the use of the computer and the operation of the software, I primarily assumed the role of observer and recorder. I answered any questions the children had, offered suggestions if a child was having difficulty performing a particular function, and made suggestions as to what a child might try to do if I thought it relevant to the child's own intentions. When a child was experiencing a problem with revision of his or her actions, I provided leading suggestions up to the point of offering the exact solution, if in fact there was one. In addition, I provided potentially reinforcing and motivating support through my attention and verbal comments about what the children were doing.

At the beginning of each session, I gave the children a printout of what they had produced the session before. These printouts served two purposes. One, they represent part of my data base for the study, and two, they served as a form of motivational reward for participating in the study. This seemed to be effective because the children invariably asked to see their "picture" from the week before as soon as we entered the room which housed the computer. The children were also given the option of beginning each week's session by continuing work with the previous week's product, which had been saved to disk. The printout was an additional reminder of what they had done the week before.

The Software

Three software programs were used in the study: Kids at Work (Scholastic), Picture Perfect (Methods and Solutions), and Stickers (Springboard). In each case, the software is marketed as having the potential for fostering children's intellectual growth and creative development. In addition, these programs were selected for use in this study because of their constructive nature: children can use the tools provided by the software to combine various elements into two-dimensional constructions of some kind. Two of the programs, Kids at Work and Stickers are graphics-only programs, while Picture Perfect supports the combining of text with graphics. A description of each program, in particular the characteristics relevant to this study, follows.

Kids at Work. This is a graphics-only program, one which provides the child with two sets of predrawn pictures, each set containing individual graphics, for example, buildings, farm machinery, and farm animals. By depressing one of four keys, a construction worker (City Shapes with 34 graphics) or a farm girl (Country Shapes with 35 graphics) is moved from a blank screen into these sets for the purpose of selecting individual graphics. (The screen appearance of these characters mimics the movements of walking and running.) Once the character is positioned next to the desired graphic, depressing the space bar raises the character's arm, which represents the "grabbing" of the graphic. The child can then move this character, with graphic, back to the blank screen and by depressing the space bar, release and position the graphic wherever he wants. The number of elements he combines into his picture is limited only by the amount of space on the screen.

In terms of this study, the characteristics of this program which are important to note include the following. The position of individual graphics can be changed at any time, including after loading a previously saved creation. A particular graphic can be selected for use as many times as the child wants, and a previously placed graphic can be removed from the scene at any time. Modif'cation of individual graphics, including color change, is possible, but the skills necessary to do so proved to be beyond the abilities of these children.

<u>Stickers</u>. With this graphics-only program, children can place geometric shapes of different sizes and colors onto a blank screen, simulating the act of placing stickers on a piece of paper. Through the use of keyboard or joystick input, the child selects a sticker from a set which appears at the bottom of the screen. By depressing the space bar of joystick button, the sticker is "grasped" and can then be moved around the screen until a position

for it has been chosen, at which point the space bar or joystick button releases it. There are also menu options which allow the child to change the color of the stickers and to rotate their orientation. These changes must, however, be accomplished prior to the placement of a particular sticker on the screen. Once a sticker has been placed, it can not be directly changed in any way. (Previously placed stickers can be changed by superimposing other stickers over them.) As was the case with Kids at Work, it is possible to modify individual stickers in terms of their structure, but this again proved to be beyond the abilities of the children studied.

Picture Perfect. This program provides children with the ability to create pictures from scratch through the use of drawing tools, select and modify predrawn pictures, and add text to what they create. Once a drawing is made or a picture positioned, it can be filled in with color and/or modified by erasing part of it or adding to it with the drawing tools. Individual elements of a picture, however, can not be manipulated as wholes once placed on the screen.

What makes this program somewhat unique is the "rubber-band" function that is part of the drawing tools and the selection of predrawn pictures. Each of the 72 available pictures can be placed anywhere on the screen and then "stretched" to a desired size. Once that size is determined, depressing an input button freezes the picture at that size. From that point on, size is no longer open to manipulation. This "rubber-band" function is also part of the tools for drawing individual lines, connected lines, and boxes, with the same limitations applying.

Results of the Study

In order to understand the type of conflict experienced by the children in this study it is necessary to first describe the typical kind of activity in which they engaged. This was the first computer experience for all of the children involved. They approached this material as they would any new toy--they played with in and attempted to see what they could do. At first they experimented with the various functions, and their initial products are characterized by a randomness of both graphic selection and placement. Graphics were placed without any sense of pattern or form, and the children seemed to be fascinated by the "magical" quality of the computer, i.e., you push a button and something appears on the screen.

After a few sessions, however, all the children became more deliberate in their activity. They paused before selecting graphics, and then they would move the graphic around the screen before deciding on a position. Rather than a random collection of graphics, they began to build conceptual pictures, sometimes with forethought and sometimes as a result of an accidental placement of a graphic. This was when the first examples of conflict occurred.

As Piaget noted in his descriptions of early development, fortuitous events often serve as the motivator for children's activity. This was the case with the children's computer-generated constructions. An example of a child's work with Stickers serves as a good illustration. At first the child was content with simply stamping different stickers on the screen, without any apparent pattern or design in mind. During the fourth session, this random stamping of stickers resulted in two circles being placed below a horizontally oriented rectangle. At that point, the child indicated that this configuration looked like a car. He began to make car noises and added other stickers to this picture. This also proved to be the first instance of his becoming frustrated by the constraints of the software. One of the circles was not exactly touching the rectangular body of the car. The child attempted to move it by bringing the cursor to the circle and depressing the button of the joystick, as if to regrasp the sticker. He was clearly disappointed when this did not work, looking to me for help and asking why the sticker would not move when he pressed the joystick button.

There were additional examples of this same problem for the children using Stickers and Picture Perfect. Once they became intrigued by the results of their actions and what appeared on the screen, they wanted to rearrange some of what they had already done in order to elaborate on what had caught their eye. These two programs do not allow for such revision. Judging by their expressions, actions, and questions, they found this to be frustrating. An analysis of the productions they created suggests that it also limited the amount of conceptual and theme-bound pictures that they made. The children who worked with Kids at Work had a different experience. The predrawn graphics built into this program can be manipulated (moved and positioned) at any time. The child, for example, who first placed the graphic of a fence, and then went back to get the graphic of the cow, was able to rearrange his picture in order to produce the appearance of the cow being behind the fence. The software allowed him to regrasp the fence, move it out of the way, place the cow where the fence had been, and finally reposition the fence in front of the cow. The idea for combining the two graphics in this way did not occur to the child until he had chosen the fence and cow as separate graphics and saw them within the same visual field on the screen. The results of his actions fed back into the planning of the actions. In this case the software allowed the child to "go back into" the scene and rearrange the pieces. This type of reflection followed by revision was not facilitated by the other two software programs. This is possibly one of the reasons why the children working with the program Kids at Work created a greater number of conceptually organized pictures.

A second type of frustration for the children involved the orientation of individual graphics. As the children constructed their pictures, there were a number of instances when they wanted an individual graphic to be facing in a different direction; for example, a truck facing east to be turned west. In the case of Stickers, there was a rotation function which could be used to rotate all of the stickers 90 degrees. This proved helpful in a limited way, because the rotation function had been engaged prior to the selection and placement of a sticker. That kind of pre-planning on the part of the children was not observed very often. When something occurred to them concerning the orientation of a sticker after the placement of that sticker, however, they could not do anything about it. Since the other two programs were the ones which contained realistic graphics, the children were even more frustrated by these programs. As opposed to the manipulation of concrete, three-dimensional models or the free-hand drawing of two-dimensional pictures, these two-dimensional computer representations were set in pre-determined and fixed orientations.

A third type of frustration was specifically generated by the program, Picture Perfect. As part of its collection of drawing tools, this program provided what is called a "color fill" function. This function, which is found in many graphics programs, allows the child to choose a color and then fill any closed figure on the screen. The word 'closed' is emphasized because this function is often a source of surprising experience for children. They are able to master the rudiments of the operation--selecting the color and placing the cursor in the area to be filled--but they do not at first understand that the color will "leak" out of the area if there is any opening, and will fill as much of the screen as is open to it. This happened a number of times to the children using Picture Perfect. The experience of the color leaking was actually enjoyable; they found it to be aesthetically pleasing and somewhat of a challenge to figure out why it was happening. The constraints of the software program, however, limited the degree to which exploration of the phenomenon would be productive. The leaking color fill could be stopped by the depression of the joystick button, but there is no way to "undo" what had already happened. There is an erase function, but it is cumbersome and inexact in its operation, and there is no way to "cover" the fill with the original color of the background in order to effectively return the picture on the screen to its original state. In other words, when the accidental occurrence took place, it was interesting and motivating to the children, but there was very little they could do in order to explore what had happened. They couldn't for example, simply erase and redo the fill in order to study what had taken place, nor could they undo the effect of the fill, change something about their picture, and try it again. The nature of the color fill function has the potential for providing children with a graphic representation of the difference between closed and open geometric figures; a representation that is highlighted by what is often a discrepant event, i.e., the color leaking out of the intended area to be filled. The results of this study, however, suggest that the constraints of this particular software program short-circuited the children's interest in and attention to that experience. When they discovered that they were unable to take the information provided by the feedback and "try again," they lost interest in the event.

Discussion

The introduction of microcomputers into early childhood education is not at this point a totally accepted proposition (Barnes and Hill, 1983; Cuffaro, 1984). Where there might be a favorable attitude towards the use of the technology with young children, there is concern about the software that exists (Chaillé and Littman, 1985). Zajonc (1984) voices this concern about software for young children quite strongly when he writes: "Simply by pressing a button the child can transform his visual field at will. The use of language here is intentional, for there is clearly [no will] involved. The child is a passive doodler in such a situation, captivated by the image it apparently creates. But there is no creation here either. All these activities have been usurped by the machine" (p. 575). At least two of the three programs examined in this study put the child in the position of being a "passive doodler."

What, then, might be the characteristics of good software for young children? I propose that software should provide the opportunity for a child to exercise his intentions, reflect on the results of his actions, and subsequently revise these intentions and actions. Such software is likely to maximize the possibility for the child to experience the kind of cognitive conflict which is related to cognitive development in general.

The importance of facilitating the exercise of intention is discussed by Chaillé and Littman (1985) when they write that there are two types of problem solving: theory-oriented and success-oriented. The authors suggest that theory-oriented problem solving encourages a child to focus on the relationships, processes, and principles involved in his experience. Success-oriented problem solving, on the other hand, tends to focus a child's attention on goals and end points. Intelligence, according to the authors, is fostered by engaging in theory-oriented problem solving.

Glick (1983) relates the exercise of intention to Piaget's concept of assimilation. Glick writes that assimilation is comprised of two parts: patterned actions or schemes, and intention. He suggests that when we think about the process of assimilation, our focus is too often on the patterned actions involved as opposed to the ideas or theories which activate those schemas of action.

The children in this study seemed to get "stuck" at the level of repeating patterned actions. When pressing one button or one key makes something interesting happen on the screen, creates that captivating image Zajonc (1984) refers to, it is easy to be seduced into simply pushing that button over and over again. It is easy for intention to become subordinate to what the machine can do, particularly when intention becomes thwarted by what the computer <u>can't</u> do. The child's focus fixates on the success of his actions as opposed to a theory behind the actions. This also relates to Barnes and Hill's (1983) concern that the precision of action required by the microcomputer is perhaps antithetical to reflective thinking, and to Streibel's (1984) concern that most educational computer programs emphasize immediate, non-reflective, action-based decision-making.

Two types of software which might facilitate the exercise of intention and the subsequent reflection include the LOGO programming language pioneered by Papert (1980) and software which emphasizes kinetic movement (Forman, 1984; 1985). The essence of LOGO consists of the execution of a plan, receiving feedback, and then engaging in the debugging process. The LOGO user is continually generating theories as to how to create desired effects on the screen. Very often these theories do not work as the child intended them to, which leads the child into the debugging process. Papert points out that debugging--reflecting upon and analyzing the step-by-step workings of a program and then making revisions--is not the same as erasing, which is the primary way the software in the present study allowed the children to revise their actions. LOGO is an example of software which both encourages children to engage in theory-oriented problem-solving and also provides them with the means for using the results of their actions as the material for productive reflection. Results of a study by Clements and Gullo (1984) suggest that experience with programming in the LOGO language does in fact promote reflectivity.

Regarding the second type of software that has the potential for fostering cognitive conflict, Sheingold (1984) points out that one of the unique properties of the microcomputer is its power to represent dynamic movement. Software which has the child program the movement of images on the monitor screen would engage the child in the act of making predictions based upon his intentions, provide him with feedback regarding the results of his actions, and open up the opportunity for the child to experience the conflict that occurs when the results contradict his intentions. The cycle would then be completed if the software allowed the child to revise his original plan. Forman (1984) suggests that the power of this feedback is enhanced if the child is presented with images representing the path of the movement itself, not simply the starting and arrival points of the movement. With this information, the child can reflect more completely on the relationship between his intention and the results of his actions.

In summary, I would like to cite a statement from an article by Jeanne Bamberger entitled, "The computer as sandcastle" (Bamberger, 1983). Bamberger writes that the medium of computer software, like any good medium for children's activity, should foster what she calls conversational learning. Her definition of this concept embodies the three characteristics of intention, reflection, and revision just discussed. "By conversation I mean the conversation we have with materials as we build or fix or invent. As we perturb these materials, arranging them and rearranging them, watching them take shape even as we shape them, we learn. The stuff talks back to us, remaking our ideas of what is possible. The backtalk leads to new actions on our material objects in a spiral of inner and outer activity. Inner intention gives way to reflection on and responsiveness to the backtalk of the materials, leading to new outer actions on objects, and hence once more to changed intention" (p. 35).

References

Bamberger, J. (1983). The computer as sand castle. In Chameleon in the classroom: Developing roles for computers. Technical report No. 22. New York: Bank Street College of Education. (ERIC Document Reproduction Service No. ED 249 921)

Barnes, B.J., & Hill, S. (May 1983). Should young children work with microcomputers? - LOGO before LEGO? The Computing Teacher, 10(9), 11-14.

Chaillé, C., & Littman, B. (June 1985). Computers in early education: The child as theory builder. In E.L. Klein (Ed.), Children and computers. New Directions for Child Development, no. 28. San Francisco: Jossey-Bass.

Clements, D.H., & Gullo, D.F. (December 1984). Effects of computer programming on young children's cognition. Journal of Educational Psychology, 76, 1051-8.

Cuffaro, H. (Summer 1984). Microcomputers in education: Wh' is earlier better? Teachers College Record, 85(1), 559-568.

Forman, G.E. (August 1948). Computer graphics as a medium for enhancing reflective thinking in young children. Paper presented at the annual Conference on Thinking, Harvard University, Cambridge, MA.

Forman, G.E. (June 1985). The value of kinetic print in computer graphics for young children. In E.L. Klein (Ed.), Children and computers. New Directions for Child Development, no. 28. San Francisco: Jossey-Bass.

Glick, J. (1983). Discussion. In Chameleon in the classroom: Developing roles for computers. Technical report No. 22. New York: Bank Street College of Education. (Eric Document Reproduction Service No. ED 249 921)

Kuschner, D. (June 1985). A study of the possibilities for reversible actions in software for young children. Paper presented at the Fifteenth Annual Symposium of the Jean Piaget Society, Philadelphia, Pennsylvania.

Kuschner, D., & Clark, P. (1977). Children, materials, and adults in early learning settings. In L. Golubchick and B. Persky (Eds.), Early childhood education. Wayne, NJ: Avery.

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

Sheingold, K. (1984). The microcomputer as a medium for young children. Technical report No. 26. New York: Bank Street College of Education. (ERIC Document Reproduction Service No. ED 249 923)

Streibel, M.J. (1984). An analysis of the theoretical foundations for the use of microcomputers in early childhood education. (ERIC Document Reproduction Service No. ED 249 971)

Zajonc, A.G. (Summer 1984). Computer pedagogy? Questions concerning the educational technology. Teachers College Record, 85(4), 569-577.