Pointer Warping in Heterogeneous Multi-Monitor Environments

Hrvoje Benko Steven Feiner

Department of Computer Science
Columbia University
New York, NY
{benko, feiner}@cs.columbia.edu

Abstract

Warping the pointer across monitor bezels has previously been demonstrated to be both significantly faster and preferred to the standard mouse behavior when interacting across displays in homogeneous multi-monitor configurations. Complementing this work, we present a user study that compares the performance of four pointer-warping strategies, including a previously untested frame-memory placement strategy, in heterogeneous multi-monitor environments, where displays vary in size, resolution, and orientation. Our results show that a new frame-memory pointer warping strategy significantly improved targeting performance (up to 30% in some cases). In addition, our study showed that, when transitioning across screens, the mismatch between the visual and the device space has a significantly bigger impact on performance than the mismatch in orientation and visual size alone. For mouse operation in a highly heterogeneous multi-monitor environment, all our participants strongly preferred using pointer warping over the regular mouse behavior.

Keywords: Multi-monitor, mouse pointer, interaction technique, distributed display environments.

CR Categories: H.5.2. [User Interfaces]: Graphical User Interfaces, Input Devices and Strategies.

1 Introduction

Multi-monitor display configurations can be characterized as either homogeneous or heterogeneous. The most frequently encountered examples are *homogeneous*, where two or more displays of the same size, resolution, and relative orientation to the user, are tiled next to one another. When displays of different size, resolution, or orientation are used together, they form a *heterogeneous* multi-monitor configuration, such as that shown in Figure 1.

Both homogeneous and heterogeneous configurations extend the available desktop space. However, enlarged distances, coupled with the need to cross individual monitor edges (bezels), present difficulties to regular pointer interaction. Several researchers [1-5, 12, 16, 17] have noted serious drawbacks with standard mouse interactions across displays in multi-monitor configurations. In homogeneous configurations, most of the problems arise from the exaggerated distances that the mouse has to traverse and the path discontinuities that are caused by monitor bezels. In heterogeneous configurations, these problems are further exacerbated by the discrepancies between the device space and visual space behavior

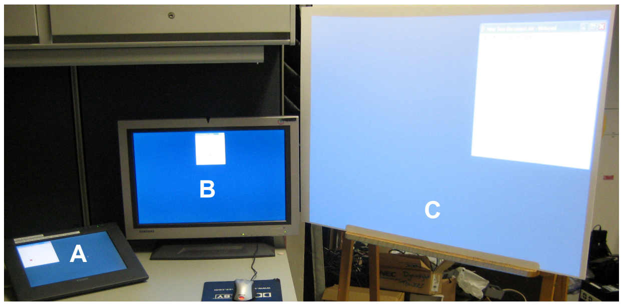

Figure 1: Experimental heterogeneous multi-monitor environment consisting of a small low-resolution near-horizontal display (A), a medium-size high-resolution vertical display (B), and a large low-resolution vertical display (C).

of the mouse (Figure 2, left). We use the term *device space* to describe the system's perspective of the desktop space, where the number of pixels determines the area. This is what the computer graphics community calls device coordinates. In contrast, *visual space* is the user's view of the desktop space, which is determined by the physical display size. Furthermore, it is possible to consider a *perspective space* where the distance and orientation between the user and the displays affect how the user perceives the presented information.

Pointer warping was introduced simultaneously and independently by us [5] and Ashdown and colleagues [1] as an alternative to standard monitor bezel traversal. Pointer warping is defined as *instantaneous relocation of the cursor to the desired virtual frame* (e.g., monitor screen), and such techniques attempt to reduce the need to traverse monitor bezels through mouse motion, while allowing conventional mouse interactions within each screen. We presented a set of Multi-Monitor Mouse (M^3) techniques [5], which evaluated pointer warping in a homogeneous multi-monitor environment and showed significant improvements when traversing two or more monitor bezels.[1] Performance improvements of up to 29% were achieved when traversing three monitor bezels.

In this paper, we extend our previous M^3 research on pointer warping by implementing the improvements suggested in the earlier study [5] and evaluating the performance of several warping techniques (including a previously untested strategy) in a heterogeneous multi-monitor environment.

[1] Virtual frames in M^3 were not restricted to entire screens, but could be arbitrary user-defined upright rectangular areas; however, our evaluation tested only full-screen frames.

Graphics Interface Conference 2007
28-30 May, Montréal, Canada

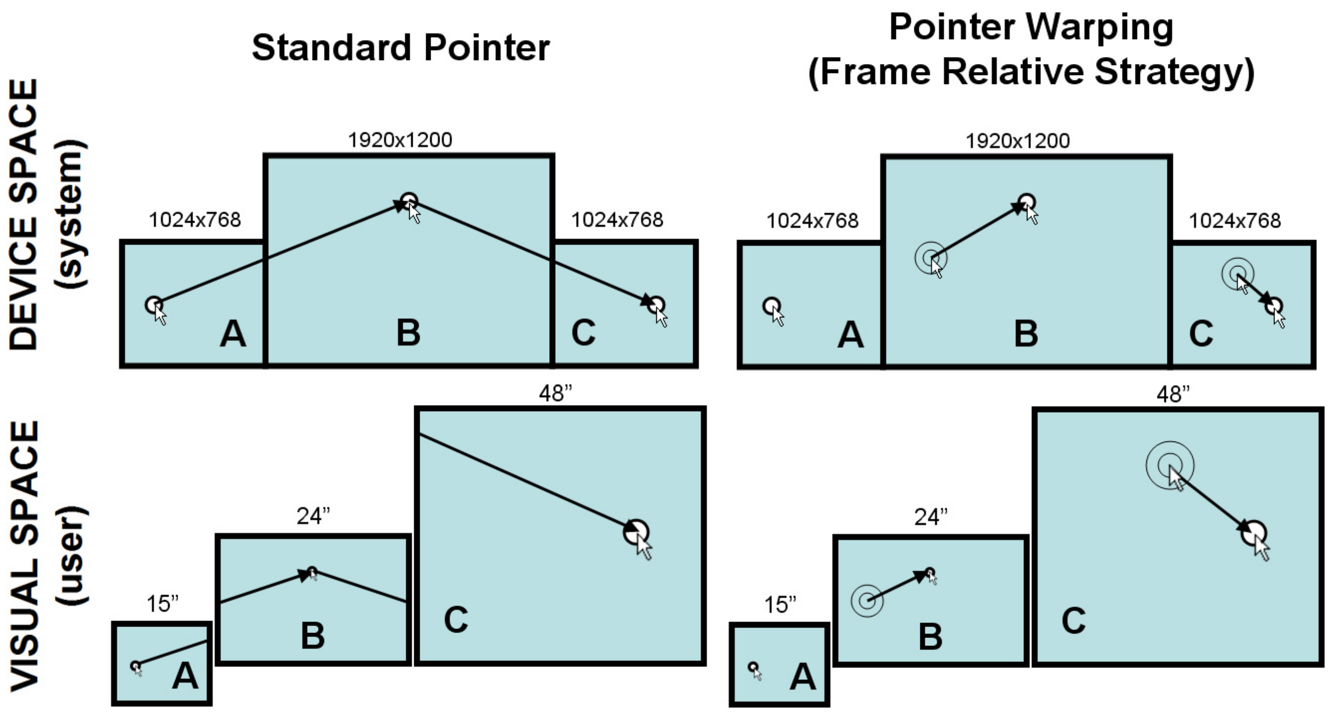

Figure 2: The differences in device space (top) and visual space (bottom) representations of our experimental setup. Traversing a continuous device space path (top left) with a standard pointer can be a cognitively demanding task in visual space (bottom left). Pointer warping aids the user by removing the need to traverse the bezel (right). Concentric "sonar" circles help increase the visibility of the cursor after the warp by highlighting the destination.

2 Related Work

A substantial amount of work has been done to alleviate the effects of bezel traversal and enhance pointer interaction across multiple displays. Baudisch and colleagues accelerated the mouse cursor to ease access to distant locations in *high-density cursor* [4]. A complementary effect has been achieved by bringing distant targets closer to the current cursor location in *drag-and-pop* [3]. Forlines and colleagues developed *HybridPointing* [8], which lets the user switch easily between absolute and relative pointing to enable access to distant and close targets on a large display. Reduction of discontinuities caused by mismatched monitor alignment, bezels, and resolutions has been explored in *mouse ether* [2] and *wideband displays* [12]. Mouse ether solved the problem of mismatched visual and device space, by adjusting the pointer speed on all monitors so that the pointer moves at a consistent visual speed irrespective of the monitor resolution. Nacenta and colleagues [13] took a more general approach to display position and orientation differences by displaying a perspectively corrected pointer based on the position of the user's head; however, their approach relied on 3D position tracking of all displays in the environment as well as the user's head.

M^3 [5] introduced several pointer warping strategies for placing the pointer on the target display after a user-initiated warp in a homogeneous multi-monitor configuration. In addition, M^3 explored using mouse buttons, mouse location, and head orientation to trigger the warp. Independent of M^3, Ashdown and colleagues [1] presented another implementation of homogeneous multi-monitor pointer warping using head orientation.

Interactions that warp the pointer closer to a target location have been explored on a single monitor in combination with eye gaze (e.g., eye gaze interaction [15] and MAGIC pointing [18]) or hand gestures (e.g., flick [7]). Tan and colleagues [17] explored the effects of visual separation between displays that varied in size and depth, while Su and Bailey [16] examined various horizontal distance and angle arrangements between the displays and their effect on users' performance in a multi-monitor environment. In work on *Semantic Pointing*, Blanch and colleagues [6] provided valuable insight into decoupling the visual and device space and showed how that can be used to provide assistance in mouse selection.

3 Frame Switching Techniques

To test the effectiveness of pointer warping in heterogeneous multi-monitor configurations, we chose the *mouse button* frame switching technique that showed the biggest performance gains (up to 29%) and received overwhelming user preference (7 out of 8 participants) in the previous M^3 study [5] and compared it to a new technique, *head-orientation–mouse* switch. This new technique extends the original M^3 *head-orientation* switch by incorporating the improvements suggested by the study participants.

The *mouse button* (MB) switch trigger is issued by pressing one of the two side buttons (XButtons) on a five-button mouse. The top side button cycles through the monitors forward (clockwise) and the bottom side button cycles backwards (counter-clockwise). The virtual frames form a loop, making it possible to cycle through all the screens using just one of the buttons.

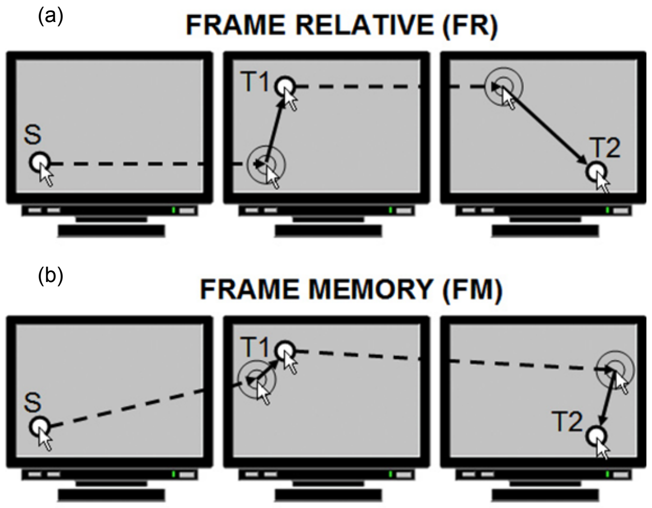

Figure 3: A comparison of two pointer placement strategies tested in the experiment (shown here in a homogeneous environment). Traversing between locations S, T1, and T2 using: (a) frame-relative (FR) and (b) one of many possible frame-memory (FM) scenarios. Dashed lines indicate warping; solid lines indicate conventional movement. Note that the position of the pointer after the warp in the FM strategy is the last position of the cursor on that frame (i.e., the position that the pointer occupied before it warped out of that frame).

The *head-orientation–mouse* (HEAD) switch is a hybrid technique that combines head orientation measurement (for determining the screen at which the user is currently looking) with a side mouse button trigger (to trigger a warp to that screen). We measure the user's head orientation with a 3DOF orientation sensor mounted on a pair of headphones. The original M^3 *head-orientation* switch was the close second-best switching alternative [5], but suffered from the "Midas touch" problem [10], causing the cursor to warp across monitors even when the user just wanted to glance over without switching monitor focus. Identical problems were present in the head-orientation-based implementation of pointer warping by Ashdown and colleagues [1]. To eliminate such spurious switching, we have introduced a mouse button trigger, which adds an additional click overhead while switching, but still reduces the number of clicks compared with the mouse button switch technique.

4 Pointer Placement Strategies

We were interested in comparing two strategies for locating the cursor on the target screen after the frame switch. Therefore, we decided to compare the winning strategy in the homogeneous multi-monitor case (*frame-relative*) [5] with a previously untested strategy (*frame-memory*).

Frame-relative (FR) placement works by translating the pointer to the next frame at the same location relative to the new frame's upper left corner as it was relative to its old frame's upper left corner (Figure 3a). This strategy essentially collapses the entire desktop space into one frame of mouse movement and is the only M^3 strategy in which the effect of pointer movement prior to the frame switch will not be negated by the switch itself.

Frame-memory (FM) placement (called *frame-dependent* in the previous M^3 work [5]) considers all frames as completely independent spaces. It stores the last location of the cursor in each frame, and warps the incoming cursor to that location (Figure 3b). Thus, the last position of the cursor when the user warps out of the frame, becomes the starting location when the user eventually warps back to that frame. We had not tested FM in our previous homogeneous multi-monitor experiments, due to the presumed high short-term memory load imposed by having to remember each frame's cursor position [5]. However, we hypothesized that this strategy would work well in a heterogeneous setup, where the physical differences between monitors might make it easier for the user to remember each frame's cursor position.

5 User Study

To evaluate the performance of these display switching and pointer placement methods in a heterogeneous multi-monitor environment, we conducted a user study with ten right-handed participants (seven male, three female, ages 21–27), all unfamiliar with our pointer warping techniques and with no connection to our lab. The participants were recruited by mass email to students in our department, and received a small monetary compensation for their participation.

5.1 Setup

The experiment was performed on a PC running Windows XP Pro, with two ATI Radeon 9800 and 9000 graphics cards. The virtual desktop was extended over three displays of different orientation, resolution, and size (Figure 1). From the system's perspective, the displays were aligned at the bottom (Figure 2). The displays were arranged in a semicircle (approximate radius 80cm)

Position	Left A	Middle B	Right C
Type	Wacom Cintiq 15X LCD	Samsung SyncMaster 240T LCD	NEC WT600 Projector
Size	12"×9" (15" diag.)	20.5"×12.75" (24" diag.)	38"×29" (48" diag.)
Resolution	1024×768	1920×1200	1024×768
Visual Pixel Size	0.28mm	0.24mm	1mm
Orientation	Near-horizontal	Vertical	Vertical

Table 1: Displays used in the study.

Screen Transition	A–B	A–C	B–C
Visual Area Mismatch	2.4	10.2	4.2
Visual–Device Space Mismatch	0.85	3.57	4.2
Orientation Mismatch	73°	73°	0°
Bezel Crossings	1	2	1

Table 2: Display transition characteristics in our experiment.

around the participant's seat and ordered by increasing diagonal size from the left: A (15"), B (24"), and C (48"). Table 1 summarizes the relevant characteristics of the displays.

Head orientation was tracked by a set of headphones on which was mounted an InterSense InertiaCube2 head orientation tracker. These were worn by participants throughout the entire experiment to eliminate the potential confound of wearing them only during head-tracking conditions. Mouse pointer speed and acceleration were kept at the default Windows XP setting.

In Table 2, we summarize the various types of mismatch present when traversing between monitors in our experiment. *Visual area mismatch* represents the ratio between the physical areas of the two screens and provides us with a simple metric of the visual size difference between displays. (Note that a more complete metric would take into account the distances between the displays and the user and the solid angle subtended by these displays in the user's visual field.)

Visual–device space mismatch is the ratio of visual pixel sizes between displays. It is important to note that all screens were used at their native resolution, which caused the largest adjacent monitor mismatch between the visual and the device space to occur between B and C (i.e., pixels on C are 4.2 times larger than on B).

Orientation mismatch is the tilt (pitch) angle difference between screens. The left display (A) was oriented at a near-horizontal angle of 17^o with respect to the desk surface, as per ergonomic guidelines suggested by the manufacturer (Wacom). Thus, A was offset by 73^o about the horizontal, relative to the other two displays. Note that arranging all displays in a semicircle around the participant ensured equal distance to each display. Therefore, we do not consider differences in yaw as orientation mismatch. Su and Bailey [16] found that for the optimal performance for a stationary user, the vertical displays should not be positioned in the same plane, but positioned at an angle of up to 45° with respect to each other to ensure equal visual angles and minimal amount of distortion. Our setup follows their guidelines; however, note that Nacenta and colleagues [13] take a more general approach to display position and orientation differences.

5.2 Method

We decided to test standard unassisted mouse movement (CTRL) and compare it to four pointer warping combinations: mouse button with frame relative (MB-FR), mouse button with frame memory (MB-FM), head-orientation–mouse with frame relative (HEAD-FR), and head-orientation–mouse with frame memory (HEAD-FM). This resulted in a total of five different conditions.

The study design was a 5 condition x 2 direction (left-to-right and right-to-left) x 9 paths (specific paths across displays) x 4 trials within subjects design. In our within-subject experiment, each participant performed five blocks of 72 trials for a total of 360 trials per participant. Each block tested one condition (CTRL, MB-FR, MB-FM, HEAD-FR, or HEAD-FM) and the order of presentation of blocks was counterbalanced across participants. Each block consisted of four identical trials for each combination of nine paths and two directions (R and L). All trials within a block were randomized to reduce ordering and learning effects. In summary, the experiment consisted of:

5 blocks (one per condition) ×
9 paths ×
2 directions ×
4 identical trials
= 360 trials per participant

5.3 Procedure

After greeting the participant, the experimenter gave a brief tutorial demonstrating each of the 5 conditions. Before completing each block of trials, the participant was familiarized with the current test condition and given a short practice session (32 trials) which was similar to the actual experiment. The participant completed the entire session with the experimenter watching. Total running time per session was approximately one hour. At the end of the experiment, the participant completed a satisfaction questionnaire.

5.4 Task

The task was based on a Fitts' Law target acquisition task [11], but without the variation of start and target sizes in device space (fixed at 25×25 pixels). To eliminate the overhead of the visual search time, we presented the participant with both start and target buttons simultaneously, asked them to locate both before commencing a trial, and recorded the elapsed time between clicking on the start and target buttons. While this reduced the visual search time, it also allowed the user to plan their action before the trial. However, we specifically instructed the participants not to pre-position their pointer on each screen before each trial to avoid skewing the results.

We selected three target locations on each screen that were aligned, but separated by 100 vertical pixels. Connecting the corresponding targets resulted in nine conceptual paths (Figure 4), none of which are straight paths in visual space. In device space, paths 1, 2, and 3 are the symmetric equivalents of paths 7, 8, and 9, but in visual space these paths cross different size and resolution boundaries. Furthermore, paths 1, 2, 7, and 8 are not straight

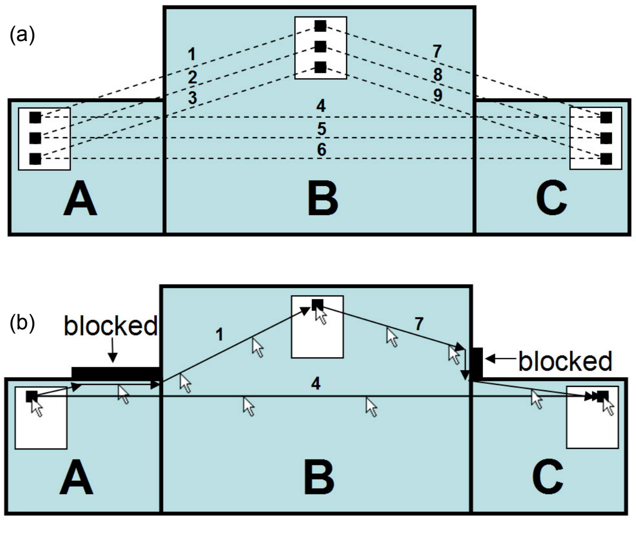

Figure 4: (a) Our experimental task setup consists of nine paths, shown here as dashed lines, each connecting two of nine targets, distributed over three screens. Notice that portions of paths 1, 2, 7, and 8 are blocked by the edges of the screens. (b) The actual paths in device space that a pointer could follow in CTRL condition on path 1 (from A to B), 4 (from A to C), and 7 (from B to C).

paths in device space, since the edges of the screen block standard cursor movement in off-screen space and instead require the participant to move along screen edges, as noted by Baudisch and colleagues [2]. All paths were evaluated in both left-to-right and right-to-left directions.

To increase the visual grouping of the targets, the display background showed an inactive Notepad window (of identical pixel size) on each screen at the target location. This was intended to reinforce the idea pointed out by Grudin [9] that users tend to switch among tasks (windows) when switching displays.

5.5 Hypotheses

Prior to our experiment, we postulated the following four hypotheses:

H1: Pointer warping conditions should outperform CTRL, due to the overall reduction of necessary mouse movement.

H2: Pointer warping conditions using the FM strategy should be the fastest for this task, since they will require the least amount of mouse movement.

H3: Pointer warping conditions should not be as affected by the distance or visual-device space mismatch between screens as CTRL.

H4: Paths 1, 2, 7 and 8 should require longer targeting times than paths 3 and 9 in the CTRL condition, due to screen edges blocking the direct path between targets; however, this should not be the case for pointer warping.

5.6 Results

Movement times were first cleared by removing outliers (movement times more than two standard deviations further from the mean for each condition), which accounted for less than 1% of all trials. We performed a 5 (Condition) × 9 (Path) × 2 (Direction) repeated measures ANOVA on median movement time, with our participants as a random variable. As expected, there were significant effects for the Condition factor, $F_{(4,36)}=11.46$, $p<0.001$ (Figure 5). Additionally, the paired samples t-tests comparing CTRL and HEAD-FM ($t_{(17)}=4.758$, $p<0.001$) and CTRL and MB-FM ($t_{(17)}=5.101$, $p<0.001$) showed that FM conditions significantly outperformed CTRL. Since both FR conditions were not found to be statistically different from CTRL in our experiment, we conclude that our H2 hypothesis was confirmed and that FM was overall the fastest strategy. Our participants demonstrated an overall performance gain of 19% for pointer warping with the FM strategy compared to CTRL.

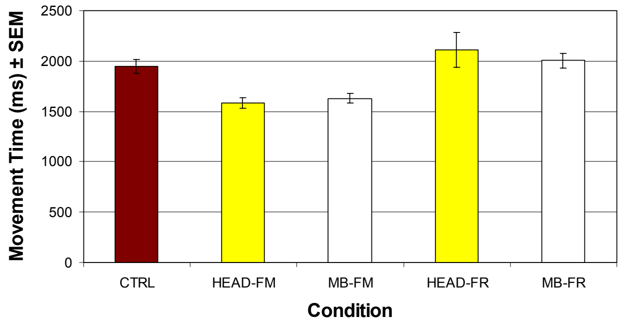

Figure 5: Aggregated movement mean times (ms) for the Condition factor.

H1 was not completely confirmed, since the performance of both FR conditions was not statistically different than CTRL. Although the graph in Figure 5 shows CTRL slightly outperforming HEAD-FR and MB-FR, our analysis shows no statistically significant difference in performance among these three conditions (when compared via t-tests). We believe that there are two reasons why the FR conditions (HEAD-FR and MB-FR) did not outperform CTRL. First, the previous study of pointer warping that explored homogeneous displays [5] confirmed that the FR strategy significantly improves targeting performance only when crossing two or more monitor bezels. However, 2/3 of our current trials required only one bezel crossing. Second, given our particular experimental task design, the FR conditions were always required to traverse a significant distance on the target display after the warp. This, in effect, penalized them in comparison with the FM conditions, which in our test cases always warped the cursor to a location near the target.

We made an interesting observation about the behavior of our participants in CTRL condition. During the experiment, we noticed that they appeared to adopt the following strategy for reaching the target on the next display. First, they appeared to use a fast, deliberate, but often imprecise, hand movement to move the mouse pointer to the target screen. Second, once the pointer was found on the next screen, the participants appeared to perform a precise targeting task. This, in effect, is very similar behavior to pointer warping strategies, where the first action warps the pointer to the target screen, followed by precise targeting afterwards. This

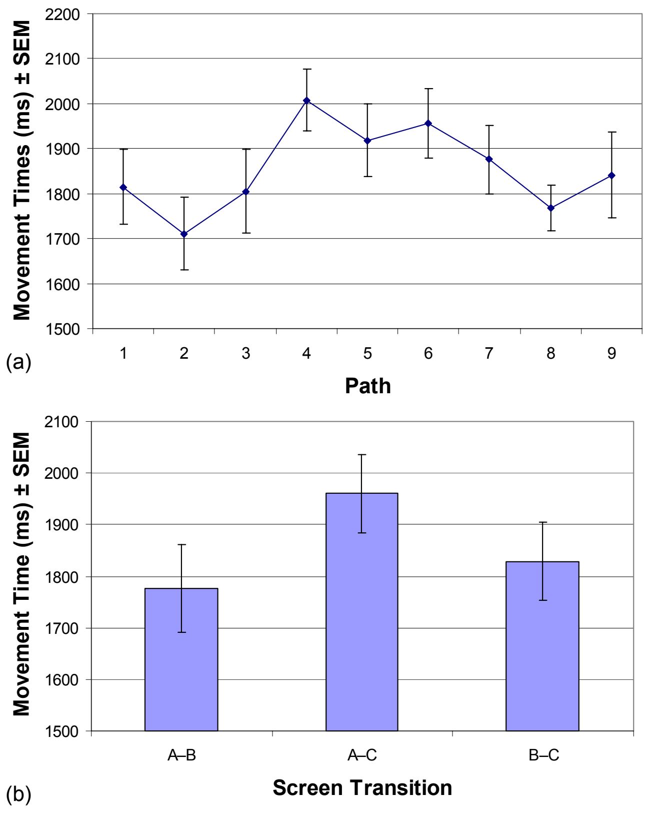

Figure 6: (a) Movement mean times (ms) for the Path factor. (b) Mean screen transition times (ms), computed as aggregated path times (e.g., A–B screen transition is the aggregated times of paths 1, 2, and 3).

observation is consistent with Sears and Shneiderman's analysis of touch screen pointing [14], which separated the targeting task into a gross arm movement, followed by finer finger movements.

Direction contained a significant effect ($F_{(1,9)}$ = 17.447, p<0.005), with overall movement left to right (visually smaller to visually larger screens) found to be about 5% slower than right to left. We speculate that visually larger screens required a longer visual search time to locate the pointer after the pointer transition, resulting in increased overall time when moving from left to right. However, our study gives us relatively limited data to make any definitive conclusions, and we believe that this phenomenon should be further explored in a separate study.

The Path factor, $F_{(8,72)}$=7.625, p<0.001, showed significant effects, with the longest overall paths (4–6) taking the longest time (Figure 6a). The middle paths (2, 5, 8) were the fastest paths overall for their respective screen transitions (A–B, A–C, B–C). This was probably due to the middle target's placement at the mid point between the other two targets, which required the least amount of mouse movement on average. The aggregated screen transition times (Figure 6b) show that two-bezel screen transition (A–C) required on average 10% more time than one-bezel screen transition (A–B or B–C). While this was expected, due to the additional bezel transition, it is interesting to note that the A–B screen transition was slightly faster than B–C, even though both consisted of transitioning identical device space distances and only a single bezel.

An explanation for this discrepancy comes from the analysis of the interaction of Path and Condition (Figure 7), $F_{(32,288)}$=4.351, p<0.001. Performance of CTRL in the A–B screen transition (paths 1–3) was significantly faster than in the A–C or B–C monitor transition. In fact, performance of CTRL seems to be inversely proportional to the degree of mismatch between visual and device space: higher visual–device space mismatch (A–C and B–C) resulted in lower targeting performance, while lower mismatch (A–B) showed improved performance. Interestingly, the mismatch between the visual and the device space (B–C) had a significantly bigger impact on CTRL performance than the mismatch in orientation and visual size alone (A–B).

In contrast, we did not observe this drastic difference between A–B and B–C in any of the pointer warping conditions, which performed almost uniformly across both the high and low mismatch transitions and across one and two bezel transitions (confirming H3). In particular, pointer warping conditions combined with the FM strategy seemed to be the least affected when transitioning visual–device space mismatched screens, requiring on average 1.6s to reach the target in our experiment. When traversing the paths of high visual–device space mismatch, pointer warping (with FM strategy) provided a performance speedup of up to 30% compared to CTRL (e.g., path 8 in Figure 7).

Our last hypothesis (H4) was not confirmed, since we did not observe significant effects of screen edges affecting some cursor paths (1, 2, 7, and 8) more than others (3 and 9) in CTRL condition. We speculate that the difficulties from visual–device space mismatch may have overshadowed the influence of screen edges in this particular task. Overall, visually locating the pointer on the targeting screen seemed to be a fairly challenging task with or without the screen edge interruption. However, a more careful further investigation should be performed to systematically evaluate these influences.

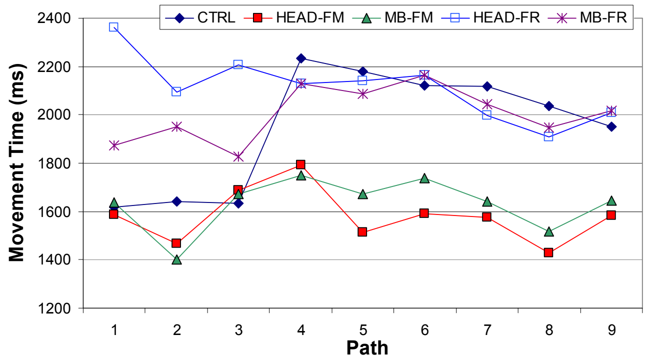

Figure 7: Aggregated movement mean times (ms) for the interaction of Path and Condition.

5.6.1 Subjective Evaluations

The participants filled out a post-experiment questionnaire rating their experience with five techniques on a 10 point Likert scale (1 being most negative and 10 being most positive). They were asked to comment on the ease of use of each condition and performance of each condition, as well as provide their overall preference for condition and strategy. The participants rated HEAD-FM the easiest to use (7.9) and the fastest (8.5), followed by MB-FM (ease of use of 6.8 and performance of 7.8). In contrast, CTRL was rated the hardest to use (5.2) and slowest (4.3) overall.

All participants agreed that they would strongly prefer to use one of the pointer warping techniques over CTRL, with HEAD-FM being the top choice for five participants, MB-FM for three participants) and HEAD-FR for two participants. Overall, seven out of ten participants preferred some form of HEAD switching.

Several participants commented on their frustration with CTRL. One stated that "just using mouse is painful" and another that "it gets very difficult to move between monitors without warping if their resolution varies significantly." Additionally, several participants commented about their lack of familiarity with the side mouse buttons on a five-button mouse, saying that they would definitely improve their performance with extended use of such mice. This is encouraging, because even though the participants were all very familiar with regular mouse use and completely unfamiliar with pointer warping and the side mouse buttons, pointer warping conditions performed at least as fast as CTRL, and outperformed CTRL with the FM strategy. Therefore, we hypothesize that further extended use would further improve the overall pointer warping benefits.

6 Discussion and Conclusions

Our study confirmed that pointer warping offers a significant improvement over standard mouse behavior for heterogeneous multi-monitor environments, even when crossing only a single bezel. Our previous experiment with homogeneous multi-monitor configuration showed that performance improvements from pointer warping were mostly due to gains achieved when crossing two or more bezels [5].

In heterogeneous environments, we found two major advantages to pointer warping compared with standard mouse behavior. First, the benefits grew in proportion to both the distance and the amount of visual–device space mismatch between monitors. Therefore, performance improvements were present even when crossing a single bezel (visible in paths 7–9 in Figure 7). Second, pointer warping conditions performed almost uniformly across both the high and low visual–device space mismatch transitions, providing a targeting speedup of up to 30% over standard mouse behavior.

In addition, the performance of pointer warping was largely dependent on placement strategy, and not on switching implementation. Remembering cursor locations for each screen was not considered too difficult by our study participants, which makes FM a particularly suitable strategy for pointer warping in heterogeneous environments. However, we believe that this conclusion depends largely on the task the user is performing. For our task, where the targets are clustered on each display, FM strategy appears to be preferable. For tasks in which targets are scattered across or span multiple displays, FR might be preferable.

While seven out of ten of our study participants preferred HEAD switching, MB performed similarly. Because of its simplicity, robustness, and reliance solely on standard desktop technology, we would recommend MB as the best technique for multi-monitor arrangements with relatively few displays (up to three). However, for a larger number of displays or display arrangements that do not easily lend themselves to a sequential access (e.g., a 3×3 display grid) we believe that HEAD switching is superior because it allows immediate access to a desired display.

One of the important benefits of pointer warping is that it is a completely optional enhancement. Pointer warping is only invoked if the user wants to warp across screen bezels, and the behavior of the mouse pointer within any particular screen is left completely unchanged. In informal observations of our own extended use, we have noticed that we often employ a hybrid approach: we use pointer warping to traverse a larger distance across screens and we resort to traditional bezel crossing when trying to access nearby targets on the next screen.

Additionally, we hypothesize that it would be possible to combine pointer warping with some of the existing multi-monitor pointer techniques, such as mouse ether [2], to achieve further benefits. Since mouse ether essentially attempts to reduce the display space to that of a homogeneous configuration, in which pointer warping has already proven to be of value [5], one would expect that a combination of mouse ether and pointer warping in a heterogeneous configuration would outperform either alone.

Overall, pointer warping is an easy-to-implement and completely optional enhancement that does not hinder existing mouse behavior in any way. It provides significant improvements in performance in multi-monitor configurations, as confirmed formally in the study reported here, and informally in regular use in our lab. Furthermore, our informal experience shows that extended regular use of pointer warping results in further improvements in performance. We are making the pointer warping widget available for download for use with Windows XP from http://www.cs.columbia.edu/~benko/projects/m3/.

7 Acknowledgments

We would like to thank our study participants and the anonymous reviewers for many helpful comments. This work is funded in part by NSF Grant IIS-01-21239, and a gift from Microsoft Research.

8 References

[1] M. Ashdown, K. Oka, and Y. Sato, "Combining Head Tracking and Mouse Input for a GUI on Multiple Monitors," *Extended Abstracts CHI '05*, ACM Press, 2005, pp. 1188-1191.

[2] P. Baudisch, E. Cutrell, K. Hinckley, and R. Gruen, "Mouse Ether: Accelerating the Acquisition of Targets Across Multi-Monitor Displays," *Extended Abstracts CHI '04* Vienna, Austria, ACM Press, 2004, pp. 1379-1382.

[3] P. Baudisch, E. Cutrell, D. Robbins, M. Czerwinski, P. Tandler, B. Bederson, and Z. Zierlinger, "Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems," *Proc. of INTERACT '03*, 2003, pp. 57-64.

[4] P. Baudisch, E. Cutrell, and G. Robertson, "High-Density Cursor: A Visualization Technique that Helps Users Keep Track of Fast-Moving Mouse Cursors," *Proc. of INTERACT '03*, Zurich, Switzerland, ACM Press, 2003, pp. 236-243.

[5] H. Benko and S. Feiner, "Multi-Monitor Mouse," *Extended Abstracts CHI '05*, Portland, OR, ACM Press, 2005, pp. 1208-1211.

[6] R. Blanch, Y. Guiard, and M. Beaudouin-Lafon, "Semantic Pointing: Improving Target Acquisition with Control-Display Ratio Adaptation," *Proc. of CHI '04*, Vienna, Austria, ACM Press, 2004, pp. 519-526.

[7] M. S. Dulberg, R. S. Amant, and L. S. Zettlemoyer, "An Imprecise Mouse Gesture for the Fast Activation of Controls," *Proc. of INTERACT '99*, IOS Press, 1999, pp. 375-382.

[8] C. Forlines, D. Vogel, and R. Balakrishnan, "HybridPointing: Fluid Switching Between Absolute and Relative Pointing with a Direct Input Device," *Proc. of UIST '06*, ACM Press, 2006, pp. 211-220.

[9] J. Grudin, "Partitioning digital worlds: focal and peripheral awareness in multiple monitor use," *Proc. of CHI '01*, Seattle, Washington, United States, ACM Press, 2001, pp. 458-465.

[10] R. Jacob, "The use of eye movements in human-computer interaction techniques: What you look at is what you get," *ACM Transactions on Information Systems (TOIS)*, vol. 9, 1991, pp. 152–169.

[11] I. S. MacKenzie, "Fitts' Law as a Research and Design Tool in Human-Computer Interaction," *Human-Computer Interaction*, vol. 7, 1992, pp. 91-139.

[12] J. D. Mackinlay and J. Heer, "Wideband Displays: Mitigating Multiple Monitor Seams," *Extended Abstracts CHI '04*, Vienna, Austria, ACM Press, 2004, pp. 1521-1524.

[13] M. A. Nacenta, S. Sallam, B. Champoux, S. Subramanian, and C. Gutwin, "Perspective cursor: Perspective-based interaction for multi-display environments," *Proc. of CHI '06*, Montreal, Canada, ACM Press, 2006, pp. 289-298.

[14] A. Sears and B. Shneiderman, "High Precision Touchscreens: Design Strategies and Comparisons with a Mouse," *International Journal of Man-Machine Studies*, vol. 43, 1991, pp. 593-613.

[15] L. E. Sibert and R. J. K. Jacob, "Evaluation of Eye Gaze Interaction," *Proc. of CHI '00*, The Hague, Amsterdam, ACM Press, 2000, pp. 281-288.

[16] R. Su and B. P. Bailey, "Put Them Where? Towards Guidelines for Positioning Large Displays in Interactive Workspaces," *Proc. of Graphics Interface '05*, Victoria, British Columbia, 2005, pp. 337-349.

[17] D. S. Tan and M. Czerwinski, "Effects of Visual Separation and Physical Discontinuities when Distributing Information across Multiple Displays," *Proc. of Interact '03*, 2003, pp. 252-255.

[18] S. Zhai, C. Morimoto, and S. Ihde, "Manual and Gaze Input Cascaded (MAGIC) Pointing," *Proc. of CHI '99*, Pittsburgh, PA, USA, ACM Press, 1999, pp. 246-253.