Drag-and-Drop versus Point-and-Click Mouse Interaction Styles for Children

KORI M. INKPEN
Dalhousie University

This research investigates children's use of two common mouse interaction styles, drag-and-drop and point-and-click, to determine whether the choice of interaction style impacts children's performance in interactive learning environments. The interaction styles were experimentally compared to determine if either method was superior to the other in terms of speed, error rate, or user preference, for children. The two interaction styles were also compared based on children's achievement and motivation, within a commercial software environment. Experiment I used an interactive learning environment as children played two versions of an educational puzzle-solving game, each version utilizing a different mouse interaction style; Experiment II used a mouse-controlled software environment modeled after the educational game. The results were similar to previous results reported for adults: the point-and-click interaction style was faster; fewer errors were committed using it; and it was preferred over the drag-and-drop interaction style. Within the context of the puzzle-solving game, the children solved significantly fewer puzzles, and they were less motivated using the version that utilized a drag-and-drop interaction style as compared to the version that utilized a point-and-click interaction style. These results were also explored through the use of state-transition diagrams and GOMS models, both of which supported the experimental data gathered.

Categories and Subject Descriptors: D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User interfaces*; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human factors*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Input devices and strategies*; *Interaction styles*; *Evaluation/methodology*; I.3.6 [**Computer Graphics**]: Methodology and Techniques—*Interaction techniques*; K.3.1 [**Computers and Education**]: Computer Uses in Education

General Terms: Experimentation, Human Factors, Measurement, Performance

Additional Key Words and Phrases: Interface design, input techniques, computers in education, drag-and-drop, point-and-click, interaction styles, children, gender, mouse interaction, electronic games

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, the TeleLearning NCE, and the Media and Graphics Interdisciplinary Centre at the University of British Columbia.
Author's address: EDGE Lab, Faculty of Computer Science, Dalhousie University, Halifax, Nova Scotia B3H 1W5, Canada; email: inkpen@cs.dal.ca.

1. INTRODUCTION

Many children today are exposed to computers from a very early age. The types of interaction techniques used in children's software are quite varied. In general, children appear to adapt to whatever interaction style is present, but this is not an ideal situation because some children may be apprehensive using an interaction style with which they are not comfortable. In order to design effective interaction techniques, "we need to use a deeper understanding of task, device, and the interrelationship between task and device from the perspective of the user" [Jacob et al. 1994]. Paying attention to the perspective of the user means that we cannot assume that children are just like adults. There has been significant research on interaction styles for adults, but until recently very little research has focused on children's interactions with computers; "most of [computer] technology has not been leveraged to make the child-computer interaction optimal" [Hanna et al. 1999]. It is important that user interface issues are investigated for children.

In an attempt to understand children's interactions, one focus of our research has been children's activities using the commercial computer game *The Incredible Machine* which is described more fully in Section 2. In earlier work [Inkpen et al. 1995], children were observed as they played in various collaborative arrangements. IBM-compatible computers were utilized in the initial phase of this study while in a larger, follow-up phase, Macintosh LCIII computers were utilized. *The Incredible Machine* software is available on both platforms, and on the surface, the IBM-compatible and the Macintosh versions look identical. However, closer examination revealed that the IBM-compatible version utilizes a point-and-click mouse interaction style, whereas the Macintosh version of the game utilizes a drag-and-drop mouse interaction style. In the follow-up phase of the study we observed that many children were having difficulty using the drag-and-drop interaction style on the Macintosh version. In addition, the number of puzzles that girls in particular were able to solve while playing the game was fewer than what children achieved during the preliminary phase of the study. While the differences in mouse interaction style certainly were not the only factors that might have contributed to these differences, we decided to perform an additional study, consisting of two experiments, that would attempt to determine whether mouse interaction style has a significant effect on children's performance and motivation. This paper describes that general study.

The remainder of this introductory section provides a theoretical justification for the study, references to the previous literature related to the study, a brief description of the research setting in which the study took place, and an overview of the study. Section 2 covers the features of *The Incredible Machine* that are important for understanding the current study. The two experiments of the study are described and their results discussed in Sections 3 and 4, respectively. The paper concludes in Section 5 with some remarks on the lessons learned from the study that apply to the design of children's mouse-based interactive learning environments. Suggestions are then made for future research questions that should be investigated.

1.1 Theoretical Justification

Looking at computers that are commonly found in schools today, we see machines that have been designed for, and used in, the workplace for many years. These computers have not been designed with the interactions of children in mind nor with the goal of supporting learning [Inkpen 1997; Druin 1999]. Researchers have noticed that some prominent user interface styles designed for adults may not be appropriate for children.

Berkovitz [1994] found that children had difficulty selecting groups of objects using a marquee-type selection (outlining a group of objects to be selected with an imaginary rectangle) because of difficulty choosing the initial corner of the selection rectangle. Strommen [1994] observed that children had difficulty maintaining pressure on the mouse button for extended periods. Numerous other researchers have observed that gender differences often exist with respect to interactions with computers. For example, girls and boys think about computers differently [Hall and Cooper 1991; Wilder et al. 1985], have different motivations for using computers [Inkpen et al. 1994; Upitis and Koch 1996], and have different preferences and usage styles [Lockheed 1985; Inkpen et al. 1994; Lawry et al. 1995]. Previous research has also shown that software designers incorporate their own gender biases into the software that they develop [Huff and Cooper 1987]. As a result, it is important that research on children's interactions with computers in educational environments be sensitive to gender differences such as these.

1.2 Previous Literature

Adults' use of the two mouse interaction styles of drag-and-drop and point-and-click have been explored by other researchers who examined the differences in speed and accuracy between the two methods on various tasks [Boritz et al. 1991; Gillan et al. 1990; MacKenzie 1992a; 1992b; 1991]. The MacKenzie et al. [1991] study concluded that a dragging task was slower than a pointing task and that more errors were committed during a dragging task than during a pointing task. We decided to reexamine the issue of mouse interaction style for three reasons: (1) direct-manipulation interfaces are more prevalent today than was the case in 1991, and therefore user experiences may have changed; (2) the previous research dealt only with adults, and it cannot be assumed that the results will be the same for children; and (3) gender was not examined at all in the previous research. While many other input devices such as pen-based input are becoming common in the workplace, mouse-based input is still the dominant interaction device used by children at both school and home. In order for this research to have an immediate impact on developers of educational

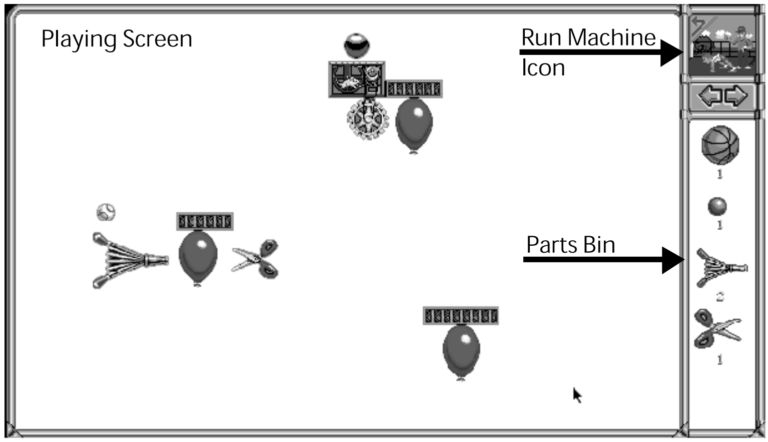

Fig. 1. The initial screen for a puzzle from the game *The Incredible Machine*. The goal is to break all of the balloons on the screen.

software, mouse-based input was the method of interaction explored in this study.

1.3 Research Setting

The research for the study described in this paper took place at Science World, in Vancouver, British Columbia, Canada, over the summers of 1994 and 1995, involving several hundred children. Science World is an interactive science museum, where children and adults explore various science concepts through hands-on activities and experimentation.

The research area for both experiments included a computer set up on a table in an open area, which was chosen for its visibility to visitors exploring other exhibits. The research area was set back slightly from the main traffic and sectioned off with small risers to provide some privacy for children taking part in the study. Children were positioned with their backs to the passers-by, which helped to reduce distractions during the study.

2. *THE INCREDIBLE MACHINE*

The Incredible Machine[1] is a commercial puzzle-solving computer game produced by Sierra On-Line, Inc. that invites players to construct "Rube Goldberg"-style machines [Marzio 1973] to achieve particular goals comprising a puzzle. A typical initial screen state for *The Incredible Machine* is shown in Figure 1. The screen is made up of three different sections: the playing screen; the parts bin; and the run machine icon, which when pressed starts the machine. Playing the game (solving a puzzle) involves moving objects from the parts bin onto the playing screen, configuring the objects among the objects already on the playing screen to achieve the goals

[1]The Incredible Machine^{TM} © Sierra On-Line Inc.

required for that puzzle. When a player has a configuration of objects to test as a solution to the puzzle, the machine can be run to see if it performs the right actions to achieve the required goals. If the configuration successfully achieves the goals, a small dialogue box appears in the middle of the playing screen, congratulating the player for solving the puzzle and inviting the player to advance to the next puzzle. If the configuration does not successfully achieve the goals, no dialogue box appears, and then the player is required to press a mouse button to stop the machine. Stopping the machine results in all of the configured objects being returned to the positions they occupied before the machine was run and permits the player to reconfigure objects or add new objects from the parts bin before running the machine again.[2]

The main purpose of *The Incredible Machine*, at least as we use it in an interactive learning environment, is to provide children an opportunity to enhance creative problem-solving skills. Reasoning by analogy and drawing on commonsense knowledge about everyday objects, players are expected to "solve" the rest of a puzzle by figuring out how to configure some or all of the objects in the parts bin to completely achieve the required goal.

The remainder of this section discusses the differences between the user interface for the IBM-compatible version and the Macintosh version of *The Incredible Machine* that are important for our study. Further information on *The Incredible Machine* is provided in the instruction manual and the Hint Guide for the game.

2.1 Moving Regular Objects

In the IBM-compatible version of *The Incredible Machine*, which uses a point-and-click interaction style, most objects are moved by clicking once on the object in the parts bin to pick it up (a click refers to pressing the mouse button down and then releasing), moving the cursor to its desired position on the playing screen, and finally clicking the mouse button again to place the object at that position on the playing screen. Once an object has been picked up, the object disappears from the parts bin, and an iconified picture of the object is then attached to the cursor to provide visual feedback confirming that the object is picked up. While this method of feedback differs from the traditional method of displaying the actual object, or an outline of the object, this difference does not appear to affect the usability of the software.

In the Macintosh version of *The Incredible Machine*, most objects are moved by pressing the mouse button down on the object in the parts bin to pick it up (the mouse button is not released at this stage), dragging the object to the desired position on the playing screen, and releasing the mouse button to place the object. The visual feedback provided is identical to that provided in the point-and-click interaction style, but additional

[2]For a more detailed description of *The Incredible Machine* see www.acm.org/pubs/citations/journals/tochi/2001-8-1/p1-inkpen/

kinesthetic (proprioceptive) feedback is provided by the muscular tension required to keep the mouse button down while dragging.

Using either of the two techniques just described, if an object is dropped somewhere other than where it was intended, so long as the position on the screen is not occupied by another object, the object will be placed at that position; but if the position on the screen is occupied, an error occurs. Using the point-and-click interaction style in the IBM-compatible version, the object will not be placed when an error occurs, but the object will remain picked up, attached to the cursor and ready for a second attempt at placing the object. However, using the drag-and-drop interaction style in the Macintosh version, once the mouse button has been released it cannot be rereleased to redo the placement part of the action. In this version of the game the object is placed back in its original position in the parts bin when a drop error is detected. To attempt the placement again, the player must return to the parts bin, pick up the object again, and move it back to the position where it is to be placed.

2.2 Moving Connector Objects

In addition to regular objects, *The Incredible Machine* has special objects called connector objects, such as elastics and ropes, which are used to connect two or more objects together. The main difference between connector objects and regular objects is that a connector object needs to be attached in two or more different places to other objects.

Connector objects are moved using the point-and-click interaction style in the IBM-compatible version of *The Incredible Machine* in a manner similar to that used for regular objects. For a connector object that needs to be attached to two connection points, such as an elastic, (1) the mouse button is clicked once on the connector object in the parts bin to pick it up, (2) the connector object is then moved to the first attachment point, (3) the mouse button is clicked to attach the first end of the connector object, (4) the connector object is then moved to the second attachment point, and finally (5) the mouse button is clicked again to connect the second end of the connector object.

In contrast, the drag-and-drop interaction style in the Macintosh version of *The Incredible Machine* moves connector objects by first clicking on the connector object in the parts bin to pick it up. Because the pickup action for the connector object is a click, the mouse button is released immediately. If a drag is attempted at this stage (if the mouse is moved without releasing the button) an error occurs, and then the connector object returns to its original position in the parts bin. Once the connector object has been successfully picked up, the cursor is moved to the first attachment point. From the first attachment point to the second, a drag-and-drop motion is performed to attach the connector object. The mouse button is pressed down on the first attachment point to attach the connector object; then the cursor is moved to the second attachment point where the mouse button is released to make the second attachment.

Connector objects can only be attached to specific objects. Using either of the two techniques described above, if either attachment point for the connector object is an incorrect position (not on a valid object or violating a constraint), an error occurs. Using the point-and-click interaction style in the IBM-compatible version, if an error occurs the end of the connector object will not be placed, but the connector object will remain picked up, attached to the cursor and ready for a second attempt at making the attachment. Using the drag-and-drop interaction style in the Macintosh version, if either attachment point is an incorrect position, an error results, and the connector object is placed back in its original position in the parts bin. Thus, in the Macintosh version, to attempt to reconnect the objects the player must return to the parts bin and pick up the connector object again and move to each of the attachment points again. Not only is the penalty for an error greater for drag-and-drop than for point-and-click, it is even more severe if it happens after the first attachment is successful because both the incorrect second attachment and the correct first attachment will have to be performed again with the drag-and-drop interaction style. This problem is frequently encountered when playing the game and appears to cause a great deal of frustration.

3. EXPERIMENT I: MOUSE INTERACTION IN *THE INCREDIBLE MACHINE*

Experiment I examined children using a version of *The Incredible Machine* that utilized either a point-and-click interaction style or a drag-and-drop interaction style. The goal of this experiment was to determine the impact these interaction styles have on children's achievement and motivation.

3.1 Method

3.1.1 *Participants*. The participants in Experiment I were 189 girls between the ages of 9 and 13. The experiment took place at Science World during the month of August, 1995. All participants were visitors to Science World who volunteered to take part in the experiment, and who had never played the computer game *The Incredible Machine*. The children were informed that the experiment would take approximately 30 minutes, and consent forms were signed by a parent or guardian.

Because previous research has found that gender differences often exist with respect to children's interactions with computers it was important that our investigation be sensitive to the possibility of gender differences [Inkpen 1997; Upitis and Koch 1996; Hall and Cooper 1991; Lockheed 1985; Wilder et al. 1985]. Due to a limited amount of time available at Science World we chose to use participants from only one gender. Previous observations indicated that girls in particular had difficulty using a drag-and-drop interaction style as opposed to a point-and-click interaction style; therefore, girls' interactions were the focus of this experiment.

The age range of 9 to 13 was chosen because previous observations had shown, that although children of all ages play and enjoy *The Incredible Machine*, in general, children under the age of 9 have difficulty learning and playing the game without assistance. In addition, to limit the impact that age would have on the results, we chose to keep the age range small with 13 as the upper bound.

One hundred and fifty-five girls were observed using the Macintosh version of *The Incredible Machine* game, while the other 34 girls were observed using the IBM-compatible version of *The Incredible Machine*. The reason for the difference in the sample sizes was that the data for the girls playing the Macintosh version of the game were also being used for a separate study that required a larger sample size [Inkpen et al. 1995].

3.1.2 *Hardware and Software*. The two platforms used were a 386 IBM-compatible computer running Windows 3.1 with a three-button mouse and a Macintosh LCIII computer, with a one-button mouse. Color 14-inch monitors were used with both the PC and the Macintosh. Because *The Incredible Machine* is not a time-critical game, no visible differences in system performance were observed between the two platforms. As the game utilized only one mouse button, whether or not a one or three-button mouse was used should not have impacted the results. Because operation of the game only required input from the mouse, the keyboard was not presented to the children. Instead, the mouse and mouse-pad were placed in front of the monitor, and the girls were able to place the mouse and mouse-pad wherever they felt comfortable using them.

3.1.3 *Procedure*. The girls were randomly assigned to a particular interaction style if both platforms were free; otherwise they were assigned to use whichever platform was available. The sessions began with welcoming remarks from the researcher, followed by a brief verbal introduction to the experiment and to *The Incredible Machine*, a short interface training session, and then 30 minutes of time to play *The Incredible Machine*.

In order to reduce user interface problems during the session, the interface training session was designed to teach the girls how to manipulate objects in the game using one of the interaction styles. The interface training session demonstrated how to begin playing a puzzle, how to move objects from the toolbox onto the playing screen, how to connect objects together, how to flip objects, how to resize objects, and how to run the configured machine by clicking on the run machine icon.

Following the interface training session, the girls were asked to complete as many puzzles as they could in the 30 minutes provided. Upon completion of a puzzle, the girls were required to record the time at which they finished, and the game then automatically advanced to the next puzzle. The girls were told that they could stop playing at any time they wished.

3.1.4 *Experimental Variables*. One independent variable was manipulated in this experiment: mouse interaction style. The girls played using a version of *The Incredible Machine* that utilized either a point-and-click interaction style or a drag-and-drop interaction style. The puzzles to be solved for both versions of the game were identical. Two dependent variables were measured: achievement in the game and motivation. Achievement in

Table I. Number of Girls Who Were Unable to Solve Any Puzzles. Note: $\chi^2(1, N = 155) = 5.832$, $p < 0.05$.

Condition	n	Solved Zero Puzzles	%
Point-and-Click Interface	32	8	25%
Drag-and-Drop Interface	123	60	49%

the game was measured by whether or not the girls were able to solve any puzzles in the 30-minute playing period and the total number of puzzles they were able to solve. Motivation was measured by whether or not the girls played for the full 30 minutes.

3.2 Results

3.2.1 *Achievement*. Table I shows the number of girls who were unable to solve any puzzles in the game using the IBM-compatible version or the Macintosh version. Only girls who stayed and played for the full 30 minutes were included in this analysis (32 girls using the IBM-compatible version, 123 girls using the Macintosh version).

There was a significant difference in the proportion of girls who were unable to solve any puzzles using the point-and-click interface and the drag-and-drop interface, $\chi^2(1, N = 155) = 5.832$, $p < 0.05$. Of the girls who stayed and played for the full 30 minutes, 25% of girls using the point-and-click interface were unable to solve any puzzles compared to 49% of girls playing using the drag-and-drop interface. The analogous results for all girls who took part in the experiment, including those who left early, were 26% for the point-and-click interaction style and 59% for drag-and-drop interaction style, $\chi^2(1, N = 189) = 11.632$, $p < 0.01$.

A Mann-Whitney U test also revealed a significant difference in the number of puzzles solved by girls in each of the experimental conditions, $U = 1519$, $p < 0.05$. While girls in both conditions solved between 0 and 5 puzzles, girls using the point-and-click interaction style had higher rankings than girls using the drag-and-drop interaction style. The mean number of puzzles solved by girls using the point-and-click interaction style was 1.66 ($SD = 1.56$) and for girls using the drag-and-drop interaction style was 1.15 ($SD = 1.46$).

Therefore, girls using the point-and-click interaction style in the IBM-compatible version were more successful playing the game than were girls using the drag-and-drop interaction style in the Macintosh version of the game. Success was measured by whether or not the girls were able to solve puzzles in the game and was based on the number of puzzles the girls were able to solve for each interaction style.

3.2.2 *Motivation*. Examining all children who took part in the experiment, only two (6%) of the 34 girls using the point-and-click interaction style in the IBM-compatible version stayed and played for the full 30-minute period, while a significantly higher percentage of girls, 32 of the 155 girls (21%), left early when using the drag-and-drop interaction style in the Macintosh version, $\chi^2(1,\ n\ =\ 189)\ =\ 4.12,\ p\ <\ 0.05$. Therefore, girls using the point-and-click interaction style in the IBM-compatible version were more motivated to play the game than were girls using the drag-and-drop interaction style in the Macintosh version of the game where motivation was measured by the percentage of girls who chose to play the game for the full 30-minute period.[3]

3.3 Discussion

The results from Experiment I demonstrate that a point-and-click interaction style, used in an interactive learning environment, can be more effective in terms of performance and motivation than a drag-and-drop interaction style. The motivation result also indicates that the Macintosh version of the game may have had a more difficult and frustrating interface. Because this experiment examined girls using a rich, complex environment, implementation details and other issues related to playing the game itself may have interacted with the results.

One implementation issue that arises when using the drag-and-drop interaction style is the fact that it is a two-step movement: the mouse button is pressed down to perform the first action, and the mouse button is then released to perform the second action. Because manipulating some objects within *The Incredible Machine* requires three steps, the designers' choice of how to implement a three-step motion using a two-step interaction style could impact how users interact with the system. The Macintosh version of *The Incredible Machine* handles three-step movements by first performing a click (press the mouse button down and then release) to pick up the object, and then performing a drag-and-drop between the two endpoints. Because of this, regular objects and connector objects in this version are moved from the toolbox in two different ways, which creates an inconsistency in the interface. Because implementation details such as this may have contributed to the results, Experiment II examined children using the point-and-click and drag-and-drop interaction styles in isolation, removing other factors that may have been present in the interactive learning environment used in Experiment I.

4. EXPERIMENT II: MOUSE INTERACTION IN A SIMPLIFIED SETTING

Experiment II was conducted using a simplified software environment that presented a similar pair of mouse interaction styles to those used in the first experiment, but without the possibly confounding factors present in the more complex interactive learning environment provided by *The Incredible Machine*. In addition, by using a simplified environment, the experimenters were able to configure the run-time software to include additional data collection that was not available using the commercial software.

[3]For additional information concerning when the girls chose to leave see www.acm.org/pubs/citations/journals/tochi/2001-8-1/p1-inkpen/

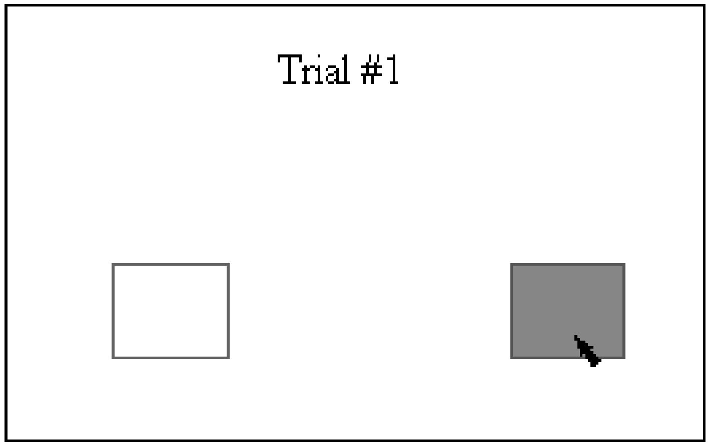

Fig. 2. The initial screen configuration produced by the software used in Experiment II. Children were required to move the solid (green) source box on the right to the outlined (red) target box on the left.

4.1 Method

4.1.1 *Participants*. The participants in Experiment II were 67 children (34 girls and 33 boys) between the ages of 9 and 13. The experiment took place at Science World during the month of August, 1996. All participants were visitors to Science World who volunteered to take part in the experiment. The age range of 9 to 13 was used to correspond with the age range used in Experiment I. The children who participated in Experiment II were not the same children who participated in Experiment I. The children were informed that the experiment would take approximately 20 minutes and consent forms were signed by a parent or guardian.[4]

4.1.2 *Hardware and Software*. A Silicon Graphics workstation with a three-button mouse was used to conduct the experiment. Because operation of the game only required input from the mouse, the keyboard was not presented to the children. Instead the mouse and mouse-pad were placed in front of the monitor and the children were able to place them wherever they felt comfortable using them.

Two versions of a special-purpose 2D graphics program written in OpenGL were developed to support the two different mouse interaction styles of point-and-click and drag-and-drop for this experiment. The program displayed two squares on the screen: a green, solid, source box and a red, outlined, target box (see Figure 2). The children were required to use a mouse to pick up the source box, move it over to the target box, and drop it inside the target box. The two boxes were displayed either 400 pixels or 800

[4]For background information on the amount of experience the participants had playing video or computer games see www.acm.org/pubs/citations/journals/tochi/2001-8-1/p1-inkpen/

pixels apart, and each box could be one of two sizes, 32 × 32 pixels or 64 × 64 pixels. This produced eight different possibilities of distance × source × target. There were approximately 90 pixels/inch on the monitor that was used in this experiment. Therefore, the width and height of the larger box was slightly less than three-quarters of an inch and the width and height of the smaller box was slightly more than one-third of an inch.

The manipulation of distance between boxes as well as source and target size was used to confirm that the movement could be accurately modeled by Fitts' Law. The two sizes were chosen to approximate small and large objects found in *The Incredible Machine*. The two distances were chosen to approximate typical short distance and long distance movements performed while playing *The Incredible Machine*. Each possible combination of distance × source × target comprised a trial. Two instances of each combination, for a total of 16 trials, comprised a block. The order of appearance of each distance × source × target combination was random within each block. After each block of 16 trials, the screen would turn black until a researcher pressed a key to progress on to the next block. The current trial number was displayed on the top of the screen throughout a session.

One version of the software used a point-and-click mouse interaction style while the other used a drag-and-drop mouse interaction style. The point-and-click interaction style required a child to click on the (green) source box to pick up the object (a click refers to pressing the mouse button down and then releasing it), then move the source box over to the (red) target box, and finally click the mouse button again to drop the source box inside the target box. The drag-and-drop interaction style required the child to press the mouse button down on the source box to pick it up (the mouse button was not released at this point), then move the source box over to the target box, and finally release the mouse button to drop the source box inside the target box. When the source box was successfully picked up, visual feedback was provided in both versions of the software by having the solid green source box turn into an outlined green box, and a small iconified picture of the source box was then attached to the cursor. During the movement of the source box, the iconified picture of the source box remained attached to the cursor until it was dropped, to provide visual feedback that the source box was picked-up and was being moved, similar to the visual feedback given by *The Incredible Machine*.

For both the point-and-click interaction style and the drag-and-drop interaction style, a successful pickup occurred when the tip of the cursor was inside the source box and the mouse button was pressed down. Whether or not the mouse button was released within the source box for the point-and-click interaction style was irrelevant. A successful drop of the source box occurred in the point-and-click interaction style when the tip of the cursor was in the target box and the mouse button was pressed down. Again, whether or not the mouse button was released inside the target box was irrelevant. For the drag-and-drop interaction style, a successful drop occurred when the tip of the cursor was inside the target box and the mouse button was released. When a child completed a trial (successfully picked up the source box and successfully dropped it inside the target box) an audible beep was sounded to signify success and the system automatically advanced to the next trial.

Two types of errors were possible during the task: pickup errors and drop errors. A pickup error occurred when the mouse button was clicked outside of the source box during a pickup attempt. A drop error occurred in the point-and-click interaction style when the mouse button was clicked outside of the target box during a drop attempt. A drop error occurred in the drag-and-drop interaction style when the mouse button was released outside of the target box. If a pickup error occurred for either the point-and-click interaction style or the drag-and-drop interaction style, neither the source box nor the cursor changed its appearance (i.e., the source box did not become outlined and an iconified picture of the source box was not attached to the cursor, which would have happened had the source box was successfully picked up). Following a pickup error, the only possible action in either interaction style was to attempt to pick up the source box again.

If a drop error occurred in the point-and-click interaction style there were three possible design choices for how the system could respond: (1) the source box could return to its original position, and the child would have to go back and pick it up again; (2) the source box could be placed in the incorrect place on the screen, requiring the child to pick it up from where it had been dropped and then move it to the target box; or (3) the source box could remain picked-up, allowing the child another attempt at correctly dropping it into the target box. Using the drag-and-drop interaction style there were only two possible design choices for the system response: (1) the source box could return to its original position and the child would have to go back and pick it up again; or (2) the source box could be placed in the incorrect place on the screen, requiring the child to pick it up again and then move it to the target box. The third option that existed for point-and-click did not make sense for drag-and-drop because the erroneous drop action required the mouse button to be released.

Because the motivation for Experiment II was to further explore the differences found in Experiment I, error-handling choices were designed to mimic the two versions of *The Incredible Machine*. Thus, the point-and-click version of the software was designed to follow the third option in order to mimic the IBM-compatible version of the game. In the event of a drop error, the source box would remain picked-up, attached to the cursor, allowing the child another attempt at correctly dropping the box inside the target box.

Similarly, the drag-and-drop version of the software was designed to follow the first option in order to mimic the Macintosh version of the game. In the event of a drop error, the source box would return to its original position, and the child would then have to go back and pick up the source box again.

The software recorded the time at which each mouse event occurred (both mouse button-down events and mouse button-up events) and the position on the screen where the mouse event occurred. In addition, times and positions were also recorded for mouse movement. The movement time for a trial began when the child first attempted to pick up the source box and ended when the source box was successfully dropped inside the target box.

4.1.3 *Procedure*. The sessions began with welcoming remarks from the researcher, followed by a brief, verbal introduction to the experiment. The task involved a series of trials where children were required to pick up a source box on the screen, move it to a target box on the screen, and drop the source box inside the target box. Each child was required to use two different interaction techniques to complete the task: point-and-click and drag-and-drop. The order of the interaction style was counterbalanced within each gender to evenly distribute any practice effects.

At the start of each interaction style, the software was demonstrated to the children, and they were given one practice block of 16 trials to become familiar with the interaction style. After the practice block, the children were asked to perform the same task as quickly as they could without making too many mistakes. The children performed four blocks of 16 trials for each interaction style. At the end of each block, the screen would go blank until the researcher pressed a button. This gave the children a break between blocks. Each child completed the experimental session in one 20-minute period. During the session the computer recorded the time for movements as well as the number of pickup and drop errors committed.

Upon completion of the session, each child was asked to rank his or her preference for interaction style on a nine-point scale. To facilitate this procedure for children, a pinwheel was used. The pinwheel consisted of two different colored cardboard circles, each divided into eight pieces. Both circles were slit and then placed together so that a portion of each circle could be seen (see Figure 3). Each interaction style was assigned a color, and the children were required to turn the pinwheel to indicate which interaction style they preferred and to what degree. If the children preferred the point-and-click interaction style, they would turn the pinwheel so that more of the point-and-click color was showing. If the children preferred the drag-and-drop interaction style, they would turn the pinwheel so that more of the drag-and-drop color was showing. If there was no preference of interaction style, the pinwheel could be placed with equal amounts of both colors showing. The number of pie-shaped pieces showing for a particular interaction style color represents its ranking. For example, the pinwheel on the left of Figure 3 ranks the point-and-click interaction style at three (three of the dark-colored, point, pieces showing), while the pinwheel on the right ranks the point-and-click interaction style at seven (seven of the dark-colored, point, pieces showing). A higher ranking reflects a greater preference for that interaction style. This technique was used to make ranking easier for the children because of its visual nature [Borys and Perlman 1985].

4.1.4 *Experimental Variables*. The design for Experiment II was a $2 \times 2 \times 2 \times 2 \times 2$ mixed design, with gender (girls and boys) as the between-

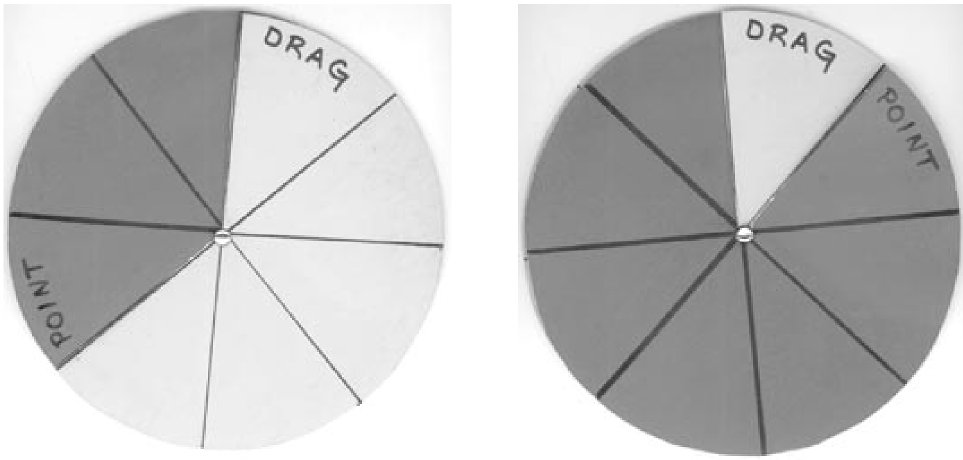

Fig. 3. Photographs of the pinwheels used by children to rank preference of interaction style on a nine-point scale.

subject factor and interaction style (point-and-click and drag-and-drop), distance between the source and the target (400 pixels and 800 pixels), size of source box (32×32 pixels and 64×64 pixels), and size of target box (32×32 pixels and 64×64 pixels) as the within-subjects factors.

The dependent measures in Experiment II were movement time, two types of errors, and preference. Movement time was the time it took to complete a trial (pick up the source box, move it over, and drop it inside the target box). Errors were the number of incorrect attempts at picking up the source box (pickup errors) and the number of incorrect attempts at dropping the source box into the target box (drop errors). Preference was children's ranking of interaction style on a nine-point scale.

4.2 Results

4.2.1 *Overall Movement Times*. The average movement times for the two interaction styles are shown in Table II. Only trials in which no errors occurred were included in the means. A significant main effect for interaction style was found with the point-and-click interaction style being faster than the drag-and-drop interaction style, $F(1, 65) = 10.026$, $p < 0.005$. No significant effect was found for gender, $F(1, 65) = 0.621$, ns. As expected, the main effects for target distance and target size were also significant, $F(1, 65) = 1369$, $p < 0.001$ and $F(1, 65) = 380$, $p < 0.001$ respectively. The size of the source box also had a significant effect on movement time, $F(1, 65) = 15.772$, $p < 0.001$.

4.2.2 *Errors*. The average number of errors for each interaction style is shown in Table II. A significant main effect for interaction style was found, $F(1, 65) = 5.202$, $p < 0.05$, with children committing more errors using the drag-and-drop interaction style than the point-and-click interaction

Table II. Average Movement Times and Number of Errors for Each Mouse Interaction Style

	n	Drag-and-Drop	Point-and-Click	F	p
Mean Movement Time (ms)	67	1342ms (SD 224ms)	1261ms (SD 227ms)	10.026	0.002
Mean Number of Errors	67	13.08 (SD 9.88)	10.58 (SD 10.10)	5.202	0.026

Table III. Average Number of Pickup and Drop Errors for Each Mouse Interaction Style

		Drag-and-Drop		Point-and-Click			
Error Type	n	M	SD	M	SD	F	p
Pickup Errors	67	9.15	6.75	7.15	6.78	5.853	0.018
Drop Errors	67	3.97	3.81	3.43	5.11	0.760	0.386

style. A significant main effect for type of error was also found, $F(1, 65) = 14.551$, $p < 0.001$, with children committing more pickup errors than drop errors. The average number of pickup and drop errors for each interaction style is shown in Table III. As expected, a significant interaction effect was found between type of error and source or target size, with the size of the source box significantly effecting the number of pickup errors and the size of the target box significantly effecting the number of drop errors. Type of error also interacted significantly with distance $\times$ source size $\times$ target size. No significant main effect of gender was found, $F(1, 65) = 0.036$, ns.

Pickup and drop errors were analyzed independently to examine the significant interaction effects. For pickup errors, significant main effects were still found for interaction style and size of the source box, $F(1, 65) = 5.853$, $p < 0.05$ and $F(1, 65) = 29.237$, $p < 0.001$ respectively, although a significant interaction of distance was also found. Further analysis revealed a significant main effect of interaction style for long-distance movements, $F(1, 65) 11.017$ $p < 0.01$, but not for short-distance movements, $F(1, 65) = 1.722 ns$.

For drop errors in general, no significant main effect was found for interaction style, $F(1, 65) = 0.760 ns$, while target size did reveal a significant main effect, $F(1, 65) = 25.775$, $p < 0.001$. However, gender also interacted significantly with target size, $F(1, 65) 8.113$, $p < 0.01$. Examining girls and boys separately, we found that interaction style was the only significant factor on the number of drop errors for girls, $F(1, 33) = 8.183$, $p < 0.01$ (see Figure 4), while for boys interaction style was not a significant factor on the number of drop errors, $F(1, 32) = 0.174 ns$. However, target size was a significant factor on the number of drop error for boys, $F(1, 32) = 29.522$, $p < 0.001$.

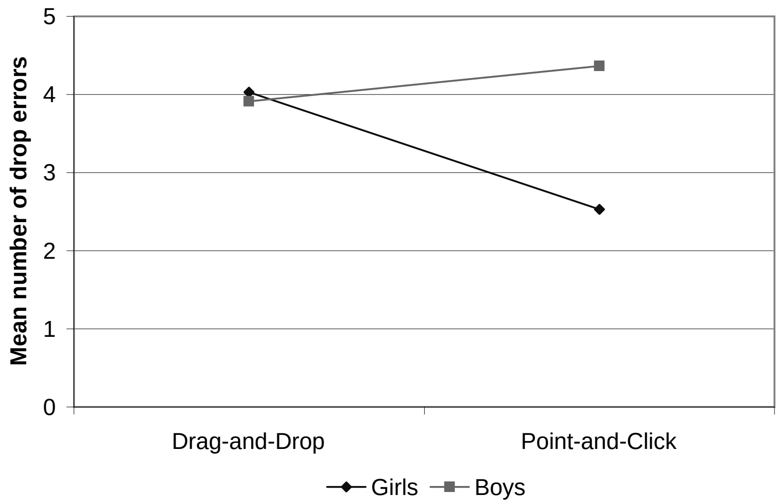

Fig. 4. Significant interaction of gender on the number of drop interactions for each interaction style.

4.2.3 *Fitts' Law*. Fitts' Law is an information processing model used to predict time to move to a target, where the movement time (MT) is dependent on the distance of the movement and the size of the target, $MT = a + b\ log_2(2A/W)$. A is the amplitude or distance moved to the target and W is the width of the target [Fitts 1954]. Welford proposed a slight variation of Fitts' Law which was used in this experiment: $MT = a + b\ log_2(A/W + 0.5)$. The logarithmic term is commonly referred to as the index of difficulty (ID), and the coefficients a and b are computed through linear regression. Fitts' Law is commonly used to compare mouse-based interaction techniques [MacKenzie 1992a; 1992b; 1991; Card et al. 1978; Akamatsu and MacKenzie 1996]. The comparison measure is the index of performance (IP), the reciprocal of the coefficient b from the Fitts or Welford equations.

The movement times from this experiment were analyzed to see if they were accurately modeled by Fitts' Law. In order to compensate for the variance in errors between conditions, Welford's computation of effective target width (We) was used to normalize the results based on the children's observed error rates for each interaction style [Welford 1968]. High correlations (shown in Table IV) were found between the time to complete the movement (MT) and the index of task difficulty (ID) for each of the interaction styles for both girls and boys, indicating that the movement was accurately modeled by Fitts' Law.

The average indices of performance for the drag-and-drop interaction style was 2.75 bits/sec. and for the point-and-click interaction style is 2.89 bits/sec. These figures are within the range expected (see MacKenzie [1992a] for a review of other studies using Fitts' Law). The Fitts' Law models computed through linear regression are shown in Table IV. Each

Table IV. The Fitts' Law Model for the Two Interaction Styles Computed through Linear Regression. MT is the movement time (ms), and ID is the index of difficulty (bits) where $ID = log_2\ (A/We+0.5)$.

Interaction Style	Fitts' Law Model	R^2
Drag-and-Drop	$MT = -244 + 434ID$	0.993
Point-and-Click	$MT = -123 + 381ID$	0.995

model was computing by regressing the four distance × target size conditions (IDs), calculated using Welford's variant of Fitts' Law. The slopes and intercepts are within the range expected, and both interaction styles have high R^2 values, which indicates a good fit with the model. Thus, for IDs similar to the ones used in this study, Drag-and-Drop will produce longer movement times than Point-and-Click.

4.2.4 *Preference*. The Mann-Whitney Test for preferences of mouse interaction styles revealed a significant difference in preference rankings by order in which the children performed the interaction styles, $U = 361.5$, $p < 0.05$. Children who used the drag-and-drop interaction style first were more likely to state a preference for the point-and-click interaction style than were children who used the point-and-click interaction style first.[5]

Children's preference of interaction style was analyzed using a t-test for a single mean. The mean was tested against an expected value of four which represents the neutral point of equal preference for the drag-and-drop and the point-and-click interaction styles. Children who used the drag-and-drop interaction style first significantly preferred the point-and-click interaction style, $t(34) = 4.92$, $p < 0.001$, while no significant preference of interaction style was found for children who used the point-and-click interaction style first, $t(31) = 0.15$, ns. A Mann-Whitney U test showed no significant interaction of gender between the preferences for girls and boys.

Children's preference of interaction style can be grouped into three nominal categories: prefer drag-and-drop, no preference, and prefer point-and-click. A ranking of zero to three would be placed in the "prefer drag-and-drop" group; a ranking of four would be placed in the "no preference" group; and a ranking of five to eight would be placed in the "prefer point-and-click" group. Using this grouping, 28% of the children preferred the drag-and-drop interaction style (19/67); 66% preferred the point-and-click interaction style (44/67); and 6% of the children did not have a preference for either interaction style (4/67). A Chi-square analysis of the children who expressed a preference of interaction style revealed that significantly more children preferred the point-and-click mouse interaction style than preferred the drag-and-drop mouse interaction style, $\chi^2(1, N = 63) = 9.921$, $p < 0.01$.

[5]A graph illustrating this data can be found at www.acm.org/pubs/citations/journals/tochi/2001-8-1/p1-inkpen/

4.3 Discussion

The results of Experiment II highlight that point-and-click is a more effective mouse interaction style than drag-and-drop, both in terms of speed and accuracy. The results also show that children tend to prefer the point-and-click interaction style. The following discussion examines other issues related to the observations gathered in Experiment II. These include the impact of pickup and drop errors for the given implementations, reasons why children prefer one interaction style over another, and the advantages and disadvantages of the drag-and-drop interaction style. In addition, state-transition diagrams and a GOMS style analysis [John and Vera 1992; Kieras 1988] of the two interaction styles from both Experiment I and Experiment II are presented to illustrate why the point-and-click interaction style is more effective than the drag-and-drop interaction style.[6]

4.3.1 *Impact of Errors*. The movement time results from the repeated measures analysis in Section 4.2.1 included only those trials in which no errors were made. This approach has been used in other similar studies [Card et al. 1978; Boritz et al. 1991]. While these results demonstrated that the point-and-click interaction style was faster than the drag-and-drop interaction style, the children made errors on approximately 14% of the trials. By including the trials in which errors occurred, differences between the movement times for drag-and-drop and point-and-click increased significantly, $F(1, 65) = 87.855$, $p < 0.001$. Instead of an average difference of 63ms between the two interaction methods, the average difference increased approximated 250% with the drag-and-drop interaction style becoming on average 220ms slower than the point-and-click interaction style.

4.3.1.1 *Pickup Errors*. The number of pickup errors children committed was, in most cases, more than double the number of drop errors. One reason for the high number of pickup errors could be the presentation of the task: the children were required to pick up a source box and move it to a target box. If the children focused primarily on the dropping portion of the task as opposed to the pick up portion, they may have been looking ahead to the task of reaching the target while they were still attempting to pick up the source box, and therefore they may have been careless.

One pickup error significantly increased the movement time for both the drag-and-drop interaction style, $F(1, 63) = 531$, $p < 0.001$, and the point-and-click interaction style, $F(1, 63) = 352$, $p < 0.001$. The time implication of one pickup error was on average 959ms for the drag-and-drop interaction style and 743ms for the point-and-click interaction style. It is interesting that a pickup error was slightly more costly using the drag-and-drop interaction style than using the point-and-click interaction

[6]Additional discussion of issues related to the experimental design of this study can be found at www.acm.org/pubs/citations/journals/tochi/2001-8-1/p1-inkpen/

style. This could be due to the increased time required to release the mouse button or that reorienting to perform a drag-and-drop motion takes more time than for a point-and-click motion. We did not perform an analysis on the detailed mouse trajectory data. Such an analysis might provide further insight into performance differences.

4.3.1.2 *Drop Errors*. The software handled drop errors quite differently between the point-and-click interaction style and the drag-and-drop interaction style. Because the point-and-click movement comprised two discrete actions (one click to pick up the object, one click to drop the object), failure during only one of the actions did not require repeating both actions. In contrast, error recovery during the drag-and-drop movement was more complicated because the action was one physical motion (press the button down to pick up the object, release the button to drop the object). If the target was missed while releasing the button, it was not possible to release the button again until it was depressed again. Experiment II implemented the assumption that it was not appropriate to leave the source box in an incorrect position because this was the case during similar circumstances in the puzzle-solving game environment from Experiment I. Because of this, the only alternative was to return the source box back to its original location, before the motion began. This required a child to go back to the very beginning of the trial and pick up the source box again, before reattempting the drop action.

While the software in Experiment II required the children to repeat the complete action in the case of a drop error using the drag-and-drop interaction style, this fact should not have affected the overall performance results presented in Experiment II (Section 4.2). For these results, only trials in which no errors occurred were included in the analysis. This choice may, however, have impacted the children's preference of interaction style or their motivation to play in Experiment I. It was important to investigate this type of error handling because for some tasks, such as the placement of connector object in *The Incredible Machine*, interim goals are not possible (i.e., leaving one end of the elastic dangling in the air). In situations such as this, utilizing a drag-and-drop interaction style is difficult because the subtasks are fused into one compound gesture. Difficulties arise when an error is made during one of the subtasks. The whole gesture must be repeated instead of just one of the subtasks, causing a substantial increase in the time for the overall movement. An alternative interaction style such as the point-and-click may be better suited in these situations.

The average time implication for one drop error was 2404ms using the drag-and-drop interaction style and 468ms using the point-and-click interaction style. On average, one drop error significantly increased the movement time 185% for drag-and-drop, $F = (1, 45) = 372.53$, $p < 0.001$, and 38% for point-and-click, $F(1, 48) = 69.98$, $p < 0.001$. Obviously, the error-handling mechanism utilized in the drag-and-drop interaction style contributed to the substantial time penalty for drop errors. This is an

Table V. Reasons Expressed by the Children for Their Preference of Interaction Style

	Comments	*n*	%
Point and Click	Point-and-Click was easier	19	43%
(N = 44)	Did not like having to hold my finger down on the mouse	12	27%
	The box went back all the way to the beginning using drag-and-drop	9	20.5%
	Drag-and-drop hurt my finger	8	18%
	My finger accidentally came off when using drag-and-drop	7	16%
	Point-and-click was faster	5	11%
	Point-and-click was more fun	2	4.5%
	No Comment	5	11%
Drag and Drop	More familiar with drag-and-drop	7	37%
(N = 19)	Did not have to click as much using drag-and-drop	2	10.5%
	Just had to let go to drop the box using drag-and-drop	2	10.5%
	Drag-and-drop was harder*	2	10.5%
	Drag-and-drop was faster	1	5%
	Drag-and-drop was more fun	1	5%
	Could see the cursor easier using drag-and-drop**	1	5%
	No Comment	4	21%

Note: Some children provided comments from multiple categories.
*Two children stated a preference based on the challenge of performing a more difficult interaction style.
** The cursor was identical for both interaction styles and therefore should not have been easier or harder to see for either point-and-click or drag-and-drop.

important result because it highlights the significant effect an interface choice can have on users' performance.

4.3.2 *Preference*. The children were asked why they preferred one method of interaction to another. The various responses are presented in Table V. Note that the percentages add up to more than 100%, since some children provided multiple comments. Many of the children who preferred the point-and-click interaction style explicitly stated that they found point-and-click easier than drag-and-drop. They also complained that the drag-and-drop interaction style made their fingers or hands tired from keeping the mouse button pressed down. Other researchers have also reported that children have difficulty maintaining pressure on the mouse button [Strommen 1994].

Most children who preferred the drag-and-drop interaction style to the point-and-click interaction style explained that it was because they were more familiar with drag-and-drop and that they commonly used software at home that involved dragging. Another positive benefit of the drag-and-drop interaction style, mentioned by some of the children, was the tactile feedback it provided. The children explained that they just had to let go of the button to drop the box. Therefore, maintaining pressure on the mouse button reinforced the fact that they were dragging the box. This notion is supported by Buxton [1986], who explains that a kinesthetic connectivity can help to reinforce the conceptual connectivity of the subtasks within a compound gesture.

4.4 Unified Analysis Using State Transition Diagrams

To understand the implications of errors for each of the interaction styles, we will analyze our implementations of the drag-and-drop mouse interaction style and the point-and-click mouse interaction style using state transition diagrams. The state transition diagram for movement using the drag-and-drop interaction style is shown in Figure 5(a) on the left while movement using the point-and-click interaction style is shown in Figure 5(b) on the right.

The diagrams show the necessary steps for completing the placement by moving from top to bottom in the diagram. The states are represented by the long rectangles. A state comprises a position on the screen and whether or not the object is picked up. The position on the screen could either be outside both the source and target boxes, inside the source box, or inside the target box. Transitions between states are represented in the diagram by arrows and correspond to mouse movements (M), mouse clicks (C) in the point-and-click version, and mouse button-down (D) or mouse button-up events (U) in the drag-and-drop version. M_1 represents mouse movements without maintaining pressure on the mouse button; M_2 represents mouse movements while maintaining pressure on the mouse button. Downward arrows represent forward progress toward completion of the placement while upward arrows represent errors or backward movement through the steps. Errors are represented by double-lined arrows.

Analyzing the state transition diagram shows, that while there are three possible errors for each interaction style, the impact of these errors differs depending on the interaction style. When an error is committed using the point-and-click interaction style the child remains in the current state, ready for another try. For the drag-and-drop interaction style, when an error is committed the child is forced to move backward to previous states. For two of these errors this means going back one state, but for the error originating in the fourth state, the child must go back three states all the way to the initial state (indicated by the long backward double arrow). The child must repeat states one to four as a result of this error. Another obvious difference is the addition of a new state for the drag-and-drop interaction style when an error is committed in the first state. The children must move through this additional state before beginning the movement again.

The state transition diagram provides the basis for a GOMS style analysis of the two interactions styles. The goal of the activity is to place the source box in the target box. This goal is accomplished through the functional-level operators of "pick up the source box," "move over to the target box," and "drop the source box." These functional-level operators are accomplished through the keystroke-level operators of "mouse movements" and "mouse button events." The methods to be used are the drag-and-drop interaction style and the point-and-click interaction style. Let t be the time to complete the goal; M_1 is a mouse movement without maintaining

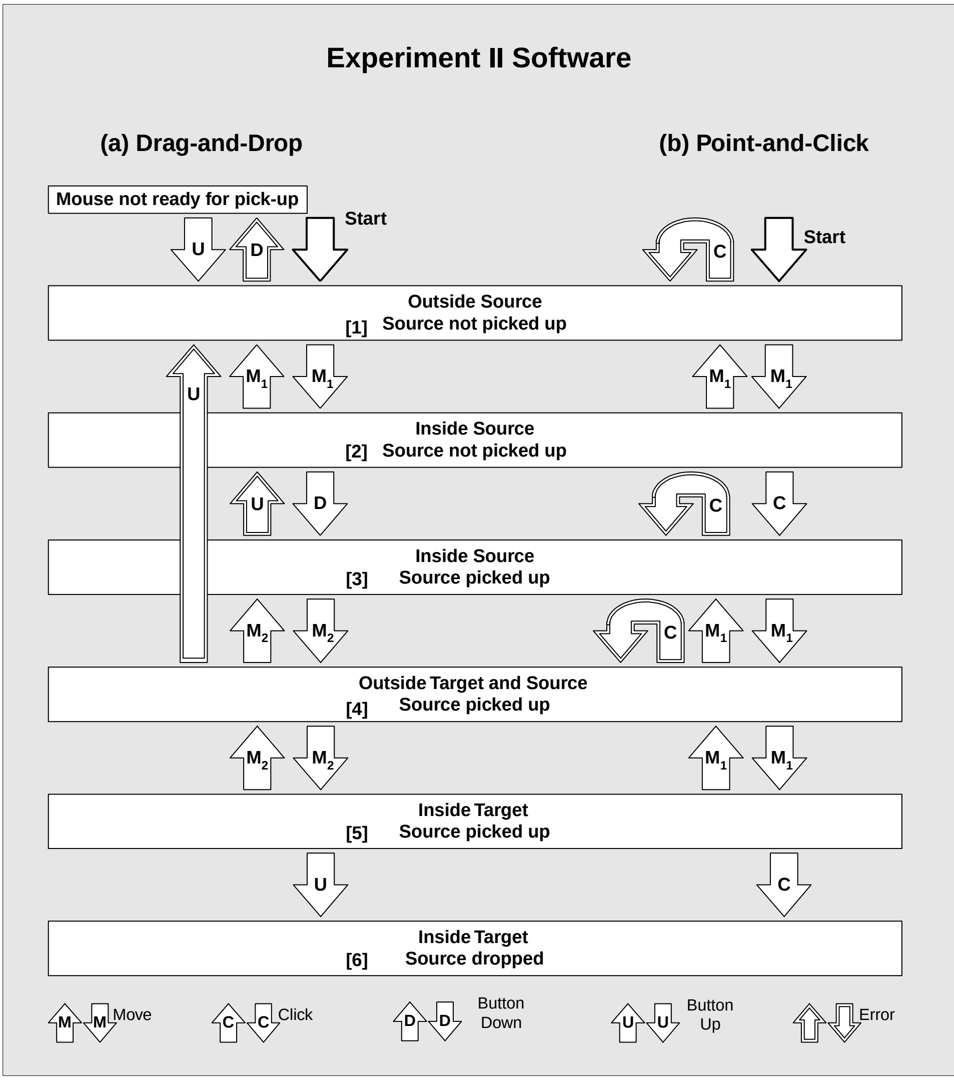

Fig. 5. State transition diagrams for (a) the drag-and-drop interaction style on the left and (b) the point-and-click interaction style on the right used in Experiment II.

pressure on the mouse button; M_2 is a mouse movements while maintaining pressure on the mouse button; B is mouse button event (a mouse button-down or button-up in the drag-and-drop interaction style and a mouse click for the point-and-click interaction style); and e_i is the number of errors committed in the ith state. The time to complete the goal for each interaction style is as follows:

$$\text{Drag-and-Drop: } t = e_1(B + B) + M_1 + B + e_3(B + B) + M_2 + e_4(B + M_1 + B + M_2) + M_2 + B$$

$$\text{Point-and-Click: } t = e_1(B) + M_1 + B + e_3(B) + M_1 + e_4(B) + M_1 + B$$

Table VI. Extra Events Required for the Drag-and-Drop Interaction Style Based on the Number of Errors Committed. M_2 is a mouse movement while maintaining pressure on the mouse; M_1 is a mouse movement without maintaining pressure on the mouse; and B is a mouse button event.

Error States	Extra Operations for the Drag-and-Drop Interaction Style
No errors	$2(M_2 - M_1)$
For each error in state 1 (e_1)	B
For each error in state 3 (e_3)	B
For each error in state 4 (e_4)	$M_1 + B + M_2$

This analysis assumes that all three mouse button events B take similar amounts of time. Table VI shows the extra operators required for the drag-and-drop interaction style depending on the number of errors made in each state. When no errors occur, the time difference between the drag-and-drop interaction style and the point-and-click interaction style is $2(M_2 - M_1)$. This equates to the difference between moving the object while maintaining pressure on the mouse button versus not maintaining pressure on the mouse button. Based on the results from previous research [MacKenzie et al. 1991] we know that $M_2 > M_1$, and as verified in Experiment II, the drag-and-drop interaction style would therefore be slower than the point-and-click interaction style. The analysis also shows, that for e_4 errors occurring in the fourth state, the time difference to complete the task between the drag-and-drop interaction style and the point-and-click interactions styles will be $e_4(M_1 + B + M_2) + 2(M_2 - M_1)$. For each error in the fourth state, this demonstrates that the drag-and-drop interaction style has one extra mouse button operation and two extra mouse movement operations.

The state transition diagram and GOMS-style analysis help validate and explain the results uncovered in this study. While error handling was initially recognized to be problematic using the drag-and-drop interaction style, this analysis provides more detailed information on where the major difficulties are and the degree of these difficulties. This method can help designers anticipate potential problems and better design the user interface or interaction style to minimize these difficulties.

Using the state transition diagrams we will also examine the implementations of these two interaction styles in the Macintosh and IBM-compatible versions of *The Incredible Machine* used in Experiment I. The state transition diagrams for placing regular objects is shown in Figure 6 with the drag-and-drop version on the left and the point-and-click version on the right.

Again, the diagrams show the necessary steps for completing the placement by moving from top to bottom in the diagram. A state comprises a position on the screen and whether or not the object is picked up. The position on the screen could either be inside the parts bin or inside the playing screen. Inside the playing screen has two variants: on top of

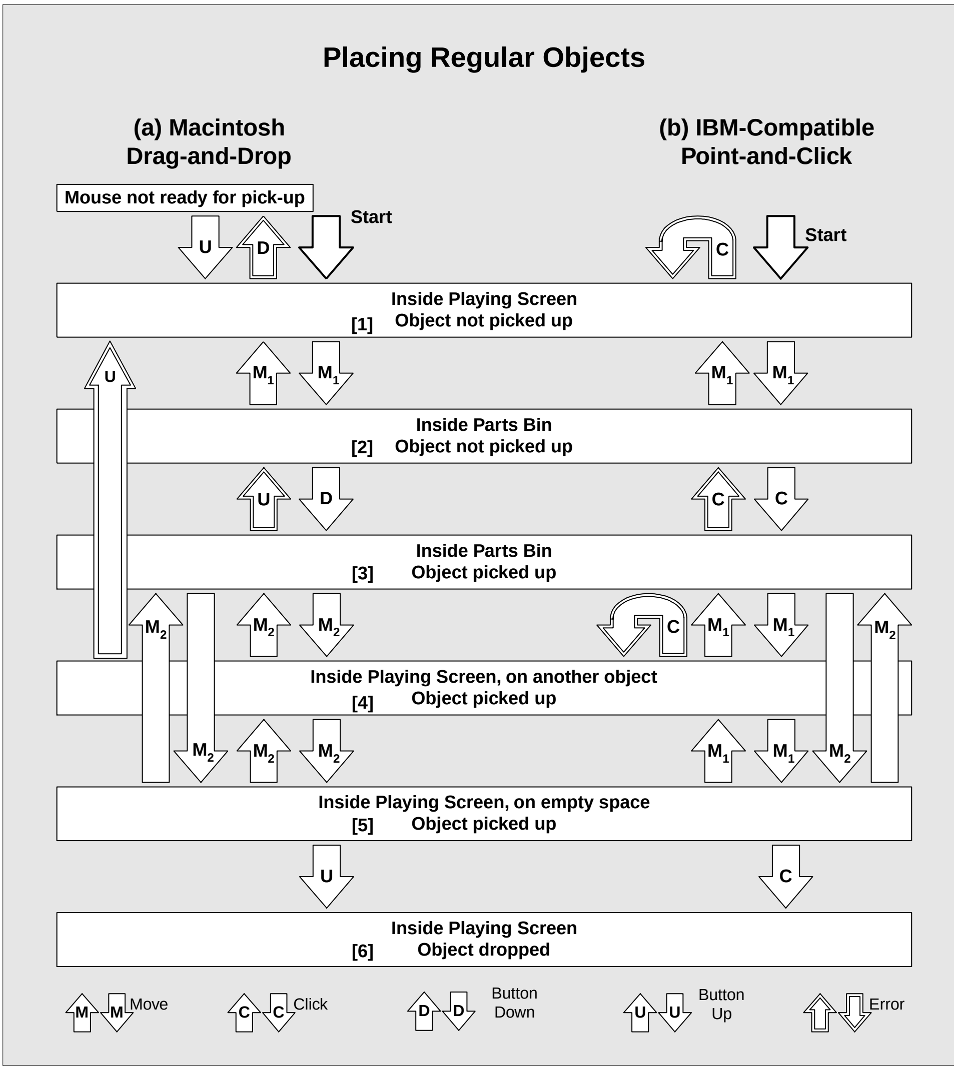

Fig. 6. State transition diagrams for placing regular objects in *The Incredible Machine* for (a) the Macintosh version on the left and (b) the IBM-compatible version on the right.

another object or on top of an empty space. The long arrows passing through the fourth state indicate that this movement may skip a state (i.e., during the placement of an object, the cursor may never need to pass over another object on the screen).

As was the case for our implementation of the drag-and-drop and the point-and-click interaction styles, detailed analysis through examination of the state transition diagrams reveal subtle but important differences, especially with respect to error handling. Analyzing the state transition diagrams we notice, that for both interaction styles, an error in the third state causes the child to back up one state. Errors in the first and fourth

Table VII. Extra Events Required for the Drag-and-Drop Interaction Style Based on the Number of Errors Committed. M_2 is a mouse movement while maintaining pressure on the mouse; M_1 is a mouse movement without maintaining pressure on the mouse; and B is a mouse button event.

Error States	Extra Operations for the Drag-and-Drop Interaction Style
No errors	$2(M_2 - M_1)$
For each error in state 1 (e_1)	B
For each error in state 3 (e_3)	0
For each error in state 4 (e_4)	$M_1 + B + M_2$

states, however, have different consequences for the two interaction styles. Using the point-and-click interaction style the child remains in the same state, ready for another try. In contrast, an error in the first state results in the addition of a new state for the drag-and-drop interaction which the child must move through before beginning the movement again. For an error in the fourth state, the drag-and-drop interaction style requires the child to move back three states, all the way to the initial state. This long backward error for the Macintosh drag-and-drop style suggests a difficulty with this interaction style that could be frustrating to children if this type of error occurs.

Using the GOMS-style analysis described previously, the time to complete the goal for each interaction style is as follows:

$$\text{Drag-and-Drop: } t = e_1(B + B) + M_1 + B + e_3(B + B) + M_2 + e_4(B + M_1 + B + M_2) + M_2 + B$$

$$\text{Point-and-Click: } t = e_1(B) + M_1 + B + e_3(B + B) + M_1 + e_4(B) + M_1 + B$$

Table VII shows the extra operators required for the drag-and-drop interaction style depending on the number of errors made in each state.

The state transition diagrams for placing connector objects is shown in Figure 7 with the drag-and-drop version shown on the left and the point-and-click version shown on the right. This technique clearly illustrates the problems children were having while placing connector objects in Experiment I.

These diagrams are similar to the ones representing the placement of regular objects in *The Incredible Machine* except for the composition of states. For the placement of connector objects, a state comprises a position on the screen and a status of the placement. The position on the screen could either be inside the parts bin or inside the playing screen. Inside the playing screen has three variants: on an invalid endpoint, on the first endpoint, or on the second endpoint. The status of the movement includes whether or not the connector object is picked up and whether or not the first endpoint has been attached.

Analyzing the state transition diagrams for placing connector objects reveals that the impact of errors is significantly different for the two interaction styles. The drag-and-drop interaction style has four long errors, causing the child to move backward anywhere from two to six states. In

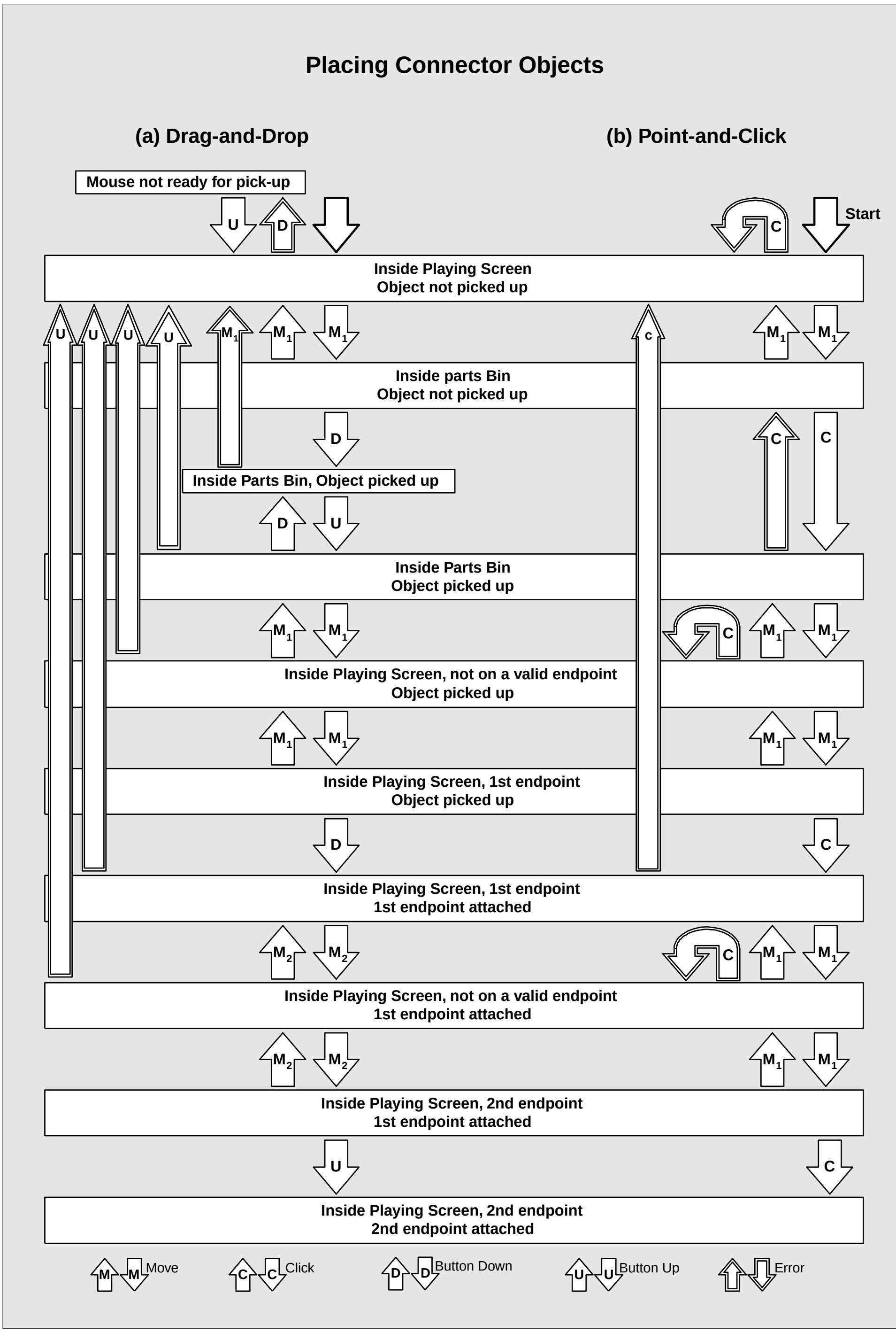

Fig. 7. State transition diagrams for placing connector objects in *The Incredible Machine* for (a) the Macintosh version on the left and (b) the IBM-compatible version on the right.

Table VIII. Extra Events Required for the Drag-and-Drop Interaction Style Based on the Number of Errors Committed. M_2 is a mouse movement while maintaining pressure on the mouse; M_1 is a mouse movement without maintaining pressure on the mouse; and B is a mouse button event.

Error States	Extra Operations for the Drag-and-Drop Interaction Style
No errors	$2(M_2 - M_1)$
For each error in state 1 (e_1)	B
For each error in state 3 (e_3)	M_1
For each error in state 4 (e_4)	$2M_1 + B$
For each error in state 6 (e_6)	0
For each error in state 7 (e_7)	$3M_1 + M_2 + B$

contrast, the point-and-click interaction style only has two backward errors, one of which only moves the child back one state. All errors in the drag-and-drop interaction style cause the child to return to the initial state to start all over again. Having to continually repeat steps as a result of errors could be frustrating for children. Another obvious difference between the two interaction styles is the addition of a new state between the second and third states for the Macintosh drag-and-drop interaction style. This means that children must be aware of this additional state, which does not exist when moving regular objects and therefore presents an inconsistency in the interaction style for the two types of objects in the Macintosh version of the game. Another new state is also added when an error is committed in the first state.

The GOMS-style analysis for the placement of connector objects quantifies the difference between the two interaction styles. The time to complete the goal for each interaction style is as follows:

$$\begin{aligned}\text{Drag-and-Drop: } t &= e_1(B + B) + M_1 + B + e_3(B + M_1 + B) + M_1 \\ &\quad + e_4(B + M_1 + B + M_1) + M_1 + B + e_6(B + M_1 + B + M_1 + M_1 \\ &\quad + B) + M_2 + e_7(B + M_1 + B + M_1 + M_1 + B + M_2) + M_2 + B \\ \text{Point-and-Click: } t &= e_1(B) + M_1 + B + e_3(B + B) + M_1 + e_4(B) + M_1 + B \\ &\quad + e_6(B + M_1 + B + M_1 + M_1 + B) + M_1 + e_7(B) + M_1 + B\end{aligned}$$

Table VIII shows the extra steps required for the drag-and-drop interaction style on the number of errors made in each state. This analysis reveals that errors made while using the drag-and-drop interaction style result in more operations than when using the point-and-click interaction style. In particular, an error in the seventh state will result in four extra mouse movement operations and one extra mouse button operation.

The state transition diagramming technique presented in this section is an effective method of analyzing interaction styles. It provides a detailed, visual representation to easily highlight potential problems with an interaction style. As illustrated in our use of the technique, comparisons can be made between alternative interaction styles to help system designers select the most effective interaction styles for given tasks. The technique can also provide quick feedback when a detailed examination of movement times and errors is not possible.

5. CONCLUSION

We draw both specific conclusions about the two mouse interaction styles used in our studies and more general conclusions about the design of children's software.

5.1 Research Strategies

A high degree of reliability was obtained in this work by utilizing multiple research methods to explore the issue of mouse interaction style, where both studies uncovered the same results. Experiment I was a field study which focused on investigating children's natural interactions while they played the computer game *The Incredible Machine*, controlling for the type of mouse interaction style used. This experiment provided realism but was limited in terms of precision because the differences observed between the computer platforms may not have been completely attributable to the interaction style used. Experiment II used a controlled experiment to gain more precise knowledge of children's use of both interaction styles but lacked realism because the task the children performed only represented one action from the complex game environment used in Experiment I. By combining the information obtained from both experiments and analyzing it with state-transition diagrams and GOMS models, we gain a better understanding of how the choice of interaction style can impact children's use of interactive learning environments.

5.2 Drag-and-Drop versus Point-and-Click

One of the most compelling results of this study deals with the children's achievement in the puzzle-solving game. Other studies have examined user interface issues with respect to learning [Root and Canby 1998; Catrambone and Carroll 1987; Jackson et al. 1998] but not from the perspective of how interaction styles impact children's performance in problem-solving environments.

The results of this study show that utilizing a point-and-click interaction style in children's software can be more effective than using a drag-and-drop interaction style. Children were able to perform point-and-click interactions significantly faster and with significantly fewer errors than drag-and-drop interactions, and more children preferred the point-and-click interaction style. This study also dealt with the issue of whether or not the choice of mouse interaction style impacts children's motivation and achievement in a learning environment. Do a few milliseconds or a couple of extra errors once in a while really make a difference for children's use of an interactive learning environment? The results of our first study (run only with girls) demonstrated that the choice of interaction style can significantly impact both motivation and achievement. The girls using the point-and-click version of *The Incredible Machine* were more motivated to continue playing and were more successful in the game than were girls using the drag-and-drop version of the game.

While this study shows a strong advantage for utilizing a point-and-click mouse interaction style over a drag-and-drop mouse interaction style, it is important to recognize that these results may be implementation or task dependent. As presented in this and other studies, Fitts' Law has shown that dragging is a slower, more error-prone interaction style than pointing, and that the distance of the movement and size of the target also effect the results [MacKenzie et al. 1991]. Therefore, the impact of utilizing a drag-and-drop interaction style over a point-and-click interaction style will depend on how far objects need to be moved and on the size of the objects or targets. The tasks, object sizes, and distances moved in this study are representative of those in many software environments. The implementations of the two interaction styles were similar to those found in other pieces of software in many aspects but atypical in others. The point-and-click interaction style was, for both Experiments, representative of most implementations of this method. In Experiment I, the movement of regular objects using the drag-and-drop interaction style was representative; however, the manipulation of connector objects was atypical, as has been discussed previously. In Experiment II, the movement of drag-and-drop was representative, but the error-handling mechanism was atypical for the type of task performed.

5.3 Impact on the Design of Children's Software

The results of this study are significant for designers of children's software because often such software is developed without involving children in the design process [Druin 1999; Druin et al. 1997]. Moreover, there is little research on effective support of children's interactions with computers, and what research does exist is often ignored by software developers. For example, previous research has observed that some children have difficulty performing a dragging motion because of the physical requirements needed to maintain constant pressure on the mouse button [Strommen 1994]. Research on adults has shown that a dragging task is slower and that more errors are made as compared to a pointing task [MacKenzie et al. 1991]. Despite this knowledge, children's software is often implemented to utilize a drag-and-drop interaction style. Bringing solid research and strong results, such as the study discussed in this paper, to the forefront may help make designers of children's software think more about the implications of their design choices.

As the use of computers becomes more prominent we need to be even more sensitive to how children interact with computers. Computers and software are no longer being used only by those who wish to play games in their spare time. While many children are quite proficient in their interactions with computers, and many have adapted to electronic game interfaces, this is not true of all children. The designers of children's software must be careful not to assume too much about children's ability to adapt to their systems, especially in the design of educational software because, as shown in this study, design choices do significantly affect children's interactions with the software. In a school environment, as a learning tool, computers will be used by all students, so the software must be accessible to all students.

5.4 Future Work

The study presented in this paper examined mouse-based input for children because, at the present time, the mouse is the major, nonkeyboard input device for computers found both at home and at school. Whether or not the mouse is an appropriate input device for children is another question, which future research should investigate.

The focus of this research has been to explore common mouse interaction styles to determine their impact on ease of use and whether or not they interfere with tasks children perform in a learning environment. Continued research on human-computer interaction issues for children in educational environments is extremely important. Only through this kind of research will we gain an understanding of how computer systems and software can be effectively designed for children and what kind of impact this technology can have on their lives.

ACKNOWLEDGMENTS

This study was a part of a large-scale project on Electronic Games for Education in Math and Science (E-GEMS) (http://taz.cs.ubc.ca/egems/home.html). E-GEMS is a collaborative effort among scientists, mathematicians, educators, professional electronic game and educational software developers, classroom teachers, and children. The goal of E-GEMS is to increase the proportion of children who enjoy exploring and learning math and science concepts through the use of electronic games. Within the scope of the E-GEMS research, we recognize, that in order to achieve the overall goals of the project, we must understand how children interact with the technology.

Thanks to Dr. Kellogg Booth for his considerable help with this paper. Dr. Maria Klawe, Dr. Rena Upitis, Dr. Shelley Hymel, Dr. Mutindi Ndunda, Hardeep Sidhu, Neil Martin, and Jim Boritz also provided much appreciated help with this study. I would also like to thank Dr. Bonnie John and the anonymous reviewers for their feedback on the statistical analysis presented in the paper.

REFERENCES

AKAMATSU, M. AND MACKENZIE, I. S. 1996. Movement characteristics using a mouse with tactile and force feedback. *Int. J. Hum.-Comput. Stud. 45*, 4, 483–493.

BERKOVITZ, J. 1994. Graphical interfaces for young children in a software-based mathematics curriculum. In *Proceedings of the ACM Conference on Human Factors in Computing Systems: Celebrating Interdependence* (CHI '94, Boston, MA, Apr. 24–28). ACM Press, New York, NY, 247–248.

Boritz, J., Booth, K. S., and Cowan, W. B. 1991. Fitts' law studies of directional mouse movement. In *Proceedings of the Conference on Graphics Interface '91* (Calgary, Alberta, June 3–7), W. A. Davis and B. Wyvill, Chairs. Morgan Kaufmann Publishers Inc., San Francisco, CA, 216–223.

Borys, S. and Perlman, D. 1985. Gender differences in loneliness. *Pers. Soc. Psychol. Bull. 11*, 63–74.

Buxton, W. 1986. There's more to interaction than meets the eye: Some issues in manual input. In *User Centered System Design: New Perspectives on Human Computer Interaction*, D. Norman and S. Draper, Eds. Lawrence Erlbaum Associates Inc., Hillsdale, NJ, 319–337.

Card, S., English, W., and Burr, B. 1978. Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a CRT. *Ergonomics 21*, 601–613.

Catrambone, R. and Carroll, J. M. 1987. Learning a word processing system with training wheels and guided exploration. In *Proceedings of the ACM CHI+GI '87 Conference on Human Factors in Computing Systems and Graphics Interface* (CHI '87, Toronto, Ont., Canada, Apr. 5–9), J. M. Carroll and P. P. Tanner, Eds. ACM Press, New York, NY, 169–174.

Druin, A. 1998. Beginning a discussion about kids, technology, and design. In *The Design of Children's Technology*, A. Druin, Ed. Morgan Kaufmann series in interactive technologies. Morgan Kaufmann Publishers Inc., San Francisco, CA, xivxxiii.

Druin, A., Stewart, J., Proft, D., Bederson, B., and Hollan, J. 1997. KidPad: A design collaboration between children, technologists, and educators. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI '97, Atlanta, GA, Mar. 22–27), S. Pemberton, Ed. ACM Press, New York, NY, 463–470.

Gillan, D. J., Holden, K., Adam, S., Rudisill, M., and Magee, L. 1990. How does Fitts' law fit pointing and dragging?. In *Proceedings of the ACM Conference on Human Factors in Computing Systems: Empowering People* (CHI '90, Seattle, WA, Apr. 1–5), J. C. Chew and J. Whiteside, Eds. ACM Press, New York, NY, 227–234.

Hall, J. and Cooper, J. 1991. Gender, experience and attributions to the computer. *J. Educ. Comp. Res. 7*, 1, 51–60.

Hanna, L., Risden, K., Czerwinski, M., and Alexander, K. J. 1998. The role of usability research in designing children's computer products. In *The Design of Children's Technology*, A. Druin, Ed. Morgan Kaufmann series in interactive technologies. Morgan Kaufmann Publishers Inc., San Francisco, CA, 3–26.

Huff, C. and Cooper, J. 1987. Sex bias in educational software: The effect of designers' sterotypes on the software they design. *J. Appl. Soc. Psych. 17*, 6, 519–532.

Inkpen, K. M. 1997. Three important research agendas for educational multimedia: Learning, children, and gender. In *Proceedings of the World Conference on Educational Multimedia and Hypermedia'97* (Calgary, Alberta, Canada). 521–526.

Inkpen, K., Booth, K. S., Klawe, M., and Upitis, R. 1995. Playing together beats playing apart, especially for girls. In *Proceedings of the First International Conference on Computer Support for Collaborative Learning* (CSCL '95, Bloomington, IN, Oct. 17–20), J. L. Schnase and E. L. Cunnius, Eds. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, 177–181.

Inkpen, K., Klawe, M., Lawry, J., Sedighian, K., Leroux, S., Hsu, D., Upitis, R., Anderson, A., and Ndunda, M. 1994. "We have never-forgetful flowers in our garden": girls' responses to electronic games. *J. Comput. Math. Sci. Teach. 13*, 4 (), 383–403.

Jackson, S. L., Krajcik, J., and Soloway, E. 1998. The design of guided learner-adaptable scaffolding in interactive learning environments. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI '98, Los Angeles, CA, Apr. 18–23), C.-M. Karat, A. Lund, J. Coutaz, and J. Karat, Eds. ACM Press/Addison-Wesley Publ. Co., New York, NY, 187–194.

Jacob, R. J. K., Sibert, L. E., McFarlane, D. C., and Mullen, M. P. 1994. Integrality and separability of input devices. *ACM Trans. Comput. Hum. Interact. 1*, 1 (Mar.), 3–26.

John, B. E. and Vera, A. H. 1992. A GOMS analysis of a graphic machine-paced, highly interactive task. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (CHI '92, Monterey, CA, May 3–7), P. Bauersfeld, J. Bennett, and G. Lynch, Eds. ACM Press, New York, NY, 251–258.

Kieras, D. E. 1997. Towards a practical GOMS model methodology for user interface design. In *Handbook of Human-Computer Interaction*, M. G. Helander, T. K. Landauer, and V. Prabhu, Eds. Elsevier Science Publishers Ltd., Essex, UK, 137–157.

Lawry, J., Upitis, R., Klawe, M., Anderson, A., Inkpen, K. I., Ndunda, M., Hsu, D., Leroux, S., and Sedighian, K. 1995. Exploring common conceptions about boys and electronic games. *J. Comput. Math. Sci. Teach. 14*, 4, 439–459.

Lockheed, M. E. 1985. Women, girls and computers: A first look at the evidence. *Sex Roles 13*, 3/4, 115–122.

MacKenzie, I. S. 1992a. Fitts' law as a research and design tool in human-computer interaction. *Human-Comput. Interact. 7*, 1, 91–139.

MacKenzie, I. S. 1992b. Movement time prediction in human-computer interfaces. In *Proceedings of the Conference on Graphics Interface '92* (Vancouver, BC, Canada, May 11–15), K. S. Booth and A. Fournier, Eds. Morgan Kaufmann Publishers Inc., San Francisco, CA, 140–150.

MacKenzie, I. S., Sellen, A., and Buxton, W. A. S. 1991. A comparison of input devices in elemental pointing and dragging tasks. In *Proceedings of the Conference on Human Factors in Computing Systems: Reaching through Technology* (CHI '91, New Orleans, LA, Apr. 27–May 2), S. P. Robertson, G. M. Olson, and J. S. Olson, Eds. ACM Press, New York, NY, 161–166.

Marzio, P. 1973. *Rube Goldberg: His Life and Work*. Harper and Row Publishers, Inc., New York, NY.

Root, R. W. and Canby, A. 1998. There's more to direct manipulation than meets the eye. In *Proceedings of the 32nd Annual Meeting on Human Factors Society*. 278.

Strommen, E. 1994. Children's use of mouse-based interfaces to control virtual travel. In *Proceedings of the ACM Conference on Human Factors in Computing Systems: Celebrating Interdependence* (CHI '94, Boston, MA, Apr. 24–28). ACM Press, New York, NY, 405–410.

Upitis, R. and Koch, C. 1996. Is equal computer time fair for girls? In *Proceedings of the 6th Conference of the Internet Society on INET '96* (Montreal, Quebec, Canada).

Welford, A. T. 1968. *The Fundamentals of Skill*. Methuen and Co., Ltd., London, England.

Wilder, G., Mackie, D., and Cooper, J. 1985. Gender and computers: Two surveys of computer-related attitudes. *Sex Roles 13*, 3-4, 215–228.

Received: June 1997; revised: December 1998, August 1999, and February 2000; accepted: February 2000