Ubiquitous Cursor: A Comparison of Direct and Indirect Pointing Feedback in Multi-Display Environments

Robert Xiao[1], Miguel A. Nacenta[2], Regan L. Mandryk[3], Andy Cockburn[4], and Carl Gutwin[3]

[1]School of Computer Science
University of Waterloo
200 University Avenue West
Waterloo, ON, Canada N2L 3G1
robert.xiao@uwaterloo.ca

[2]Computer Science Department
University of Calgary
2500 University Dr. NW
Calgary, AB, Canada T2N 1N4
miguel.nacenta@ucalgary.ca

[3]Dept. of Computer Science
University of Saskatchewan
110 Science Place
Saskatoon, SK, Canada S7N 5C9
regan.mandryk@usask.ca
carl.gutwin@usask.ca

[4]Dept. of Computer Science
University of Canterbury
Private Bag 4800
Christchurch, New Zealand
8140
andy@cosc.canterbury.ac.nz

Abstract

Multi-display environments (MDEs) connect several displays into a single digital workspace. One of the main problems to be solved in an MDE's design is how to enable movement of objects from one display to another. When the real-world space between displays is modeled as part of the workspace (i.e., Mouse Ether), it becomes difficult for users to keep track of their cursors during a transition between displays. To address this problem, we developed the Ubiquitous Cursor system, which uses a projector and a hemispherical mirror to completely cover the interior of a room with usable low-resolution pixels. Ubiquitous Cursor allows us to provide direct feedback about the location of the cursor between displays. To assess the effectiveness of this direct-feedback approach, we carried out a study that compared Ubiquitous Cursor with two other standard approaches: Halos, which provide indirect feedback about the cursor's location; and Stitching, which warps the cursor between displays, similar to the way that current operating systems address multiple monitors. Our study tested simple cross-display pointing tasks in an MDE; the results showed that Ubiquitous Cursor was significantly faster than both other approaches. Our work shows the feasibility and the value of providing direct feedback for cross-display movement, and adds to our understanding of the principles underlying targeting performance in MDEs.

Keywords: Multi-display environments, mouse ether, perspective cursor, ubiquitous cursor, large displays, targeting.

Index Terms: H.5.2 Information Interfaces and Presentation: User Interfaces - Interaction Styles; Input devices and strategies.

1 Introduction

Multi-display environments (MDEs) are systems in which several display surfaces create a single digital workspace, even though the physical displays themselves are not contiguous. There are many different types of MDE: dual-monitor computers are a simple (and now ubiquitous) example, but more complex environments are also now becoming feasible such as control rooms with multiple monitors in multiple locations, meeting rooms with wall and table displays, or ad-hoc workspaces made from laptops and mobile devices.

One main problem in MDEs is that of moving the cursor from one display to another [16]. This is essentially a targeting task, but one that differs from standard targeting in that the visual feedback is fragmented based on the locations and sizes of the physical displays. In some situations, displays may be far apart, or may be at different angles to one another or to the user. The composition of the MDE and the arrangement of physical displays can have large effects on people's ability to move between visible surfaces.

There are two common ways in which MDE workspaces can be organized: 'warping' and perspective-ether approaches. Warping means transporting the cursor directly from one display to another, without moving through the physical space between monitors. Several techniques for warping have been developed, such as *stitching* (which warps the cursor as it moves across specific edges of different displays) [5], *wormholes* (which warp the cursor when it moves into a specific screen region), *warp buttons* (in which pressing a software or hardware button moves the cursor to each display) [6], or *named displays* (in which the user selects the destination display from a list).

Warping techniques can be fast and effective for cross-display movement. However, they suffer from a number of problems. Warping requires that the user remember an additional mapping (edges, holes, buttons, or names), which might take time to learn; in some techniques (such as stitching), the mappings may become incorrect when the user moves to a new location in the environment. Warp techniques are also distinctly less natural than regular mouse movement: they introduce an extra step into standard targeting actions, and make it more difficult for the user to plan and predict the result of ballistic movements [17]. Finally, the instantaneous jumps of warping techniques cause major tracking and interpretation problems for other people who are trying to follow the action in the MDE.

Perspective-ether techniques for cross-display movement are a different approach that addresses these problems. In this approach, the entire environment is considered to be part of the workspace, including the space between the displays (i.e., 'mouse ether' [2]). The visible parts of the workspace, corresponding to the physical displays, are then arranged based on what the user can see from their current location and perspective. Perspective-ether MDE views provide a workspace in which cursor movement behaves as the user expects, and in which the arrangement of displays corresponds exactly to what the user sees in front of them.

The natural mapping of a perspective-ether view, however, comes at the cost of having to include the 'ether' (i.e., the real-world space between monitors) in the digital workspace. This implies that in order to get from one display surface to another, users must move through a displayless region where there is no direct feedback about the location of a cursor. This is not a major problem with ray-casting solutions (e.g., 'laser pointing'), but does affect indirect pointing devices such as mice or trackpads. One current solution is to use the available display surfaces to provide indirect feedback about the location of the cursor – that is, each display provides feedback (such as an arrow or halo) to indicate the location of the cursor in displayless space (Figure 3).

Although indirect feedback for non-displayed targets can be effective (e.g., [11]), it does require that the user perform (sometimes complex) estimation and inference to determine the cursor's actual location, making cross-display movement more difficult than movement within a display. To address the problems of indirect cursor feedback, we propose a simple solution: provide direct visual feedback about the location of the cursor in 'displayless' space.

We have built a novel display system, called a Low-Resolution Full-Coverage (LRFC) display that can accomplish this solution in any multi-display environment. The LRFC system uses a data projector pointed at a hemispherical mirror to blanket the entire room with addressable (although low resolution) pixels. Using an LRFC display to provide feedback about the cursor in the empty space between monitors results in a technique called *Ubiquitous Cursor* (or *UbiCursor*). The projector only draws the cursor in the space between physical monitors, and uses simple room measurements to ensure that the cursor is shown in the correct location for the user. The result is fast, accurate, and direct feedback about the location of the cursor in 'displayless' space. Our goal is not to turn the entire room into a surface for showing data [27] – only to provide information about objects that are between physical displays.

To test the new technique, we ran a study in which participants carried out cross-display movement tasks with three types of MDE: stitched, perspective-ether with indirect feedback, and perspective-ether with direct feedback (i.e., Ubiquitous Cursor). Our study showed that movement times were significantly lower with UbiCursor than with either stitching or indirect feedback. This work is the first to demonstrate the feasibility of low-resolution full-coverage displays, and shows the value of providing direct cursor feedback in multi-display environments.

2 Related Work

An MDE is a combination of several displays where some kind of interaction can take place across displays. MDEs enable a dramatic increase in the available pixels of an interactive system and have therefore been commonly adapted for commercial desktop systems; they have also been studied by the HCI community for several decades (e.g., [9], [24]).

One of the obvious operations that needs to take place in an MDE is the movement of visual elements from one display to another. Previous research has introduced a number of techniques to achieve cross-display object transfer, including direct touch [23], world-in-miniature (WIM) representations of the display space [7], laser-pointer based interaction [9][15], head-pose and gaze tracking-based interaction [1], and mouse-based techniques [2][6][20] (see [16] for a survey). Although all technique types have advantages and disadvantages, we decided to focus on techniques exclusively based on mouse operation since the mouse is a common, accessible, and inexpensive device with proven performance, and has several advantages over other technique types; for example, it does not cause the same fatigue or inaccuracy seen in ray-pointing techniques [19][20], it allows access from a distance (unlike direct-contact techniques [12][23]), and it does not require a change in visual context, such as WIM techniques [7].

One of the recognized challenges of interacting with MDEs through indirect input devices such as the mouse is *displayless space*, the real-world space between displays that cannot represent any information. In previous work, Nacenta and colleagues [17] showed that in a flat dual-monitor environment, performance diminishes proportionally to the amount of displayless space (following Fitts's Law). This study also showed that when the displayless space is modeled as part of the workspace, performance can be improved with indirect feedback (e.g., Halos [2]); but the best performance was seen when displayless space is ignored (i.e., a warping approach that resembles the standard way that current operating systems connect multiple monitors).

Another relevant issue of mouse interaction with MDEs is that displays can be of very different sizes and be placed at very different angles and distances. It has been shown that the relationship between the positions of the displays and the users can affect pointing performance with a planar mapping of the space [25]. Moreover, Nacenta and colleagues [20] showed that a spherical, perspective-aware input mapping that takes displayless space into account (called Perspective Cursor), can help improve performance in cross-display targeting compared to a simple planar mapping where displayless space is not modeled. In a similar setting comparing related techniques, Waldner and Schmalstieg [26] found that a perspective-aware mapping could be superior to border stitching for certain transitions.

In this paper we address the two challenges presented above, by investigating two different ways of providing feedback and two different input mappings in a complex MDE. The types of feedback we test are *indirect*, where we use a combination of Halos [4] and Wedges [11], and *direct*, where we provide direct low resolution feedback using projectors [8][28]. In terms of mapping, we compare a *perspective-based* approach with *stitching*, which represents an interaction baseline for current systems. The needs of perspective-ether views in MDEs, and the inspirations seen in the wide-view display systems literature, led us to the design of LRFC displays.

MDEs that use LRFC displays are related to mixed-resolution displays such as Focus+Context displays [3], Tiled++ [10], and to interfaces based on steerable projectors (e.g., [22], [27]).

3 The Ubiquitous Cursor LRFC System

Low-Resolution Full-Coverage (LRFC) displays are display systems that blanket an entire multi-display environment with addressable pixels. Large projector-based display systems have been seen before [8][21][22][27][28], but ours is the first to cover an entire room with a single static projector. In the LRFC we developed for the Ubiquitous Cursor, an ordinary data projector is beamed at a hemispherical mirror, which distributes the projector's light around the room (Figure 1, Left). The idea behind LRFC displays is that there are many display tasks in an MDE that are dependent on the physical environment, but that do not need a full-resolution display. Moving between physical displays that are located in different parts of the room is one example.

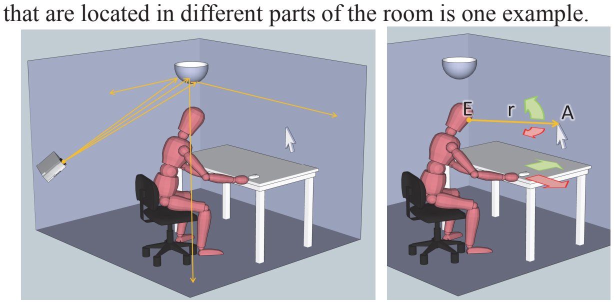

Figure 1. Left: schematic of the LRFC display. By reflecting onto the spherical mirror, the projector can project onto almost any surface. Right: the movements of the mouse cause a change in the orientation of perspective cursor's defining ray.

The algorithm to display the ubiquitous cursor has two phases: the calculation of the location of the cursor in physical space, and the reverse mapping of this position to projector coordinates.

To calculate the location of the cursor in the room we use the Perspective Cursor algorithm (described in [20]). The system calculates the ray (**r** in Figure 1, Right) that goes through the eye-position (**E**) of the user and is oriented according to the movements of the mouse. Moving the mouse left to right will make the ray rotate clockwise around a vertical axis on the user's eye position (changes the azimuth angle – red arrows in Figure 1). Moving the mouse back to fore will rotate the ray to point more vertically (changes the zenith angle – green arrows in Figure 1).

The ray intersects a 3D model of the room that has been previously provided to the system. In our prototype, the 3D model includes all active displays, the tables, the floor and all the walls of the room. Our current system relies on a manually measured model, but we are already planning an automatic calibration based on 3D volume measures from a 3D camera (similar to [28]) that could also track dynamic objects in real time.

The first intersection of the ray with one of the surfaces of the model determines the position of the cursor in physical space (A in Figure 1). If the cursor is located on an active surface, only this display will show it; if the ray intersects a wall or a non-active table, the position of the cursor in the 3D physical space is passed to the next phase of the algorithm to enable projection on a non-active surface.

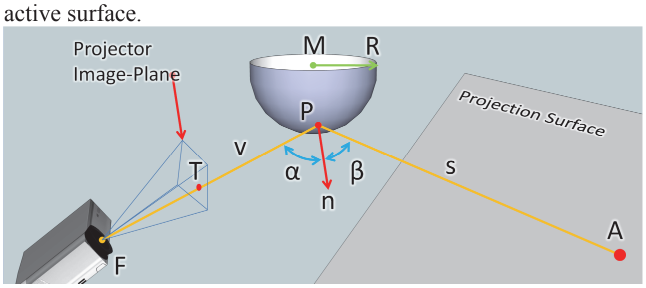

Figure 2. Graphical formulation of the reverse mapping problem.

Now that we know the physical location of the cursor, we need to know how to project onto it. The geometric problem of reverse mapping of the physical position into the image coordinates of the projector is solved by iterative Newtonian approximation. The graphical formulation is illustrated in Figure 2: to project on a given point **A**, we need to find a point **P** on the spherical mirror **M** of radius **R** such that the angles **α** and **β** formed by lines **v** (passing through **P** and the projector's focal point **F**) and **s** (passing through **P** and **A**) are symmetric with respect to the normal **n** to the mirror at **P**. The intersection between the line **v** that connects **F** and the calculated **P** in the image plane of the projector (point **T**) determines the coordinates in the 2D image of the projector that will project onto **A.** These constraints are derived from the physical properties of light propagation and mirror reflection.

The process described above can be applied to multiple points to draw polygonal shapes such as the cursor. Unlike related approaches that use steerable projectors or laser pointers, our system can easily project several cursors. Any modern desktop computer can perform the calculations necessary to provide many UbiCursors in real-time.

The size and brightness of the pixels in the room depend on the projector and size of the room. In our test setup, each pixel is approximately 10x7mm; due to differing distances from the projector, pixels are not exactly the same size all around the room. Because a single projector is used to cover the entire room, the brightness of the image is reduced. In our test setup, which uses an ordinary Sony VPL-CX11 1500-lumen projector in a low-light environment, the Ubiquitous Cursor is easily visible. A more powerful projector would easily be able to display the cursor in either a brighter or a larger room.

The control-display gain for perspective cursor in our system is fully adjustable. For our experiment we set it so that 3000 mouse pixels translate into 180 degrees for either movement; in other words, the entire field of view has the same mouse resolution as a 3000x3000 pixel display.

We conducted our study with participants in a fixed location, so we were able to achieve perspective effects without real-time head tracking. In a real-world implementation, the location of the user's head must be tracked; this is now becoming possible with low-cost equipment [21].

4 Empirical Evaluation

Our goal in the study was to compare the effectiveness of direct feedback (i.e., the UbiCursor) to both an indirect-feedback solution (modified Halos) and a warping technique (Stitching). We also tested a combined technique that used both UbiCursor and Halos. In the study, participants carried out simple cross-display pointing tasks in an MDE with five displays.

4.1 Apparatus

We constructed a multi-display environment in a meeting room, using five displays (a 17" desktop monitor, a 14" laptop, a 12" laptop, a 45" plasma TV, and a 17" SMART table). The room setup is shown in Figure 4. All displays were controlled by separate Windows computers, which ran a custom Python application to create the compound digital workspace. Computers communicated via a standard Ethernet network.

4.2 Interaction Techniques

The primary goal of our comparison was to evaluate the differences between direct and indirect feedback for cross-display movement in MDEs. Additional goals were to compare the Ubiquitous Cursor technique to a standard cursor-warping technique (Stitching) that performed best in a flat multiple-monitor test [17].

4.2.1 UbiCursor

UbiCursor implements direct cursor feedback. The Ubicursor condition was implemented as described above (see Figure 2). The cursor was displayed on all room surfaces, except for a small patch on the ceiling that was in the shadow of the hemispherical mirror. The cursor followed the C:D ratio of the perspective system (i.e., it did not switch to the monitor's C:D when inside the displayable region).

4.2.2 Perspective Cursor + WedgeHalo

Indirect cursor feedback was provided by an adaptation to the Halos technique developed by Baudisch and Rosenholtz [4]. Halos indicate the location of an off-screen object by drawing a virtual circle around the object that just intrudes into a displayable surface (Figure 3). A Halo was displayed on at least one of the screens; the user could determine the location of the off-screen object by imagining the center of the circles on the displays.

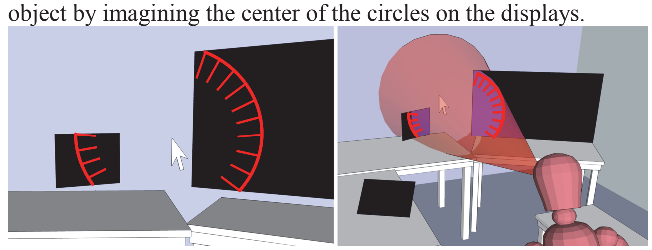

Figure 3. A Perspective WedgeHalo as seen by the user (left) and its projective cone (not visible in real implementation).

Our adaptation to the basic Halo technique was to add small orthogonal lines to the circle that each point directly to the cursor

(Figure 3). These were added because we found in pilot testing that people had difficulty determining the cursor's direction in some parts of the room; the directional lines solved this problem. Except for the directional components of the halos, this condition is the most similar to the Perspective Cursor condition of [20].

4.2.3 UbiCursor + WedgeHalo

This technique used a combination of the two methods described above. No halos were shown in the space between displays (even though this space was addressable using the UbiCursor's display).

4.2.4 Stitching

The stitching configuration (Figure 4, left) was created by the authors before the design of the experiment; we tried to achieve the best possible stitching through a compromise between using the perspective of the user and not leaving the largest display boundaries unused. Some pairs of displays did not have obvious stitching connections, and thus movement between these pairs required crossing one or more intermediate displays. For instance, movement between displays B and E required passing through either display C or A. The only active feedback provided by this condition was the cursor itself. The boundaries of connecting displays were colour-coded to indicate connection. The C-D gain in stitching was based on each display's pixel space, adjusted with a 1.5 factor that allowed crossing the space without clutching.

4.3 Tasks

Participants performed repeated aiming tasks, which always started on one display and ended on another (there were no within-display paths). We tested six paths as shown in Figure 4 (right): A→C, B→C, C→E, E→D, D→B, and A→E. Targets were presented in both directions for all paths (e.g., A→C and C→A). Paths were one of three types: *coaxial* movements across right-to-left and top-to-bottom seams (B→C, C→E), *non-coaxial* movements across right-to-top seams (A→C, E→D), and *multi-hop* movements across intermediate displays (D→B, A→E).

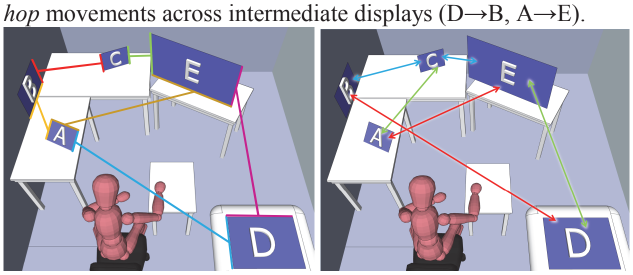

Figure 4. Stitching configuration of the displays (left) and experimental paths (right). Blue: coaxial, Green: non-coaxial, Red: multi-hop.

The aiming task was comprised of an initial selection of the source target, movement to the display containing the destination target, and selection of the destination target. The destination target of a trial and the source target of a subsequent trial were never presented on the same display, requiring participants to move the cursor to a different display between trials. We arranged the trials in this way to avoid having performance of a trial be affected by the previous trial, and to help participants visually acquire both the source and the destination targets before beginning their aiming movement; pilot tests showed that participants found it difficult to remember to visually acquire both targets if two trials were chained together by beginning a trial on the display where the previous trial had ended. The source target was always presented in the center of the monitor, and the destination target was presented either in the center or at the leading edge of the display (see Figure 5). We included both center and leading edge destination targets because we were interested in how the proximity of the target to a display edge would affect the aiming time and accuracy. Source targets were a light blue circle, destination targets were a yellow circle, and the backgrounds of all six displays were black.

All targets were circles and were the same size – 100 pixels in diameter. Because the different displays had different resolutions, the targets appeared visually different to participants, but were the same size in motor space. Keeping target size consistent in motor space between trials is more important for making all target selections comparable than maintaining the visual size of targets.

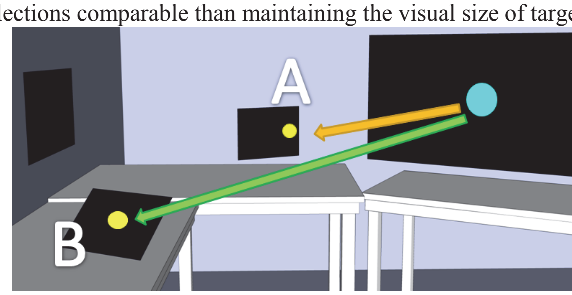

Figure 5. Leading edge target task (A) and Center target task (B).

4.4 Participants and Procedure

There were 16 participants (8 female, 8 male), aged between 20 and 38 (mean 27) in our study. All were right-handed mouse users. Participants provided informed consent and were given a $10 honorarium. The study took approximately one hour. The data of one participant had to be deleted due to a software error.

Participants began the experiment by completing a demographics questionnaire. After performing repeated trials of the tasks for each technique (described in Section 4.5), a post-condition questionnaire asked the participants to rate that technique using a modified NASA TLX questionnaire. After completing all trials using all techniques, participants completed a final post-study questionnaire.

4.5 Experimental Design and Measures

The study used a repeated-measures factorial design with three factors:

- Technique (UbiCursor, WedgeHalo, UbiCursor+Halo, Stitching)
- Path (six unique paths in both directions; see Figure 4)
- Target location (leading-edge or center)

For each combination of the three factors, there were six repetitions of a trial presented in three blocks. The first block was treated as training to account for learning effects.

All trials for a technique were completed prior to using another technique. A Latin square was used to counterbalance the order of interaction technique, but the different paths were randomly presented to the participant. If a participant made a selection error on the destination target, we required them to continue trying until the target was correctly selected. That trial was marked as an error and was added back to the queue of trials to complete in that block. As a result, we have a fully-balanced set of error-free trials for the crossing of our three experiment factors. We retained the error trials to report the accuracy as well as the speed of the different techniques. With four interface conditions, twelve paths (six in each direction), two target locations, and six trials per cell, we collected 576 successful trial data points per session and participant, of which 192 were training.

The dependent variables, recorded by the study system, were trial completion time and number of errors. We also report on the results of the NASA Task Load Index worksheets completed after each interface condition, and the post-experiment questionnaire.

4.6 Hypotheses

H1. Direct feedback (UbiCursor, UbiCursor+WedgeHalo) will perform better than indirect feedback.

H2. Perspective techniques and Stitching will perform differently according to the task and task type.

H2a. Perspective techniques will perform better than Stitching in multi-hop and non-coaxial tasks.

H2b. Stitching will perform as well as or better than perspective techniques in coaxial tasks.

H3. Tasks with centered targets will take longer to complete than tasks with leading edge targets.

H4. Learning with Stitching will be slower than with perspective techniques.

5 Results

We report the results of the study in three groups: planned objective measures, additional analyses of the objective data, and subjective measures.

5.1 Objective Measures

We present the analysis of the objective measures (completion time and errors) in four sections: global time analysis, path group analysis, learning analysis, and error analysis. Unless otherwise stated, the measures used exclude error trials.

5.1.1 Global Time Analysis

An omnibus ANOVA with three factors: *technique* (UbiCursor, WedgeHalo, UbiCursor+WedgeHalo, Stitching), *path* (12 levels, 6 different display combinations in both directions), and *target type* (centered on display, or in leading edge), and participant as random factor yielded significant differences on the log-transformed completion times for *technique* ($F_{3,42}$=8.7, p<.001, η^2=.39), path ($F_{11,154}$=166, p<.001, η^2=.92), *target type* ($F_{1,14}$=285, p<.001, η^2=.95), and for all fixed factor interactions except *technique*target type* ($F_{3,42}$=.52, p=.67, η^2=.04). Logarithmic transformation of the data was required to comply with the normality assumption of the parametric ANOVA.

Post-hoc tests on the *technique* factor reveal that all average completion times between techniques were statistically different (all p<0.006 after Tukey HSD multi-comparison correction) Averaged across all tasks and target types, UbiCursor is the fastest (μ=1.83s), followed by UbiCursor + WedgeHalo (μ=1.92s), WedgeHalo (μ=1.98s), and Stitching (μ=2.04s). See Figure 6.

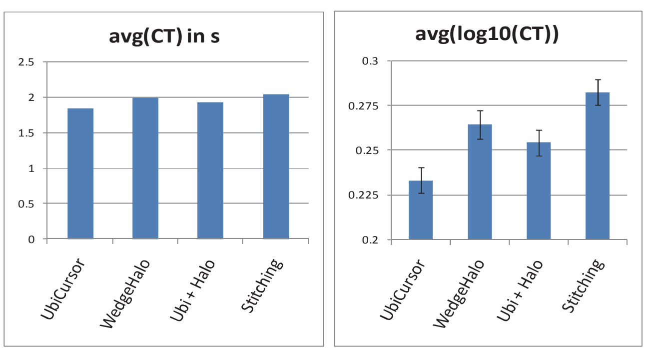

Figure 6. Average completion times by technique in a linear scale (left) and after a log10 transformation (right). Error bars indicate standard error. Note that the vertical scale starts at 0.2 for the log-transformed graph.

The results from the omnibus comparison of techniques generally support H1 (direct feedback is better than indirect feedback); the interaction between technique and path supports H2 (techniques perform differently on different paths). H3 is contradicted by the results since center targets were reached significantly faster than leading edge targets, even though the distance that needs to be covered is larger (μ_{center}=1.8s, μ_{edge}=2.1s).

5.1.2 Path Group Analysis

To test H2, H2a, and H2b, we performed ANOVAs equivalent to the global test, but separately for each of the three a-priori groups of tasks (coaxial, non-coaxial, and multi-hop). The results are analogous to the omnibus test results (*technique*, *path* and *target type* p<0.05), except that the *technique*target type* interaction was significant for the coaxial tasks (unlike the omnibus and the other task groups). The post-hoc comparisons between techniques yield the same ordering (UbiCursor, UbiCursor+WedgeHalo, WedgeHalo, Stitching), but with fewer statistically significant pairings because of the reduced power of the segmented data analysis (see Table 1).

Table 1. P values of the post-hoc multiple comparison tests for the different path groups (Tukey HSD multiple comparisons, significant if < 0.05). Green cells indicate significant, red not-significant, yellow close to significance.

	coaxial				non-coaxial				Multi-hop			
	Ubicursor	Ubi+Halo	WedgeHalo	Stitching	Ubicursor	Ubi+Halo	WedgeHalo	Stitching	Ubicursor	Ubi+Halo	WedgeHalo	Stitching
UbiCursor	x	0.56	<0.01	<0.01	x	<0.01	<0.01	<0.01	x	<0.01	<0.01	<0.01
Ubi+Halo	d	x	0.11	0.52	d	x	0.22	<0.01	d	x	0.4	<0.01
WedgeHalo	d	d	x	0.98	d	d	x	<0.01	d	d	x	<0.03
Stitching	d	d	d	x	d	d	d	x	d	d	d	x

These results generally confirm H2a and contradict H2b (i.e., Stitching did not perform better in any path group) but, more importantly, provide evidence that the grouping of paths we determined a priori is not useful to further differentiate the performance of the different techniques. We address this issue in Section 5.2 (additional analyses).

5.1.3 Analysis of Learning

To test H4 (differences in learning for the different techniques), we performed a three-way ANOVA that included block as the main factor. The block factor had three levels, where the first block was the training block, and the second and third correspond to actual experimental trials done in that order. Each block had two repetitions of each path/target type/technique combination (two repetitions of each cell). The results of the ANOVA show a significant main effect of block ($F_{2,28}$=35.5, p<0.001, η^2=.71), but the interaction between block and technique was not significant ($F_{6,84}$=.58, p=.74, η^2=.04), which indicates that differences between the learning patterns of techniques were not large enough to be detected by our experiment, and therefore H4 is unsupported by the analysis.

5.1.4 Error Analysis

The error counts across participants (see Figure 7) reveal that participants missed the target many more times with Stitching (502 misses, 33 per participant average) than with any of the other techniques (UbiCursor: 354 misses, 23.6 per participant, WedgeHalo: 364 misses, 24.3 per participant, Ubi+Halo: 391 misses, 26 per participant). Notwithstanding the size of the overall differences, a non-parametric Friedman test revealed no significant difference in the number of errors between techniques ($\chi^2(3)$=.568, p = .904), possibly due to the large variability in number of errors between participants. Some participants made large numbers of errors with Stitching – up to 86 – whereas for two participants Stitching was the only technique with no errors.

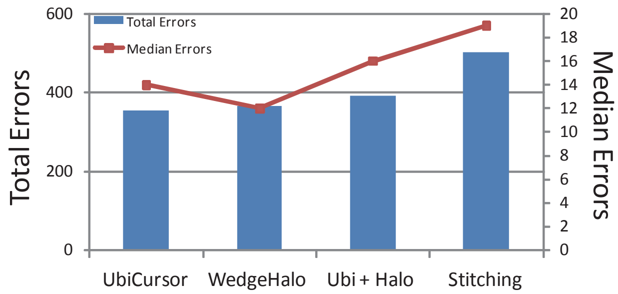

Figure 7. Total and median errors (excluding training).

5.2 Additional Analyses

As discussed above, the a-priori classification of tasks into coaxial, non-coaxial, and multi-hop yielded equivalent performance relationships between techniques. This means that our classification does not provide extra information about what kind of tasks each technique is suited for. To better understand what makes a technique good for a certain task, we performed an *a-posteriori* classification of paths according to their performance profile, i.e., according to the pattern of performance of the different techniques with the different paths. Note that due to its *a posteriori* nature, the analysis in this section should be considered exploratory, should be interpreted with caution, and should be subject to replication in future *a priori* analyses.

We observed three main groups of paths according to technique performance: paths where UbiCursor was fastest and Stitching substantially slower, with the other techniques somewhere in the middle (UbiCursor-favoured paths); paths where all techniques had comparable performance (Neutral); and paths where Stitching was faster than all other techniques (Stitching-favoured). To allow the reader to make their own judgment about this classification, the three groups of performance profiles are plotted in Figure 8.

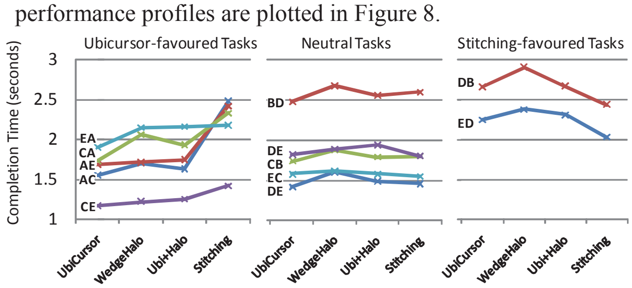

Figure 8. Performance profiles of each path (in seconds), split left to right into three kinds of tasks. Each line represents a task, labeled on its left with origin and destination display. To highlight the profiles, the scales start at 1 s.

The next step is then to speculate what the paths in the different groups have in common to explain what types of paths are better served by which techniques. Figure 9 provides a simple visualization of how the different path groups are distributed in space, with green lines indicating Ubicursor-favoured paths, red lines indicating Stitching-favoured paths and neutral paths in blue. The paths where Stitching showed a clear advantage are those where there was a large displayless distance between the displays, and where the target display was smaller than the origin display. In general, UbiCursor and the other perspective techniques showed substantial advantages for tasks between displays that were closely clustered. The clear advantage of perspective techniques for paths connecting A and C is partially attributable to A and C not being directly stitched together (through piloting we determined that providing stitching mappings from the same side to several displays made it more difficult and increased errors and variability). However, a similar advantage was present for paths between displays A and E, which were stitched directly in the stitching map but still took longer with Stitching.

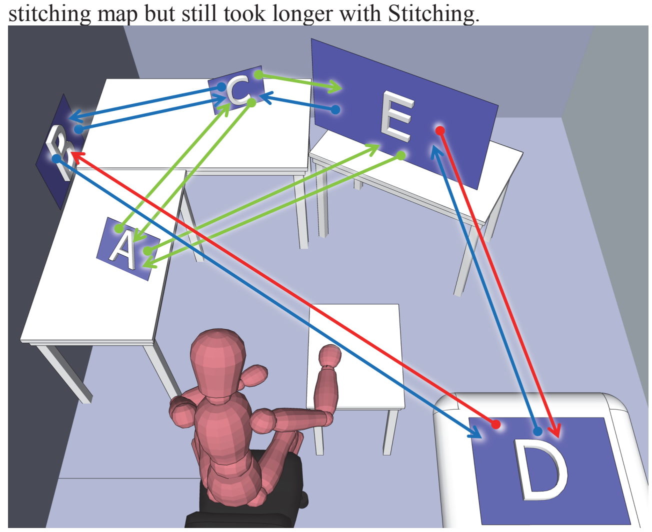

Figure 9. Performance by group: UbiCursor-favoured (green), Neutral (blue), Stitching-favoured (red).

5.3 Subjective Measures

We analyzed the responses to the post-condition TLX questionnaires through one-way non-parametric Friedman ANOVAs of the 7-point Likert-scale answers. Of all the questions, only the question about mental demand (How mentally demanding was the task?) yielded a statistically significant difference ($\chi^2(3)=10.97$, $p = .012$). For this question, participants ranked UbiCursor+WedgeHalo as the least mentally demanding technique ($\mu=2.73$), followed by Ubicursor ($\mu=3.13$), WedgeHalo ($\mu=3.38$), and Stitching as the most mentally demanding technique ($\mu_{rank}=4.13$).

Post-study user rankings of the four techniques in terms of speed and accuracy were not significantly different (all $\chi^2(3)<5$, $p>0.17$), although a careful look at the distribution of the rankings shows a certain preference for perspective techniques, and bipolar answers for Stitching (i.e., participants either thought that stitching made them fastest or slowest – See Table 2 and Table 3).

Table 2. Number of participants that ranked the techniques best to worst in terms of speed.

Speed Rank	1 (best)	2	3	4 (worst)	AVG
UbiCursor	4	5	5	1	2.2
UbiHalo	5	3	6	1	2.2
Halo	1	4	4	6	3
Stitching	5	3	0	7	2.6

Table 3. Number of participants that ranked the techniques best to worst in terms of accuracy.

Accuracy Rank	1 (best)	2	3	4 (worst)	AVG
UbiCursor	6	3	5	1	2.07
UbiHalo	4	5	5	1	2.2
Halo	1	4	5	5	2.93
Stitching	4	3	0	8	2.8

6 DISCUSSION

Our study showed that a direct-feedback, perspective-based technique for supporting cross-display movement (Ubiquitous Cursor) was significantly faster than an indirect-feedback technique (WedgeHalo), a combination technique (Ubi+Halo), or a standard cursor-warping technique (Stitching). In the following sections we explain these results in terms of the main differences between these techniques (direct vs. indirect feedback; perspective vs. warping), and also discuss the limitations of this work and the ways it can be generalized for designers of MDEs.

6.1 Direct vs. Indirect MDE feedback

The main goal of our experiment was to investigate the differences between direct and indirect feedback for mouse-based cross-display targeting. The results of our experiment provide solid evidence for our hypothesis that UbiCursor (a technique with direct targeting feedback) is better than indirect forms of feedback, such as wedges or halos. The difference between our direct and indirect conditions is underscored by the fact that the indirect feedback technique we tested – WedgeHalo – was optimized for the study in ways that would cause difficulties in real use (e.g., it occludes many pixels on the displays and would be distracting in collaborative work environments).

In addition, the combination of direct and indirect feedback (UbiCursor+WedgeHalo) was not equivalent to UbiCursor alone. Adding indirect feedback appeared to impair performance, possibly due to the extra cognitive load of deciding which type of feedback to pay attention to. This result is relevant for the design of targeting techniques in MDEs because it indicates that, for targeting tasks, more information is not necessarily better.

6.2 Perspective vs. Stitching

The empirical study presented in this paper provides further evidence that using an input mapping that corresponds to the user's position (i.e., perspective techniques) is beneficial for performance. Our results tested an MDE where the displays were sparser than in the original Perspective Cursor study [20]. Moreover, our results also help generalize the original findings to variants of perspective where feedback is direct, and to other forms of indirect feedback.

Although we expected our initial classification of paths to shed some light on the differences between techniques, it was only through a new regrouping that we could further learn about the specific strengths of each technique. Our results suggest that Stitching only has an advantage over perspective techniques if the displayless gap is large. In contrast to the planar dual-monitor setup studied by Nacenta et al. [17], where Stitching was the fastest technique even with relatively small gaps between displays, the more complicated transitions between displays in our experiment made perspective mappings a better option, even for short transitions such as the C→E path (see Figure 9).

Our additional analysis also suggests that perspective provides an advantage over Stitching for traveling from a small display to a larger display (e.g., C→E), but this advantage is reversed when targeting in the opposite direction (e.g., E→C) because of the 'funneling' effect created by stitching a large screen edge to a smaller one. In perspective techniques, traveling from large to small screens requires reaching the small display within its surrounding displayless space. We believe that this effect may be responsible for the asymmetry in results for paths B→D and D→B, and paths E→D and D→E.

6.3 Input Geometry in MDEs

Testing two kinds of target positions within the target display revealed that reaching targets that are close to the leading edge is harder than targets that are centered. This is not surprising for Stitching techniques, which are known to cause overshooting [17], but was unexpected with the perspective-based techniques (including UbiCursor, which provides direct feedback). This results contradicts linear and angular [14] formulations of Fitts's law (i.e., by definition, leading-edge targets are closer to the starting point and should therefore be faster to reach). We speculate that the visual transition from background display to foreground display may have caused people difficulty; however, this is a phenomenon that should be investigated in future work.

Combining our findings about displayless space with Nacenta et al.'s earlier results [17] implies that the targeting geometry of complex MDEs is very different from that of small and large single displays. Designers of multi-display environment interfaces can take this into account: for example, commonly accessed interface elements could be placed at locations that are unlikely to be leading edges (e.g., top center of display E in our configuration), and displays that are frequently used in combination can be located so that they have only a small gap.

6.4 Limitations of the Study

We designed our study to test a broad range of targeting transitions that represent a sample of many of the types of targeting tasks that could take place in complex MDEs. For example, the paths that we selected represent transitions from horizontal to vertical displays, from large displays to small, and between displays that are close or distant from each other. This provides a fairly generalizable set of tasks, but makes it difficult to quantify the specific contributions of factors to overall performance. It is therefore necessary to follow up with experiments that are designed to investigate the factors that our study highlighted as most relevant: the effect of angle difference between displays, the threshold at which displayless space becomes detrimental for performance, and the effect of display size differences on targeting. To further generalize the results, it would also be interesting to test tasks with different target sizes.

Finally, the focus of our study was on targeting feedback for mouse-based interaction. Although we believe that mouse interaction will still be predominant for future MDEs, new MDE control techniques from other emerging input paradigms such as multi-touch interaction and free-air gesturing should be designed and tested against perspective mouse interaction.

6.5 Recommendations for Practitioners

Although additional work needs to be done to replicate and extend our results, there are several principles and guidelines that can be generalized from our experiences. These ideas will help designers of MDEs understand the issues underlying cross-display targeting performance.

Stitching becomes problematic in complex MDEs. Although Stitching is a simple solution for composing an MDE's workspace, and although Stitching outperforms Ether-based approaches in simple setups [17], this technique becomes more difficult for users when paths do not map easily to a 2D plane. For highly complex MDEs, perspective-based approaches should be considered as a way to simplify cross-display movement.

Use direct feedback for cross-display feedback. In situations where perspective-based techniques are used, our study shows conclusively that direct feedback improves performance. The low-resolution full-coverage display system that we developed shows that direct feedback can be provided simply and inexpensively.

Stitching will still be faster if the real-world distance between displays is large. As distances between displays increases, eventually the advantage of cursor warping overshadows any benefits of perspective-based techniques. If an MDE's displays are very far apart, Stitching will likely be the best choice (although hybrid techniques are also possible).

Combined direct/indirect feedback is likely not valuable. Our study showed that combining both feedback types did not improve performance; although participants rated this technique as lowest in mental effort, it may be that the added indirect feedback actually complicates the process of tracking the cursor.

Leading edges of displays are more difficult to target. Participants had difficulty acquiring leading-edge targets with all of the techniques that we tested. Designers can avoid problems with target location by placing commonly-accessed objects on locations that are unlikely to be leading edges (given the probable between-display transitions for the MDE).

7 Conclusions

Multi-display environments present the problem of how to support movement of objects from one display to another. We developed the Ubiquitous Cursor system as a way to provide direct between-display feedback for perspective-based targeting. In a study that compared Ubiquitous Cursor with indirect-feedback Halos and cursor-warping Stitching, we showed that Ubiquitous Cursor was significantly faster than both other approaches. Our work shows the feasibility and the value of providing direct feedback for cross-display movement, and adds to our understanding of the principles underlying targeting performance in MDEs.

Our initial experiences with Ubiquitous Cursor suggest several directions for further research. First, we plan to test the UbiCursor technique with more realistic MDE tasks; in particular, we will explore the effects of having different C:D ratios in the projected display and the MDE displays. Second, we will further investigate the principles uncovered in our study (effects of angle differences between displays, performance thresholds for the different techniques, the effects of different display and target sizes, and the use of the technique with other input devices). Third, we will explore the other possibilities presented by the idea of a low-resolution full-coverage display, which can enable augmentation of and interaction with real-world objects inside the scope of the projected display.

8 Acknowledgements

This work has been funded through the contributions of Canada's NSERC, Alberta's iCore, the SurfNet strategic network, and SMART Technologies. The authors would also like to thank the participants of the study, Brett Watson, and the members of the interaction lab for their feedback.

References

[1] Ashdown, M., Oka, K., and Sato, Y., Combining head tracking and mouse input for a GUI on multiple monitors. *Proc. CHI 2005*, 1188-1191.

[2] Baudisch, P., Cutrell, E., Hinckley, K. and Gruen, R., Mouse Ether: Accelerating the Acquisition of Targets Across Multi-Monitor Displays. *Proc. CHI 2004*, 1379-1382.

[3] Baudisch, P., Good, N., and Stewart, P. Focus plus context screens: combining display technology with visualization techniques. *Proc. UIST '01*, 31-40.

[4] Baudisch, P., and Rosenholtz, R., Halo: A Technique for Visualizing Off-Screen Locations. *Proc. CHI 2003*, 48-488.

[5] Benko, H. & Feiner, S., Pointer Warping in Heterogeneous Multi-Monitor Environments. *Proc. Graphics Interface,* 2007, 111-117.

[6] Benko, H. & Feiner, S., Multi-Monitor Mouse. *Proc. CHI 2005*, 1208-1211.

[7] Biehl, J. and Bailey, B. ARIS: an interface for application relocation in an interactive space. *Proc. Graphics Interface 2004*, 107-116.

[8] Birnholtz, J., Reynolds, L., Luxenberg, E., Gutwin, C., and Mustafa, M., Awareness Beyond The Desktop: Exploring Attention And Distraction With A Projected Peripheral-vision Display, *Proc. Graphics Interface 2010*, 55-62.

[9] Bolt, Richard A., "Put-that-there": Voice and gesture at the graphics interface. *Proc. SIGGraph 1980*, 262-270.

[10] Ebert, A., Thelen, S., Olech, P.-S., Meyer, J., Hagen, H. Tiled++: An Enhanced Tiled Hi-Res Display Wall. IEEE TVCG Jan/Feb 2010, 120-132.

[11] Gustafson, S., Baudisch, P., Gutwin, C, and Irani, P., Wedge: Clutter-Free Visualization of Off-Screen Locations, *Proc. CHI 2008*, 787-796.

[12] Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., and Smith, M., Stitching: pen gestures that span multiple displays. *Proc. AVI 2004*, 23-31.

[13] Johnson, T., and Fuchs, H. A Unified Multi-Surface, Multi-Resolution Workspace with Camera-Based Scanning and Projector-Based Illumination, *Proc. IPT-EGVE Symposium*, 2007.

[14] Jota, R., Nacenta, M., Jorge, J., Carpendale, S., and Greenberg, S., A comparison of ray pointing techniques for very large displays. Proc. Graphics Interface 2010, 269-276.

[15] Myers, B., Bhatnagar, R., Nichols, J., Peck, C., Kong, D., Miller, R., and Long, A., Interacting at a distance: measuring the performance of laser pointers and other devices. *Proc. CHI 2002*, 33-40.

[16] Nacenta, M., Gutwin, C., Aliakseyeu, D., and Subramanian, S., There and Back again: Cross-Display Object Movement in Multi-Display Environments, *JHCI*, 24, 1, 2009, 170-229.

[17] Nacenta, M., Mandryk, R., and Gutwin, C, Targeting across displayless space. *Proc. CHI 2008*, 777-786.

[18] Nacenta, M., Sakurai, S., Yamaguchi, T., Miki, Y., Itoh, Y., Kitamura, Y., Subramanian, S. and Gutwin, C., E-conic: a Perspective-Aware Interface for Multi-Display Environments. *Proc. UIST 2007*, 279-288.

[19] Nacenta, M., Pinelle, D., Stuckel, D., and Gutwin, C., The Effects of Interaction Technique on Coordination in Tabletop Groupware. *Proc. Graphics Interface*, 2007, 191-198.

[20] Nacenta, M., Sallam, S., Champoux, B., Subramanian, S., and Gutwin, C., Perspective cursor: perspective-based interaction for multi-display environments. *Proc. CHI 2006*, 289-298.

[21] Nacenta, M., *Computer Vision approaches to solve the screen pose acquisition problem for Perspective Cursor*, Report HCI-TR-06-01, Dept. of Computer Science, University of Saskatchewan, 2006.

[22] Pinhanez, C., Kjeldsen, R., Levas, A., Pingali, G., Podlaseck, M., and Sukaviriya, N., Applications of Steerable Projector-Camera Systems, *Proc. ICCV Workshop on Projector-Camera Systems (PROCAMS'03)*, 2003.

[23] Rekimoto, J. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. *Proc. UIST 1997*, 31-39.

[24] Robertson, S., Wharton, C., Ashworth, C. & Franzke, M., Dual Device User Interface Design: PDAs and Interactive Television. *Proc. CHI 1996*, 79-86.

[25] Su, R., and Bailey, B., Put Them Where? Towards Guidelines for Positioning Large Displays in Interactive Workspaces, *Proc. Interact 2005*, 337-349.

[26] Waldner, M., and Schmalstieg, D., Experiences with Mouse Control in Multi-Display Environments, *Proc. Workshop on coupled display visual interfaces at AVI 2010*, 6-10.

[27] Welch, G., Fuchs, H., Raskar, R., Towles, H., and Brown, M., Projected Imagery in Your Office in the Future. *IEEE Computer Graphics and Applications*, Jul-Aug 2000, 20, 4, 62-67.

[28] Wilson, A., and Benko, H., Combining multiple depth cameras and projectors for interactions on, above and between surfaces. *Proc. UIST 2010*, 273-282.