Dynamic Size and Speed Cursor for Large, High-Resolution Displays

Robert Ball, Michael Szwedo, and Chris North
Center for Human-Computer Interaction
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061, USA
north@cs.vt.edu

ABSTRACT
As larger displays become more available their lack of adequate input techniques become apparent. In this paper we show the scalability of the dynamic size and speed cursor for large, high-resolution displays. We introduce the idea of a dynamic paradigm for input devices, explain three implementations of the dynamic size and speed (DSS) cursor and explain results of an experiment. In our experiment we compared the three different implementations of the dynamic size and speed cursor to cursor warping and standard cursor settings. In the experiment we found gender bias for two different tasks (clicking and simple drag and drop), found that one of the dynamic size and speed cursor implementations generally outperformed cursor warping and the standard cursor setting, and explain how distance to and size of targets effected results. We conclude by suggesting the use of a dynamic size and speed cursor with large, high-resolution displays.

Author Keywords
input techniques, dynamic, large displays

INTRODUCTION

Large, high-resolution displays are becoming increasingly more common. With the ability to use off the shelf hardware and free open-source software such as DMX [11] and Chromium [8] more and more people are able to build graphics clusters as one large, unified display.

The standard mouse technique is problematic with large, high-resolution displays. As displays grow a problem known as 'mouse rowing' is introduced. Mouse rowing is the act of trying to move one's cursor from one side of the display to the other through a series of repeated physical movements of moving the mouse rapidly on the desk. This problem can be amplified as people use more than two monitors. For example, see figure 1.

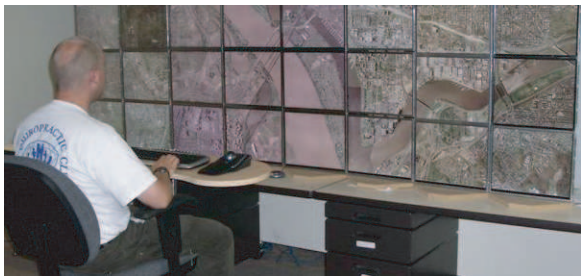

Figure 1. A picture of a user interacting with the large display used in this paper.

To overcome this problem, some researchers have suggested cursor warping [7] or head tracking [7] [1] to non-continuously reposition the cursor to a different area of the display. For example, with head tracking, to reposition one's cursor to another part of the display, one need only look at a different part of the display and the cursor is automatically repositioned to that area.

The purpose of this paper is to introduce a new interactive paradigm of how an existing mouse cursor's usability might be improved. In this paper we introduce the idea of a dynamic size and speed cursor (DSS cursor). Following the idea of increased cursor visibility from [6], we have developed a technique of both greater visibility on the display as well as overcoming the problem of mouse rowing.

Fundamentally, how does one quickly and precisely point to one pixel among millions and millions on a large, high-resolution display? Using Fitt's law as a basis, the farther away a target, the longer it takes to point to a target. At the same time, the smaller the object the longer it takes to point to it. Hence, time to point to targets across large, high-resolution displays, where targets are significantly smaller than display size, increases dramatically.

In this paper we suggest that cursor size and speed should not be static, but dynamic for the purpose of using Fitt's law to one's advantage, not disadvantage. It should be based not on historical standards, but on what would be best suited to

each display. We introduce the dynamic size and speed cursor concept, explain our implementations, discuss an evaluation of our implementations compared to other techniques, and discuss possible future input techniques.

PREVIOUS WORK

As multiple monitor usage has increased, several studies have been performed to evaluate different aspects of the usability of larger displays and multiple monitor displays. For larger displays, such as projector-based displays, there have been several studies that have shown better performance on the larger display than a similar smaller counterpart, such as a desktop monitor. Such studies have shown an increase in memory [16, 27], spatial performance [25], 3D virtual navigation [26], and multi-tasking [23].

On multiple monitors there have also been several studies showing improvements in user performance. Such studies include an increase in performance in multi-tasking [2, 9], basic perception and navigation [3], and offset gender bias in performance [10, 24].

There have also been a few experiments and ideas that deal with bezels in tiled displays and overcoming their distortive side affects. Both Baudisch, et al. [4] and Mackinlay, et al. [17] investigate different aspects of dealing with bezels.

As more studies show the usefulness of large displays, different interactive techniques have followed. A number of different types of techniques, from using less traditional input techniques to different ways of interacting with the mouse have been developed. Large displays and multiple monitor displays are inherently different from smaller displays.

A large amount of research has been done on pen-based interaction. For example, Tivoli [22], Flatland [18], and Fluid Interaction [15] are all well-known examples. These techniques have historically been used for white-board type interactions.

Other well-known interaction techniques also exist, such as laster pointers (e.g. [19]) and head-tracking (e.g. [1] and [7]). A range of techniques have also been created for facilitating users in accessing objects physically far from them on large displays using various software techniques (e.g. [12], [5], [14], and [21]).

Mouse-based interactions have also been developed. Of particular note is the high-density cursor by Baudisch, et al. that focus on a greater visibility of the mouse cursor [6]. Benko, et al. introduced the concept of cursor warping [7]. Cursor warping is the act of instantly jumping (moving) the position of the cursor from one monitor to another. In this paper we explain our extended implementation of cursor warping. Wallace, et al. created a multi-cursor X window manager at the systems level [28]. Their contributions differ from other cursor ideas in that they implemented their cursor at the desktop level instead of a single application.

DYNAMIC CURSOR INTRODUCED

Cursor speed is dependent on operating systems and user preference. However, in general, most operating systems use a bimodal speed paradigm, a base speed in which the cursor will move m pixels per unit of physical mouse movement. They then use a certain physical speed threshold to increase the cursor speed, usually called *acceleration*, to move n pixels per mouse movement.

In our dynamic cursor paradigm we propose to proportionally change the cursor's speed by maintaining the cursor threshold and increasing the acceleration so that users can move the cursor across the display at much faster rates. In other words, by increasing the accelerated speed, with a marginally faster movement than normal the cursor goes much farther than it normally would. However, slow speed is still preserved for fine control. With the DSS cursor the acceleration speed is set proportional to the size of the display.

	Threshold	**Cursor Acceleration Rate**	**Cursor Size in Pixels**
Control Cursor	10	1x	32x32
DSS Cursor	10	6x	400x400

Table 1. This table shows how the different mouse setting compare to each other.

In this paper we describe an experiment that compares two different cursor setting. Table 1 shows that if the user were using either cursor and were detected as physically moving the mouse slowly across the desk, below the given threshold of 10, then the operating system would move the cursor m pixels on the screen accordingly. Then, supposing the user made a faster physical movement of the mouse on the desk, with a speed above the given threshold of 10, then the operating system would move the control cursor 1*m pixels on the screen and the DSS cursor 6*m pixels on the screen.

Suppose that the normal acceleration of the cursor sends the cursor from one side of a desktop monitor to the other by moving the mouse two inches on the desk when mouse movement is detected above the threshold. Our DSS cursor would alter the acceleration of the cursor in such a way as to send the cursor from one side of the display to the other with two inches of physical mouse movement regardless of the size of the display. So, if the display had four, twelve, or one-hundred times the number of pixels, the accelerated speed component would be set appropriately.

Although the cursor size is also based on individual operating systems and user preference, it usually ranges from 16x16 pixels to 32x32 pixels large. This means that if the display had a low density of pixels like a resolution of 640x480, then the same cursor would appear larger than if it had a higher pixel count like on a resolution of 1600x1200. In general, the larger the display, the smaller the percent of space that the cursor takes up. As a result, the cursor becomes increasing more difficult to visually detect for higher pixel count displays.

By increasing the size and speed of the cursor proportionally to the display we are able to overcome the problems of cursor visibility and mouse rowing. By increasing the size of the cursor proportional to the size of the display, users are able to quickly find the cursor's position and reposition it accordingly. For example, if the cursor takes up one percent of the display on a normal desktop monitor then the DSS cursor would still take up one percent of a larger display.

For the purpose of this paper we had two research questions.

- How does the DSS cursor compare to the standard cursor?
- How does the DSS cursor compare to other known cursor interaction techniques, specifically cursor warping with a keyboard?

We chose cursor warping with a keyboard to compare against for three reasons. First, it is easy to implement into modern operating systems without additional hardware than what already comes standard (i.e. mouse and keyboard). Second, we felt that the cursor warping with a keyboard technique might perform better than the standard cursor at great distances. Third, we wanted to compare against the best-known technique from Benko, et al. [7], in which cursor warping (using a physical mouse) was shown to be the best technique.

Our hypothesis was that the DSS cursor would do better than the standard cursor and cursor warping on large displays. Also, we thought that cursor warping would perform better than the standard cursor, though not better than the DSS cursor.

IMPLEMENTATION

This section explains the hardware and software specifications used for the purpose of replication of the experiment.

Hardware and Software Specifications

The experimental hardware consisted of twenty-four seventeen inch LCD monitors (1280x1024 resolution each) and twelve computers running GNU/Linux for the purpose of creating a large, high-resolution display. Each computer powered two monitors. We removed the plastic bezel of each monitor to reduce the gap between monitors. We then set the monitors on reconfigurable wooden stands in a 8x3 matrix (see figure 2).

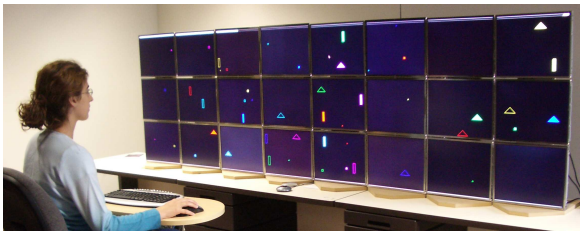

Figure 2. A picture of a participant performing the third task of the experiment on the 8x3 monitor cluster.

We networked the twelve computers together in a private network using a gigabit switch. We then installed DMX (Distributed Multihead X) [11] to create a unified display. DMX is a proxy X server that provides multi-head support for multiple displays attached to different machines. For all appearances to the user, when running DMX the display appears to be one single GNU/Linux desktop that runs standard windows manager (e.g. KDE, GNOME, Fluxbox, etc.).

We used all twenty-four tiled monitors to test the different interactive techniques. We did not compare our technique on smaller displays as the smaller the display, the closer the DSS cursor would be to the standard cursor in both speed and size.

We created an OpenGL [20] application using glut [13] in order to create a working prototype of our DSS cursor technique. However, OpenGL does not work efficiently on clustered displays. As a result, we also used Chromium [8], an open-source library that uses real-time parallel rendering of OpenGL.

Cursor Technique Implementation

We implemented three different variations of the DSS cursor. We then compared our implementations to the standard cursor and to cursor warping.

The first implementation of the DSS cursor was based on a simple assumption that a bigger, **high-speed cursor** would always be more helpful on a larger display. We set the speed and size by manual calculations and trial and error of the high-speed cursor to 6x the normal acceleration and a size of 400x400 pixels.

After developing the high-speed cursor, we found that having a larger cursor was an advantage to cursor visibility, but introduced the problem of obscuring data. As a result, we created another implementation of a dynamic speed and sized cursor. We created a dual-speed and dual-sized cursor that could be either large and high speed or small and slow depending on user input called the **manual cursor**. By pressing either the right button of the mouse or by pressing the space bar, users were able to toggle between a large and high-speed cursor mode and the normal size and normal speed cursor mode. The normal mode was the same size and speed as the control cursor. The faster mode was the same speed and size as the high-speed cursor (the first DSS cursor implementation). We change the speed at real-time by using the Xlib library and size using the OpenGL library.

For the purpose of our experiment, we ran a small formative study to determine which toggle input would be better - the right button on the mouse or the space bar. We found that using the right button on the mouse was slower and confused users more than the space bar on the keyboard. As a result, we chose to test our manual cursor using the space bar. The space bar was chosen over other keys based on the size of the space bar, making it easier for participants to not mistakenly press a wrong key without having to look at the keyboard.

After completing the implementation of the manual cursor we postulated that people might both prefer and perform better on a cursor that changed size and speed automatically. By sampling the mouse's input every half second we automatically changed the size and speed of the cursor based on physical speed. We shall hereafter refer to this cursor as the **automatic cursor**.

We also extended the cursor warping idea presented by Benko, et al. [7], which we will call the **Benko technique** hereafter. Benko introduces the idea of warping one's cursor from monitor to monitor by pressing mouse buttons or a combination of keys on the keyboard. Cursor warping is the reposition of one's cursor to an adjacent monitor. By pressing one button on the mouse, the cursor is warped to a monitor on the left and pressing another button the cursor is warped to the right. In the event that the user tries to warp the cursor beyond the edge of the display the cursor wraps around to the other side of the display as if the right and left edges of the display were connected like a cylinder. We did not introduce the idea of wrapping the display with the other cursor techniques in order to reduce experimental complexity.

For the cursor warping technique each monitor is considered its own dependent display where the cursor is not allowed to go between bezels. For instance, if one were to try to click an object that is on the same monitor as the cursor then one would use the mouse normally. However, if one were to try to click an object on another monitor, one would have to warp the cursor to that other monitor then use the mouse regularly within that monitor.

We extended Benko's cursor warping idea from four (4x1) monitors to twenty-four (8x3) monitors. We also extended the idea of cursor warping to include warping up and down to monitors above or below the current monitor. As well as having the furthest left and right parts of the display connected we also connected the top and bottom, creating a virtual toroid. Also, we used frame-relative positioning as suggested by Benko. For reference, frame-relative positioning is warping the mouse in the same relative position from one monitor to another. For example, if the cursor were at the top left corner of a monitor then if the cursor were warped to another monitor, the cursor would be located at the same relative top left corner of the second monitor.

Instead of using four different buttons on the mouse for navigating as well an additional button for clicking in the tasks we used the arrow keys on the keyboard. Besides alleviating participants of the arduous task of quickly clicking between a number mouse buttons while also using the left-click button for tasks we found that participants were already familiar with the keyboard's arrow keys and needed little time to adapt to using them.

The last cursor used in the experiment was the **control cursor**. We did not modify any setting to the standard cursor for this cursor. To see how the control cursor compares to the high-speed cursor see table 1. To summarize the different techniques see table 2.

Cursor Technique	Summary
Control	Standard cursor.
High-Speed (DSS 1)	Large cursor with a high acceleration.
Manual (DSS 2)	Either standard cursor setting or large cursor with a high acceleration. User manually toggles between the two cursors.
Automatic (DSS 3)	Either standard cursor setting or large cursor with a high acceleration. Algorithm automatically toggles between the two cursors.
Benko (Cursor Warping)	Cursor warps from monitor to monitor using arrow keys. Standard cursor functionality within each monitor. Extended from [7].

Table 2. This table summarizes the different attributes of each cursor technique.

EXPERIMENT SETUP

The goal of this experiment is to show that a dynamic size and speed cursor is better than other techniques, specifically the standard cursor mouse warping. The independent variables are:

- Cursor type
- Target size
- Target distance
- Task type (e.g. simple click, simple drag and drop, and complex drag and drop)
- Participant gender

Our dependent variable is:

- Performance time (i.e. How long it took the participants to click the targets and drag and drop the targets)

In the experiment there were 15 participants. The average age of the participants was 24 years old with a range from 20 to 30. None of the participants were color blind; all had normal or corrected normal vision; 8 were male; 7 were female.

After the experiment all participants filled out a general questionnaire about their age, gender, the cursor technique they preferred, and the cursor technique they felt was the easiest to use. The participants were a mix of undergraduate and graduate students from a mix of majors.

A repeated Latin Square design was used for counterbalancing. Also, each participant was required to spend a minimum amount of time practicing each technique before performing the tasks for baselining purposes. A with-in subject design was used to reduce variability between participants. Each participant performed three different isomorphic tasks for each of the five different interactive techniques. The tasks were all given in the same order for each technique.

Participants used a wireless keyboard and wireless optical three-button mouse. The reason for the wireless devices was

due to the distance of the participants from the computing cluster, which was behind the display.

Tasks

Three types of tasks were used in the experiment: simple click, simple drag and drop, and complex drag and drop. The first two tasks were based on Fitt's law. Fitt's law roughly states that for simple movements, movement time is a logarithmic function of distance when target size is held constant, and that movement time is also a logarithmic function of target size when distance is held constant.

In the first task, the participants were presented a sequence of twenty-four squares of three sizes, where only one square was present on the display at a time. After each successful click on a square another square would immediately appear in another location on the display. Similarly, the second task involved dragging a sequence of twenty-four solid circles to their perspective outlines.

The third task involved higher order thinking than the first two tasks. Instead of clicking or dragging and dropping objects as quickly as possible, participants were presented with twenty-four solid shapes along with their corresponding twenty-four outlines at once. Different shape/color combinations were presented so that the participants could easily tell each shape from the others. Figure 2 shows a participant performing the third task.

All three tasks used three different sized shapes: small (16x16 pixels), medium (40x40 pixels), and large (150x150 pixels). Each shape was represented eight times in each task. The small shape was the size of the minimize, maximize, and close icons at the top of applications in Linux and Windows. The medium-sized shape was the size of a standard icon on a desktop. The large-sized shape was four times the size of the medium-sized shape. The different sized shapes were interspersed equally in the different tasks.

Each of the fifteen (three tasks X five cursor techniques) isomorphic tasks was created prior to the experiment. All shapes for each of the tasks were positioned randomly on the display and then saved to a file. However, shapes that fell on a bezel were nudged to one side.

All tasks were tracked automatically by the computer. After each task was completely successfully the computer automatically logged and time stamped the performance for each successful click or drag and drop.

QUANTITATIVE RESULTS

This section reports the different quantitative results found from running the experiment. All statistical analysis was performed using SAS Institute's JMP application. Standard ANOVA analysis was performed.

For performance time, for all three tasks, we found a main effect of cursor technique ($p = 0.006$). Using Tukey's HSD post-hoc test we found that specifically the high-speed cursor outperformed Benko's technique of cursor warping. In addition, we found that the same general pattern of performance as seen in figure 3 was repeated in all three tasks individually - see each task subsection.

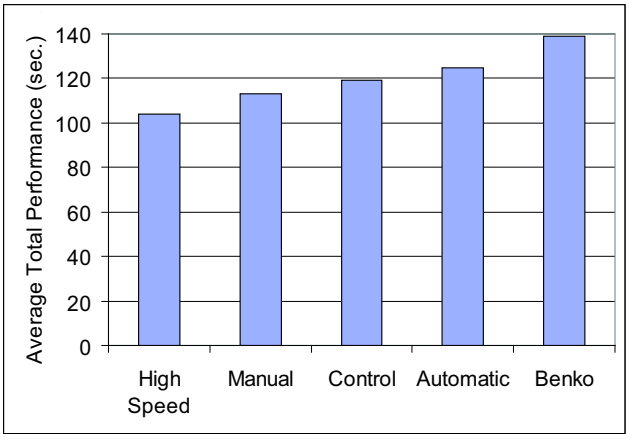

Figure 3. A chart showing the summary results of all three tasks. The high-speed cursor (DSS 1) was significantly faster than the Benko technique.

Although we had hypothesized that the DSS cursors would be faster than the control cursor and cursor warping we were surprised by the results for several reasons. First, we did not hypothesize that the control cursor would be overall faster than cursor warping. Second, we had postulated that the manual cursor or the automatic cursor would be faster than the high-speed cursor, but we were wrong.

It appears that participants were able to adapt to the higher speed of the high-speed cursor better than we had anticipated. Although most participants did not prefer the high-speed cursor (see section), in general, it outperformed all other techniques.

One possible explanation for the high-speed cursor performing faster than the manual cursor is that it took time for participants to press the space bar. Second, there were many times when participants would accidently toggle to the wrong sized cursor and then quickly toggle back. Third, by using two hands instead of one it is possible that participants had a higher cognitive load with the manual cursor than the high-speed cursor. Fourth, participants had to adapt to two different acceleration speeds and two different sizes with the manual cursor where with the high-speed cursor they only had to adjust their behavior to one acceleration speed and one size.

The most probable reason for the automatic cursor performing worse than the high-speed cursor and the manual cursor is an implementation issue. Participants felt comfortable with the automatic cursor by the end of the practice session, however, they often felt frustrated by it when it came time to actually perform timed tasks. They often explained that it did not accurately predict what they wanted to do. We observed participants jiggling the physical mouse back and forth to toggle the size of the cursor.

It is possible that the additional cognitive load of using two hands was amplified in the cursor warping technique. Not only did participants have to use two hands, but they also had to use four different arrow keys to navigate. It appears that the high-speed cursor was able to go greater distances faster than the cursor warping technique even with display wrapping as an aide.

Clicking Task

The first task involved clicking a sequence of twenty-four squares that appeared in random positions on the screen. Analysis of the data showed a gender bias ($p < 0.01$), an interaction between distance to targets and size of the target ($p < 0.0001$), and an interaction effect between distance to targets and cursor technique used ($p < 0.0001$).

Figure 4 shows an interesting relationship between cursor technique and distance to target. Although in general the best performance was found in the high-speed cursor (see figure 3), that general trend was not always true at all distances. As the distance to target was small, such as the points in bucket 0 (less than 1000 pixels), the control cursor appears to outperform all other cursor techniques ($p = 0.0012$). However, as the distance increases, an interaction effect is seen that the control cursor performs increasingly worse to finally have the worse performance at the greatest distances. To illustrate the point, the difference in performance between the first to last point for the control cursor is 2.80 seconds, where the difference between the first to last point for the high-speed cursor is 0.74 seconds.

Performing post-hoc analysis shows that all cursor techniques were statistically significant from each other except for the manual and control cursor. In other words, the high-speed cursor was faster than *all* other techniques for the clicking task.

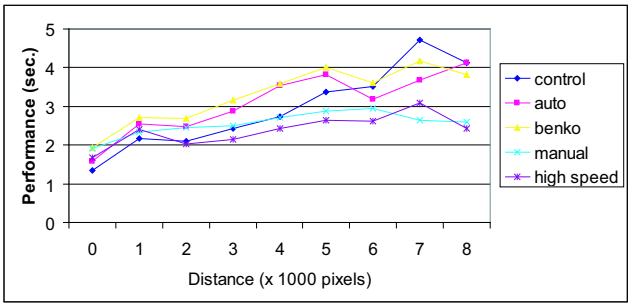

Figure 4. Chart showing how each cursor technique varied in performance as the distance to target increased from left to right for the clicking task. Control cursor is fastest for short range targets, but is overtaken by other techniques for long range targets. High-speed and manual cursors are fastest for long range targets.

With the gender bias we found that males performed 8% faster than females, on average 3.1 seconds faster. In other words, for the task of clicking a sequence of objects, we found that men were able to click the twenty-four squares faster than women, regardless of what cursor technique was used.

Figure 5 shows the relationship of target distance to target size. The y axis represents the average performance time in seconds for each square that was clicked. The x axis represents distance separated into discrete buckets of 1000 pixels. For example, bucket 0 represents all the squares that were clicked from 0 pixels to 999 pixels in distance from the cursor's origin. This should not be interpreted as how far the cursor actually traveled as this differed per participant, but the shortest distance between where the cursor started and the target. All other charts that show distance in this paper use the same mapping scheme of distance buckets and can be interpreted similarly.

The longest distance possible for a cursor to start away from a target is the diagonal from one far corner of the display to another. In the case of a typical desktop monitor (1280x1024), using the equation $(a^2 + b^2 = c^2)$, the longest distance possible is 1639 pixels, which falls into the second bucket. However, for the display used for this experiment, the resolution was 10,220x3030, which means that the longest distance from one far corner to another is 10,690 pixels, which falls into the last bucket. However, as all targets were prepossessed to random locations by a random generator, the largest distance between origin of cursor and target for the display used was 9,336 pixels, and the shortest distance was 322 pixels.

Figure 5 shows that the main cause for the interaction effect of target size to target distance occurs at the last two points of each curve on the chart. A note to the reader: due to an empirical error, we were not able to reliably record distances of greater than 9000 pixels for the clicking task and, therefore, do not report them. We did not have the error for the simple drag and drop task.

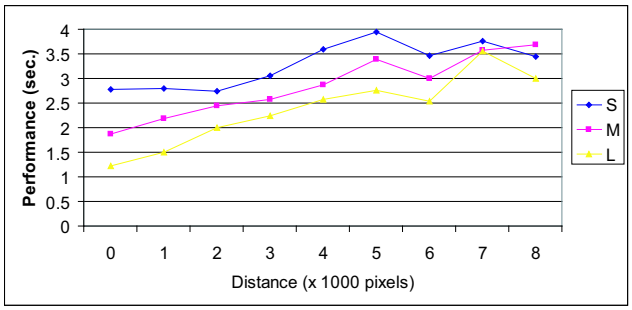

Figure 5. Chart showing the relationship of distance to targets and target size for the clicking task. Performance time increases with distance to target.

Simple Drag and Drop Task

The second task required the participants to perform a sequence of twenty-four simple drag and drop operations. Similar to the first task, we automatically recorded every instance where a participant successfully dragged a solid circle to its respective outline, where only one solid/outline circle combination was shown at once.

Upon analyzing the data, we found an interaction effect between distance to target and gender of participant ($p < 0.0001$), an interaction effect between distance to target and size ($p < 0.0001$), and an interaction effect between distance to target and cursor technique ($p < 0.0001$).

Figure 6 shows the same trend of performance that is seen in the clicking task. Specifically, there appears to be trendline for each cursor technique. As the distance from where the cursor starts to the target is slight (less than 1000 pixels), the control cursor appears to have the best performance and the high-speed cursor has the worst performance. However, as the distance to the target increases the opposite is true.

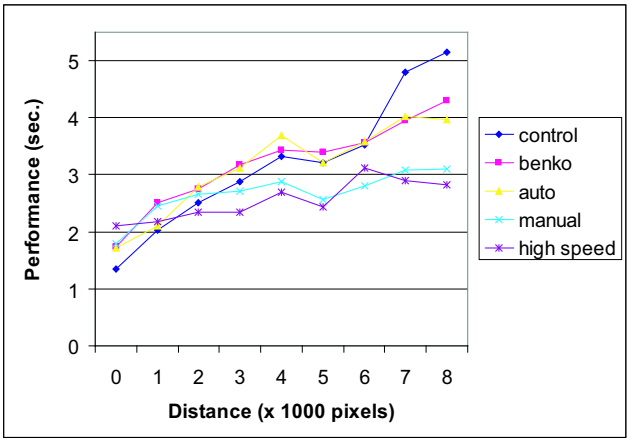

Figure 6. Chart showing how each cursor technique varied in performance as the distance to target increased from left to right for the simple drag and drop task.

Once again, post-hoc analysis shows that the high-speed cursor is statistically faster than all the other techniques (with the exception of the manual cursor).

Males performed 12% faster on the drag and drop task than females, on average 13.7 seconds faster. Looking at figure 7 one can see a consistent trend of males outperforming females. It appears that the reason for the interaction effect is found at the last data point for each gender where the lines overlap at distances of greater than 9000 pixels in distance. Other than that one distance point, it appears that men are consistently faster at dragging and dropping objects than women, similar to the first task (clicking).

Figure 8 shows the comparison of distance to target and size. As the distance becomes greater the longer it takes to complete the task. However, at the same distances, the smaller the target the longer it takes. Except for the first small and medium data point, where it appears to be the reason for the interaction, there is a clear separation of curves for each target size, regardless of cursor technique used.

Complex Drag and Drop

The third task differed from the first two tasks in two main ways. First, all the shapes were presented to the user at once, not in sequence. Second, when a solid shape had successfully been dragged to the correct outline, it did not disappear, but simply turned white to indicate a correct match to the user. The white shapes were left on the screen intentionally as a form of distractor to increase the cognitive load on the user.

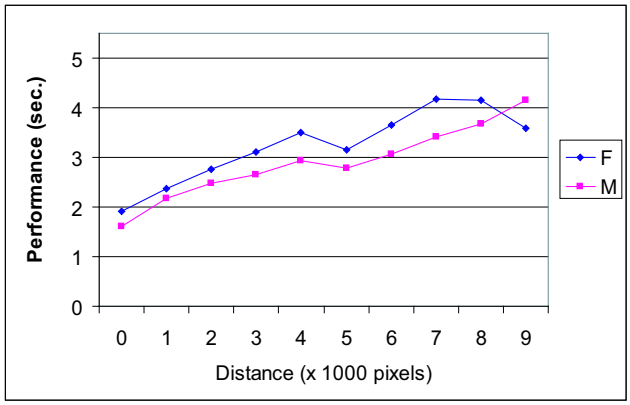

Figure 7. Chart showing the differences in performance for the two genders as the distance to target increased from left to right for the simple drag and drop task. In general, men outperform women by 12%.

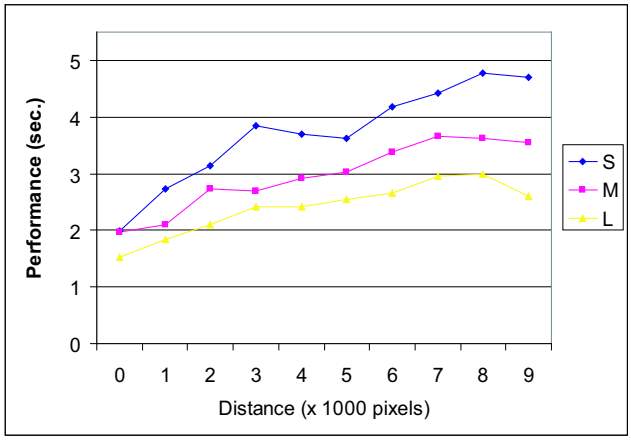

Figure 8. Chart showing the relationship of distance to targets and target size for the simple drag and drop task.

In the third task we also found a main effect of cursor technique to performance time ($p < 0.001$). Figure 9 shows the overall trend of performance time with the high-speed cursor performing the fastest, followed by the control cursor, manual cursor, automatic cursor, and cursor warping. This result is similar to the overall result (see figure 3) with the exception that the control cursor performing slightly faster on average (2.1 seconds) than the manual cursor. Specifically, performing post-hoc analysis shows that all cursor techniques were faster than the Benko technique.

Unlike the first two tasks, we did not find a gender bias for the third task ($p = 0.71$). One explanation for this result is that for the first two tasks (clicking and simple drag and

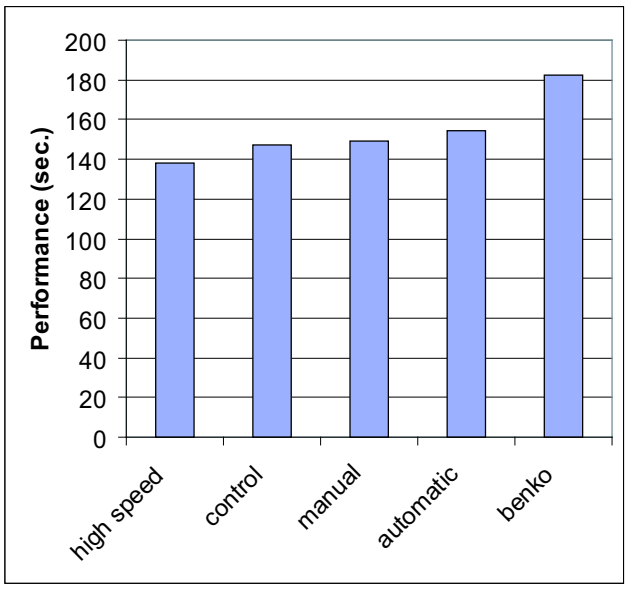

Figure 9. Bar chart showing average performance time of the different cursor techniques used in the complex drag and drop task.

drop) little cognitive load was placed upon the participants and one could say that we were measuring the reflexes of the individuals. That said, it is possible that our results indicate that men are slightly faster with reaction times or fine motor skills.

PREFERENCE

The techniques most preferred and thought to be easiest to use by participants were the manual cursor and the cursor warping. The following table shows the preference of the participants listed by technique.

Cursor Technique	Technique Preferred	Easiest to Use
Manual (DSS 2)	7	5
Cursor Warping	5	6
High-speed (DSS 1)	2	2
Automatic (DSS 2)	1	0
Control	0	2

Table 3. Table showing participant preference and ease of use for each technique.

Although the high-speed cursor was found to be consistently faster on all tasks regardless of size of the target, participants preferred the manual cursor and found it easier to use.

Even though the manual cursor had slightly lower performance time than the high-speed cursor, participants indicated that the different sizes of the cursor made them think that they could control it better. Participants indicated that when the cursor size was small that they felt that it was easier to use for small movements. Likewise, when the cursor was big, they felt that it was easier to move quickly. As one participant explained, Ït is natural for a big cursor to go fast and a small cursor to go slow."

The cursor warping technique was the second preferred and the first easiest to use even though it was consistently the slowest cursor technique. Participants that preferred it often cited that they felt it was the fastest technique. However, looking at individual performance, there was not a single case where cursor warping was faster than any other technique for any task. Another reason that participants preferred the cursor warping technique was the ability to wrap around from one side of the display to another (i.e. move the cursor from the right most to left most part of the display or from the top most to the bottom most) in a single key stroke.

CONCLUSION

Although the high-speed cursor had the best overall performance, the manual cursor was the most preferred technique and whose performance was only slightly slower. It also appears that the main reason why participants preferred cursor warping was due to display wrapping (virtual toroid), not due to warping.

To summarize the results of our experiment we have include the following list:

- The high-speed cursor performed consistently 15% faster than the control cursor and 35% faster than cursor warping.
- Participants felt that the manual cursor seemed more nnaturalthan the high-speed cursor. In other words when the manual cursor was large, they felt that it was intuitive that it should move quickly and when the manual cursor was small, they felt that it was intuitive that it should move slowly.
- Participants liked display wrapping (virtually connected the side of the display). However, it is not clear if participants liked display wrapping due to it being novel or whether it truly helped them perform faster.
- For clicking and simple drag and drop operations on large, high-resolution displays there appears to be a gender bias of men performing 8% and 12% better respectively for the two tasks. However, there does not appear to be a gender bias for the higher thinking task that required a degree of strategy.

In conclusion, when using a mouse as the input device for large, high-resolution displays, we suggest that the dynamic size and speed cursor be used over the standard cursor settings or cursor warping. We would like to point out that the increase in time from short distance to long distance targets took from about 1.5 seconds on small displays to about 3 seconds on large screens with the high-speed cursor. That is a 2x increase in performance time for a 24x increase in screen size. In contrast, the control cursor went from about 1 second to about 5 seconds for the same targets.

FUTURE WORK

The results of this paper have raised a number of issues. First, do the results of this paper transfer to different types of mice, like the gyro mouse (3 degree of freedom (DOF) tracking mouse) and the trackball mouse?

Second, would a better implementation of the automatic cursor increase user performance over the nave approach of the high-speed cursor? We hypothesize that a smart cursor that changes the speed and size of the cursor dynamically would perform better than a simple bimodal cursor such as the high-speed and the control cursor.

Third, would the high-speed cursor continue to have the best performance on more sophisticated tasks than the simple ones in this experiment? We purposely chose simple tasks in this experiment. Generally speaking, if a new technique does poorly with simple tasks it will also do poorly in complicated tasks. However, it is not necessarily true that a technique that performs well in simple tasks will also perform well in more complex tasks.

Fourth, how do these, and similar techniques, compare when display wrap is introduced. Participants often cited the cursor warping technique as their favorite, or the most easily used, due to the wrap around effect. As cited earlier, the reason for not adding wrap around in this preliminary experiment was to reduce complexity and, therefore, make the results easier to understand.

Fifth, we only looked at the time it took participants to succeed, not how many tries they made. Future work should include studies on the accuracy of different cursor techniques.

ACKNOWLEDGEMENTS

This research is partially supported by the National Science Foundation grant #CNS- 04-23611. This study was also supported and monitored by the Advanced Research and Development Activity (ARDA) and the Department of Defense. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Geospatial-Intelligence Agency or the U.S. Government.

REFERENCES

1. Mark Ashdown, Kenji Oka, and Yoichi Sato. Combining head tracking and mouse input for a gui on multiple monitors. In *Extended abstracts of CHI '05*, pages 1180–1191, 2005.

2. Robert Ball and Chris North. An analysis of user behavior on high-resolution tiled displays. In *Interact 2005 Tenth IFIP TC13 International Conference on Human-Computer Interaction*, pages 350 – 363, 2005.

3. Robert Ball and Chris North. Effects of tiled high-resolution display on basic visualization and navigation tasks. In *Extended abstracts of CHI '05*, pages 1196–1199, 2005.

4. Patrick Baudisch, Edward Cutrell, K. Hinckley, and R. Gruen. Mouse ether: Accelerating the acquisition of targets across multi-monitor displays. In *Extended Abstracts in CHI 2004*, pages 1379 – 1382, Vienna Austria, April 2004.

5. Patrick Baudisch, Edward Cutrell, Dan Robbins, Mary Czerwinski, Peter Tandler, Benjamin Bederson, and Alex Zierlinger. Drag-and-pop and drag-and-pick: techniques for accessing remote screen content on touch- and pen-operated systems. In *Proceedings of Interact '03*, pages 57 – 64, 2003.

6. Patrick Baudisch, Edward Cutrell, and George Robertson. High-density cursor: A visualization technique that helps users keep track of fast-moving mouse cursors. In *Proceedings of Interact 2003*, pages 236 – 243, Zurich, Switzerland, 2003.

7. Hrvoje Benko and Steven Feiner. Multi-monitor mouse. In *Extended Abstracts of CHI '05*, pages 1208 – 1211, Portland, Oregon, April 2005.

8. Chromium. http://chromium.sourceforge.net/.

9. Mary Czerwinski, Greg Smith, Tim Regan, Brian Meyers, George Robertson, and Gary Starkweather. Toward characterizing the productivity benefits of very large displays. In *Proceedings of Interact 2003*, 2003.

10. Mary Czerwinski, Desney Tan, and George Robertson. Women take a wider view. In *Proceedings of CHI '02*, pages 195–201, 2003.

11. Dmx (distributed multihead x). http://dmx.sourceforge.net/.

12. Jrg Geiler. Shuffle, throw or take it! working efficiently with an interactive wall. In *CHI '98 conference summary on Human factors in computing systems*, pages 265 – 266, 1998.

13. Glut - the opengl utility toolkit. http://www.opengl.org/resources/libraries/glut.html.

14. Tovi Grossman and Ravin Balakrishnan. The bubble cursor: Enhancing target acquisition by dynamic resizing of the cursors activation area. In *Proceedings of CHI '05*, pages 281 – 290, Portland, Oregon.

15. Franois Guimbretire, Maureen Stone, and Terry Winograd. Fluid interaction with high-resolution wall-size displays. In *Proceedings of UIST 2001*, pages 21–30. ACM, 2001.

16. James Jeng-Weei Lin, Henry B.L. Duh, Donald E. Parker, Habib Abi-Rached, and Thomas A. Furness. Effects of view on presence enjoyment, memory, and simulator sickness in a virtual environment. In *Proceedings of IEEE Virtual Reality*, 2002.

17. Jock D. Mackinlay and J. Heer. Wideband displays: Mitigating multiple monitor seams. In *Proceedings of CHI '04*, pages 1521 – 1524, Vienna, Austria, 2004.

18. Elizabeth D. Mynatt, Takeo Igarashi, W. Keith Edwards, and Anthony LaMarca. Flatland: New dimensions in office whiteboards. In *Proceedings of CHI '99*, pages 346–353, 1999.

19. Dan R. Olsen and Travis Nielsen. Laser pointer interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 17 – 22, 2001.

20. Opengl. http://www.opengl.org.

21. J. Karen Parker, Regan L. Mandryk, Michael N. Nunes, and Kori M. Inkpen. Tractorbeam selection aids: Improving target acquisition for pointing input on tabletop displays. In Maria Francesca Costablie and Fabio Patern, editors, *Proceedings of INTERACT 2005*, pages 80 – 93, 2005.

22. Elin R. Pedersen, Kim McCall, Thomas P. Moran, and Frank G. Halasz. Tivoli: An electronic whiteboard for informal workgroup meetings. In *Proceedings of CHI '93*, pages 391–398, 1993.

23. Terry Simmons. What's the optimum computer display size? *Ergonomics in Design*, pages 19–25, Fall 2001.

24. Desney Tan, Mary Czerwinski, and George Robertson. Women go with the (optical) flow. In *Proceedings of CHI '03*, pages 209–215, 2003.

25. Desney Tan, Darren Gergle, Peter G. Scupelli, and Randy Pausch. With similar visual angles, larger display improve spatial performance. In *Proceedings of CHI '03*, pages 217–224, April 2003.

26. Desney Tan, Darren Gergle, Peter G. Scupelli, and Randy Pausch. Physically large displays improve path integration in 3d virtual navigation tasks. In *Proceedings of CHI '04*, pages 439–446, April 2004.

27. Desney Tan, Jeanine K. Stefanucci, Dennis Proffitt, and Randy Pausch. The infocockpit: providing location and place to aid human memory. In *Proceedings of PUI '01*, pages 1–4, 2001.

28. Grant Wallace, Peng Bi, Kai Li, and Otto Anshus. A multi-cursor x window manager supporting control room collaboration. Technical Report TR-707-04, Princeton University, 2004.