Gesture Select: Acquiring Remote Targets on Large Displays without Pointing

Andrew Bragdon and Hsu-Sheng Ko
Brown University
Providence, RI, USA
{acb, hsusheng}@cs.brown.edu

ABSTRACT

When working at a large wall display, even if partially utilized, many targets are likely to be distant from the user, requiring walking, which is slow, and interrupts workflow. We propose a novel technique for selecting remote targets called Gesture Select, in which users draw an *initial mark*, in a target's direction; rectilinear gestures represented as icons are dynamically overlaid on targets within a region of interest; the user then continues by drawing the *continuation mark* corresponding to the target, to select it. Extensions to this technique to support working with remote content for an extended period, and learning gesture shortcuts are presented. A formal experiment indicates Gesture Select significantly outperformed direct selection for mid/far targets. Further analysis suggests Gesture Select performance is principally affected by the extent to which users can read the gestures, influenced by distance and perspective warping, and the gesture complexity in the ROI. The results of a second 2-D experiment with labeled targets indicate Gesture Select significantly outperformed direct selection and an existing technique.

Author Keywords
Gestures, selection, large display, remote targets

ACM Classification Keywords
H5. Information interfaces and presentation: Interaction.

General Terms
Human Factors

INTRODUCTION

Large, high-resolution display walls are growing in popularity and are now available as commercial products [22] [29]. A long history of research applications in this area demonstrates the value of these systems in various domains, such as interactive whiteboards [25] and scientific visualization [24].

While these systems can be operated from a distance (e.g. [33] [21]), we believe there is significant value in using them up close. When standing in front of the display, users can write with a pen to annotate or manipulate objects with their hands directly on the display. They can also collaborate with other users gathered around the display, similar to a very large whiteboard. Finally, some wall displays tile multiple projectors to offer very high resolutions, letting users perceive fine detail even when standing at the display (e.g. [16]).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.
Copyright 2011 ACM 978-1-4503-0267-8/11/05....$10.00.

However, when working at a large display, selecting remote targets requires the user to walk to select targets that are out of reach, something that is likely to occur frequently on a 15-foot wide display that is even partially utilized, even if the user is standing in the center. Such interruptions may hinder user workflow on a large display.

In this paper we propose a novel technique, *Gesture Select*, which aids users in selecting remote targets on large displays (Fig. 1). The technique differs from prior approaches in that it does not use pointing-based selection. Rather, users select targets by first drawing a line with a pen or finger in the general direction of the desired target. Simple *continuation mark* gestures are then dynamically overlaid on targets within a region of interest defined by this direction. The user then continues their initial stroke by drawing the continuation mark overlaid on the target. We analyze the performance of Gesture Select through two formal experiments. We also explore several extensions of the technique for manipulating remote targets for an extended period, and teaching persistent command gestures.

The contributions of this paper are:

- The design of Gesture Select, a novel selection technique for large displays based on gestures
- Extensions to this technique that support portal invocation and gesture shortcut learning
- A formal experiment which indicates that selection time with Gesture Select significantly outperforms unaided direct selection. Performance changed by <12% as target size was halved/as distance was doubled; analysis suggests that Gesture Select is affected by the extent that users can read the remote disclosure icons, and gesture complexity
- A second experiment indicating significantly improved performance over unaided direct selection and a previous technique for selection, for 2-D labeled targets

RELATED WORK

Applications

Tivoli [25] provided users with an interactive whiteboard environment for supporting meetings. Flatland [23] let users define semantic types for content on the whiteboard, and then manage the content and its history. Guimbretière et al. [11] explored techniques for creating and managing sketches and objects on display walls. Rekimoto [27] explored moving objects between networked computers using pen identification.

Bringing Proxy Targets Closer

Drag-and-pop and drag-and-pick [3] brings a fixed number of remote targets closer to the user at their original size and compresses space between them using a grid-based algorithm, when the user drags in a particular direction. Push-and-pop [7] brings all targets near the user at the original size. This is effective for very sparse target fields; however, it may not scale when the targets are large or numerous, and also requires users to re-find targets, which may be difficult when the target set changes often (e.g. content-manipulation).

Another approach, The Vacuum [5], also uses a sector of influence, but shrinks remote targets, thereby preserving spatial relationships, and also scaling to larger numbers of targets, but incurring the cost that targets are now smaller and thus harder to select. A comparative study of single selection in the presence of path and area distractor targets found no significant difference in performance between The Vacuum, Drag-and-Pick and unaided direct selection. A second study showed benefits for multi-selection. Based on this result we believe there is the potential to create a technique which can outperform unaided direct, single selection on large displays.

Gestures

Marking menus [20] lay out menus using sequences of recti-linear marks, typically in 8 compass-aligned directions; the simple rectilinear marks inspire our gesture design. Flow-Menus [10] integrate marking menus with other actions such as data entry or direct manipulation in one fluid stroke, inspiring our approach of integrating two actions in one stroke.

Escape [34] aids users in selecting small targets on smart phones by displaying each target as a directional icon; users first identify the target's direction, then touch that area of the display, and finally flick in the direction. This gestural selection technique inspires the present work with large displays. Move-and-stroke overlays animated mouse gesture demonstrations on targets; users mimic the gestures with a mouse and then disambiguate errors using a pie menu [8]; though likely slower than pointing for desktop tasks, it can be applied to situations in which there is no pointer, or pointing is costly. Hinckley et al. [14] explored techniques for combining multiple networked pen-based devices into a larger display via gestures. Techniques have been developed for "throwing" objects to remote locations on a large display; we refer to [26] for a summary of work in this area. Systems for teaching gestures on demand such as [6] and [2] have also been developed.

Interacting at a Distance

Several approaches let users interact with display walls at a distance. We believe that, while this is valuable in some situations, there is also value in working at the display as well, e.g. to see high-resolution data up close. While users can "step back" to perform an interaction from a distance, we note the time cost incurred for such transitions may interrupt users' workflow. The "dollhouse" [30] or world-in-miniature approach lets users interact with small proxies for large targets. Bubble radar [1] lets users interact with a miniature of a large display on a tablet with the aid of a bubble cursor. Malik et al. developed techniques for interacting with large displays at a distance using hand gestures over a tabletop [21].

Pointing and clicking interfaces using hand tracking for interacting with large displays from a distance, including ray casting, have been explored [33]. A multimodal technique, the Speech-filtered bubble ray [32] begins with ray casting; the user then speaks a property (e.g. "green"), causing a cursor relative to the ray intersection point to snap to matching targets. This approach showed improved performance over ray casting approaches, however it requires speech, which may be undesirable in local/remote collaborations, and ray casting, which slows at large distances as the effective size of remote targets is small and thus difficult to pick, even if effective size is increased. Ninja Cursor [19] uses multiple cursors to reduce the average distance from cursor to the target, coupled with a delay algorithm for handling ambiguities that arise so a single cursor is active at a time.

Live Clippings of Remote Content

A number of techniques have been developed to allow the user to interact with live clippings of content from other areas of the display. Frisbees [18] let users position a remote target; the contents of this remote target is rendered inside a Frisbee display near the user, from which it can be interacted with. Hopping [17] lets users select targets not currently visible onscreen through a similar interface. The canvas portal framework [4], explored using Frisbee-like portals to access, semantically filter and scale remote content. Shoemaker and Gutwin explored several multi-point interaction techniques that preserve visibility and scale for multiple regions of interest [28]. WinCuts [31] lets users cut out pieces of a live window and interact with these cutouts. These techniques inspire our use of portals. We note, however, that these techniques are best-suited for working with remote content for a period of time, rather than brief transient selections.

DESIGN

Overview

Our approach is founded on the idea of using gestures to select arbitrary remote targets. We believe that this approach has a number of potential advantages:

- A unique gesture can be assigned to each potential target, eliminating the need for pointing
- Walking and searching for a proxy target is not required
- Additional gestural punctuation can be fluidly added to the gesture to modify or extend the interaction

Design Principles

We began the design process with 4 design principles:

Eyes-on: In order to make a selection, users should have to look only at the target itself. More specifically, users should not have to shift their gaze from the desired target to a second location, nor should they have to perform an additional visual search to locate a proxy of the original target to complete a selection, both expensive operations.

Lightweight: The design should be transient, only opening briefly when needed, and should not require an explicit, heavyweight mode switch or tool invocation.

Scalable: The design should be scalable to various scenarios: small/large targets, mid-range/remote targets, dense/sparse arrangements, and up to at least several hundred targets (e.g., a 15'x5' display with 6" targets can hold 300 targets).

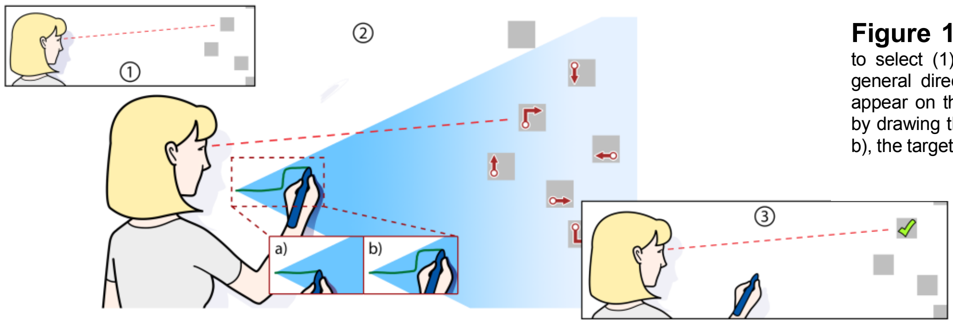


Figure 1. User identifies a remote target she wishes to select (1), draws an initial mark with the pen in the general direction of the target (2-a), continuation marks appear on the targets (2), user continues the initial mark by drawing the continuation mark of the desired target (2-b), the target is selected on pen up (3).

Versatile: The design should scale to support interactions beyond single selection; it should support multi-selection and longer-term interaction with remote targets, when needed.

Region of Interest

Inspired by prior work [3] [5], our technique begins by drawing a line (the *initial mark*) in the direction of the target. Based on this line, a region of interest (ROI) ±26˚ on each side of this line is defined and displayed visually on the screen as a filled pie wedge. As soon as the length of the initial mark exceeds 1.2" in length, the region of interest is shown (the user does not pick up the pen). Pilot testing was used to find an angle that made it straightforward to be imprecise and still open a region of interest containing the desired target. For comparison, [5] used a dynamically adjustable arc angle ranging from ±10˚ to ±60˚.

The region of interest lets us assign simpler gestures to areas nearer the user's target (see below), and also constrains the visual distraction of the selection process to a specific scope.

Continuation Marks

When the ROI is shown, icons are overlaid on targets that lie within the region of interest. Each icon is scaled to fit such that it is inscribed inside the target. Several corner cases exist, such as very small sparse targets, or very small densely packed targets (discussed below). This section discusses the typical case of targets with shortest side approx. 3" or larger.

Each icon depicts a simple rectilinear gesture that we call a *continuation mark*. To select the desired target, the user simply continues the initial mark by now drawing the continuation mark without picking up the pen, thus drawing them together as one fluid stroke. When contact is released, a segmentation algorithm segments the initial mark from the continuation mark; segmentation is necessary since users tend to draw initial marks of varying lengths. The continuation mark is then recognized using a simple mark-based gesture recognizer, and the appropriate target is selected.

Mark Design

For efficiency, we chose rectilinear mark-based gestures, inspired by marking menus [20]. This let us create a fairly large vocabulary of gestures with simple axis-aligned (up, down, left, right) marks. In pilot testing with four users, we found error rates were substantially higher when diagonals were introduced; we believe this is because users drawing the mark are typically looking at the target itself/continuation mark icon, and therefore are not looking at their hand as they draw, since it is out of their field of view, given the large distances. Thus we do not use diagonals.

From this set of possible gestures, we prune several possibilities. First, if the user's initial mark is within 15˚ of a major axis, we prune any marks beginning with that direction, except the single special case of a 1-segment continuation mark that is in the same direction as the initial mark. This solves a potential scale invariance ambiguity problem: if the user draws an initial mark to the left, and then draws a continuation mark that is left followed by down, the system cannot be sure whether the user drew a continuation mark of left followed by down, or just down; in pilot testing, users found it difficult to precisely draw the length of marks without looking. For the same reason, we also prune marks in which two consecutive marks are in the same direction. We observed in initial testing that marks longer than length 3 became too small/perspective-warped to read easily for smaller-sized targets. Therefore, to extend the mark set without requiring additional space, we added "pigtail" marks (Fig. 2, right), defined as loop-like marks with a single self-intersection. We discuss the performance of pigtail marks below.

This produces a base set of 260 marks[1] (after pruning) for 1 – 3 segment gestures. We felt this was sufficient for most large wall applications, since the region of interest will intersect 37.5% of the targets[2] onscreen if opened to the right in the center. We do note, however, that for very large numbers of targets more gestures may be required.

Simple marking menus, in which users draw separate strokes (e.g. 3 strokes for up-right-up) were shown to be more accurate and slightly faster than single-stroke compound marks [36]. However, this requires a "transient mode": if the user has inputted 2 strokes and may input a third, the system must wait, for some time threshold for the third mark, because it does not know whether the user is done. We were concerned adding this mode would add weight to the technique, violating our Lightweight design principle. We therefore used single-stroke marks to keep the mode implicit, although we note accuracy could be improved with simple marks.

Assigning Marks to Targets

Since some marks are more difficult to make than others, we use a simple heuristic to assign marks to targets (see Fig. 2). Each mark has an associated difficulty weight. We sort the targets by their distance from the center line of the region of interest; the simplest marks (that is, the ones with the lowest difficulty weight) are assigned to targets that are closest to this center line, and marks are then iteratively assigned outward by distance in increasing order of difficulty weight.

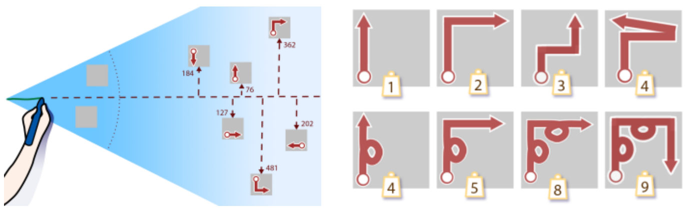


Figure 2. Simpler marks are assigned to targets closest to the center of the region of interest. No gestures are assigned for targets which are very close (left). Difficulty weights of a range of continuation marks (right).

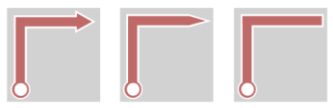


Figure 3. Disclosure icon arrowhead styles; triangular (left), minimal (center), none (right).

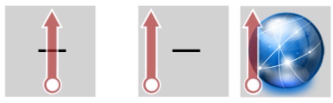


Figure 4. Placing icons on the periphery (right) helps reduce occlusion (left) of centered icons.

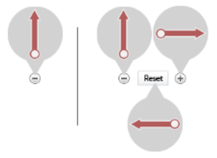


Figure 5. For small, but sufficiently sparse targets, disclosure icons are shown offset in a callout (left); when other callouts are nearby a simple algorithm places them to avoid overlap (right).

Difficulty weight has integer units and is determined primarily by the number of strokes in the mark, one unit per mark; however, there are several additional considerations. Marks containing pigtails have a greater difficulty weight (4 units per pigtail segment) than marks without; marks containing doubling back also have a higher difficulty weight (2 units). This weighting system was implemented based on pilot study observations with four users, in which users ranked the marks by difficulty after completing a study similar to Experiment 1. Note that this approach is heuristic in nature, but we expect it tends to assign the fastest marks closest to desired targets.

Finally, for marks inside the region of interest that are within easy arm's length of the user (27.5"), based on the start point of the initial mark, no continuation marks are assigned. This was done for two reasons. First, we observed that it was faster to select directly when targets are within arm's length, so we would be doing users a disservice by encouraging them to select nearby targets more slowly. Second, by excluding these targets, more "fast" gestures are available for targets that the user actually needs assistance with. We observed during testing that this was intuitive for users, with the intuition being "can I easily reach the target without moving?"; if so then invoking Gesture Select is not needed. We note this might be a problem in an ecologically valid setting with no instruction.

Note that, with the exception of command gestures (see below), gestures are not persistently assigned to a particular target. We explored several possibilities, but concluded that it would be difficult to persistently assign gestures to targets in a predictable way, and so gesture assignment is dynamic. We expect this will not be a major problem since in many cases we believe the target set will change frequently (e.g., user creates new content, etc.). We also note that during pilot testing, no users commented on the fact that gestures changed each time the region of interest was invoked.

Disclosure Icon Design

The disclosure icons are designed to clearly illustrate the continuation mark for a particular target, while attempting to minimize the extent to which the underlying target itself is occluded. To support dense arrays of targets, the icons are inscribed inside the target and scaled (uniform X/Y scale) to fit. The disclosure is *absolute*, that is, it does not depend on the direction of the initial mark.

During pilot testing with 2 users, we explored several forms of the disclosure icons. Users felt it was important that to easily identify the starting point of the mark, so that they can begin drawing quickly, so we indicate this with a white dot. We explored several arrowhead types (Fig. 3). Users preferred the triangular arrowhead to no arrowhead, and the minimal arrowhead because it helped them identify both the starting and the ending point. Icons are rendered semi-transparent and shown in red with a white stroke. This two-color design helps in situations where the target may be all-red or all-white rendering a single-color arrow invisible.

We observed that in many applications, icons are often centered and may be quite small; for example, a minus icon ('-') might indicate zoom out (Fig. 4). To avoid covering such small icons completely with the overlay, we align the icons with the outer edge rather than center them. We found that icons as small as the worst-case minus icon are typically centered, decreasing the likelihood of this occurring.

Small Target Corner Cases

Since our approach relies on disclosure icons, several corner cases must be considered for small targets. While most targets on a large display are likely to be more than 3" wide (indeed, [5] used targets that were 6" wide), there may be some exceptions. For very small targets (< 1.5%, or < 2.7" on a 15' display), the icons are too difficult to read at a distance. For small targets with sufficient nearby space, we display the disclosure icon in a callout (Fig. 5) placed (to the left, right, top or bottom of the target) so that it does not overlap other targets, using a simple, left-to-right greedy algorithm.

The other (somewhat worse) case to consider is a grid of small, tightly packed targets, such as a multiline text box: each character is essentially a small target. To handle this case, we group the small targets together using a simple adjacency-based clustering algorithm, with a join threshold of 1.5", and display a single gesture icon for all targets in the cluster. If the user selects this cluster by performing the gesture, a live copy of the cluster is brought to the user at full size for continued interaction using a Portal (see below).

Integrating Portals

We felt there would be cases where users would want to multi-select or otherwise interact with a remote region for an extended period. To support this, we adapt the Portals [4] approach. Users wishing to interact with a target for an extended period (e.g., to fill out a form and then click Search) can simply draw a closed loop at the end of the target's continuation gesture (Fig. 7). This gestural *punctuation* [35] [13], which can be fluidly added to any continuation mark, opens a

[1] After initial mark direction is pruned, there are 8 1-segment gestures, 36 2-segment gestures, and 216 3-segment gestures.

[2] Computed from the geometric union of a 26° isosceles triangle centered, directed to the right in a rectangle of width/height ratio 4

Portal-like live copy of that region of the screen with default width and height 17.5”x17.5” centered on the target. The user may then interact with the remote content, resize this view by dragging on its border or close it by tapping a close icon.

By integrating portal invocation with Gesture Select, users can open the portal directly over the desired target, without having to invoke, and then manually position a portal target.

Teaching Persistent Command Gestures

Many targets are dynamic in nature, e.g. objects in a document. However, some may be persistent commands, e.g. a Save button. We give applications the option of using Gesture Select to teach gestural shortcuts for persistent commands. Applications can define custom, free-form or mark-based gestures to be associated with a specific target; e.g., the application might assign a spiral gesture to Save. The disclosure icon appears with a different visual style to indicate that this gesture does not change (Fig. 8). Much like marking menus, users could learn over time that Save is done via a spiral gesture, so it would no longer be necessary to invoke the region of interest (expert mode); they could simply draw the spiral to Save. To prevent the region of interest from opening in expert mode, all of the persistent gestures begin with a characteristic prefix; we used a “half-loop” (Fig. 8) which is sufficiently distinct from the straight lines used to invoke the region of interest. It is notable, however, that other prefixes could be used.

Cancelling Invocation

Users may on occasion “change their mind” after opening a region of interest and decide not to execute a command, or may open it in the wrong direction. We observed in pilot testing that users naturally drew a “scribble” when they wanted to cancel, and so implemented this approach (Fig. 6).

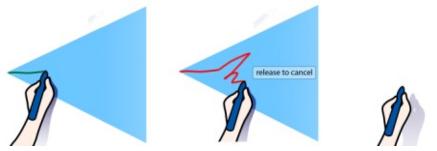


Figure 6. User opens ROI (left) but decides she does not want to execute any commands, so she performs a scribble (center), which cancels (right).

EXPERIMENT 1

Since Gesture Select does not require users to pick a target using a conventional pointing-based approach, we wanted to determine whether this new technique is dependent on target distance or size. We also sought to determine selection time and error rate compared with unaided direct selection.

Note that we do not apply Fitts’ law [9] here. We considered applying this highly successful model of pointing, however, in the case of the large display there is an important issue: walking. When selecting a remote target on a large display directly the user must first walk to the target; once they reach the target, they can then perform a pointing task. The Fitts’-law model is conceived for pointing, and does not account for this walking component. We are unaware of an adaptation of Fitts’ law that has been tested to account for a walking subtask, so we opted to analyze target selection in the presence of varying target size and distance only.

Tasks

A 1-D reciprocal pointing task was used, with the addition of a start target. Two goal targets were spaced an equal distance along the horizontal axis from the red start target. One of the two goal targets, the active goal target, was highlighted in green, while the other (inactive) goal target was gray. Users first tapped the start target and then selected the goal target. (They were required to successfully select the goal target to continue, so that they could not “race through the experiment by clicking anywhere.”) After selecting the active goal target, the active and inactive goal targets switched, with the active goal target now highlighted in green. Users completed four trials for each condition. The start target, based on [5], was used to control the distance between the user and target. Gestures were dynamically assigned (see Design, above), so would not be predictable by the user, and were not the same within each set of reciprocal trials.

We used three distances, near (46” or 26% of display), mid (68.4”, 38%), and far (90”, 50%); all were outside comfortable reaching distance (near was just outside). We used three target sizes; the smallest was based on a simple calculation: in the Windows 7 OS, desktop icons use 6.51% height of a 1024x768 screen; a proportional height on our display is 3.01”. Thus we used target sizes of 3”, 4.5” (+50%), and 6” (+100%). We avoid very small targets as in [5], as we feel these are less representative on a display of this size.

Distractor Targets

In addition to the goal targets, gray distractor targets were placed as well. In pilot testing, we found it was not the arrangement of distractors *per se* (e.g. distractors placed on the path between the user and the target vs. targets placed around the target) that affected Gesture Select, but rather simply the total number of distractors within the region of influence: in general, more targets lead to more complex gestures needed. Therefore, in this experiment, we varied the number of *area distractors* close enough to the target to affect gesture complexity. The distractors were laid out in a dense random arrangement around the goal target. We used three distractor counts: 16, 32, and 48. 48 was the maximum number of targets that would fit vertically in a dense circular formation on our large display for the large target size.

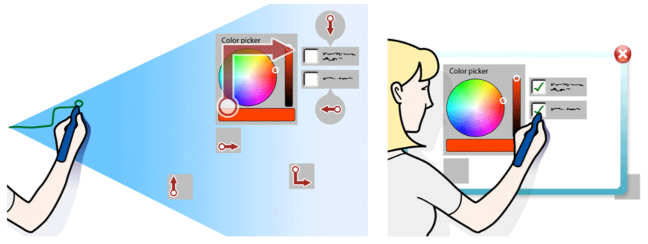


Figure 7. User draws a continuation mark with a loop punctuation at the end (left) to open a portal in place, centered on the selected target (right).

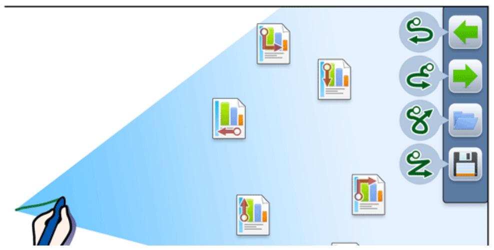


Figure 8. For persistent commands, e.g. Save, gestures can be statically assigned, and will appear in the ROI with different visuals.

Participants and Equipment

Ten participants (aged 18-28, 3 female, all right-handed) were recruited from the general population of Brown University. Two participants reported owning or using a device that has gestures. Participants used a stylus in their right (dominant) hand and were compensated.

We used a 15.4x3.85 ft, short-throw front-projected, tiled large display (effective resolution of 3,072x768). Our large display consisted of a SmartTech SB680 77" (diagonal) interactive whiteboard (input resolution 4000x4000 units) with a short-throw front projector in the center that was flanked on the left and right by two short-throw projectors projecting onto 77" rigid projection screens (Fig. 9).

The center SmartBoard-based portion could sense pen or touch contacts, whereas the two flanking screens were passive/projection only. The lack of input on the passive screens was not a problem for two reasons: (1) for Gesture Select there was no reason to walk any distance to pick remote targets, and the start target kept users centered after each task, and (2) for unaided direct selection (only) we used a camera-measurement/Wizard-of-Oz approach. Video cameras recording the session displayed on screens hidden from the user, allowed the "Wizard" to advance to the next task when appropriate. Note that the Wizard's actions only advanced the task; videos were analyzed offline at the frame level to get timing/accuracy information. This approach is similar to prior work, e.g. [12]. The effect of shadows from user's hands was minimal as the short throw allowed the projectors to be placed very close to the surface of the screen (shadows were below users' hands). Our software was C# and WPF-based, and ran on 3 networked dual-core computers with 2GB of RAM; each computer outputting to one projector.

Experimental Design

A repeated-measures full factorial within-participant design was used, with independent variables: technique (Unaided Direct Selection, Gesture Select), distance (46", 68.4", 90"), width (2", 4", 6"), and distractor count (16, 32, 64):

10 participants
x 2 techniques
x 1+3 blocks (training + measured)
x 3 distances
x 3 widths
x 3 distractor counts
x 4 trials
= 8,640 trials completed

Results

Selection Time

The results for selection time are presented in Fig. 9, left by distance, Fig. 9 center by target size, and Fig. 9, right by distractor count. There was a significant main effect of technique on selection time ($F_{1,18}$=27.66, p=0.001). There was also a significant effect of distance ($F_{2,27}$=231.07, p<0.0001), target size[3] ($F_{1.28,11.23}$=30.62, p<0.0001), and distractor count ($F_{2,27}$=26.16, p<0.0001). There was a significant technique × distance interaction effect[1] ($F_{1.34,25.01}$=142.38, p<0.0001) on task completion time. There was also a significant technique × distractor count interaction effect ($F_{2,27}$=12.50, p<0.0001). There were no other interaction effects (p>0.05).

Post-hoc pairwise means comparisons were conducted with 2-tailed t-tests using Holm's sequential Bonferroni adjustment for multiple means comparisons [15]. Gesture Select was significantly faster than unaided direct selection for 21 of 27 conditions (p<0.0013); six near conditions[4] were not significant. Two of the near conditions were significant, few distractors with large and medium-sized targets (p<0.0013).

Thus, Gesture Select significantly outperformed unaided selection for the mid (46.20%) and far distances (66.20%).

Effect of Target Size on Selection Time

Gesture Select performed significantly worse for small targets than large targets (t_9=10.60, p<0.0001), a mean difference of 189.81 ms (8.9%). However, there was no significant difference between small and medium-sized targets (t_9=3.04, p=0.014), or between medium- and large-sized targets (t_9=1.52, p=0.164).

Effect of Target Distance on Selection Time

Interestingly, Gesture Select was affected by distance: near targets were significantly faster to select than medium-distance targets (t_9=-5.37, p<0.0001), a mean difference of 109.91 ms (5.7%). Near targets were significantly faster to select than far targets as well (t_9=-5.09, p<0.0001), a mean difference of 219.88 ms (11.5%). There was no significant difference between medium-distance and far targets, however (t_9=-3.19, p=0.011).

Effect of Distractor Count on Selection Time

Gesture Select was affected by distractor count: performance with 16 distractors increased 14.54% to 48 distractors, which was significant (t_9=-8.13, p<0.001), 16 to 32 was 10.07% higher, which was significant (t_9=-7.49, p<0.001), and 32 to 48 was 4.06% higher, which was significant (t_9=-3.84, p<0.001). Unaided direct selection had no significant difference in selection time across distractor counts (p>0.05).

Errors

Technique had a significant main effect on error rate ($F_{1,18}$=27.04, p<0.001). Unaided selection had a mean error rate of 1.14%, while Gesture Select had a mean error rate of 6.08%. Distance, target size, and distractor count had no significant effect on error rate (p>0.05).

Keystroke Level Analysis

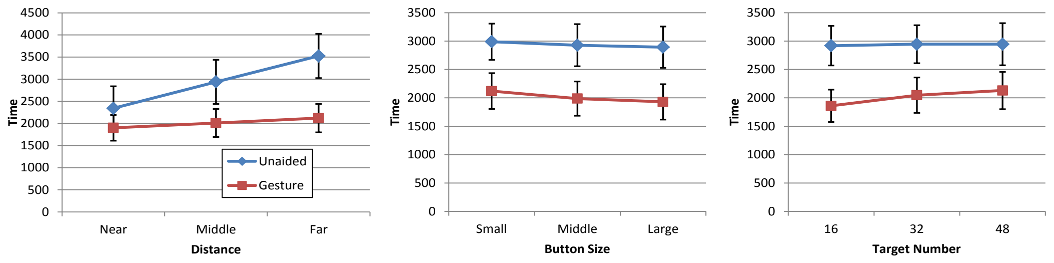
We performed a keystroke level analysis of the data to determine how Gesture Select was used (Fig. 10). We modeled Gesture Select performance, T, as 5 variables: initial reaction time, T_0; time to open the region of interest, T_{ROI}; time to stop moving after the region of interest opens, T_S; time to read/identify the gesture disclosure icon, T_R; and finally the time to draw the gesture and release contact, T_G.

$$T = T_0 + T_{ROI} + T_S + T_R + T_G$$

On average, users spent 39.6% of the time on initial reaction time and opening the region of interest, 12.1% of the time on

[3] The sphericity assumption was not met, so the Huynh-Feldt correction was applied; the corrected degrees of freedom is shown.

[4] Near, small, some distractors (t_9=2.28, p=0.049); near, small, many distractors (t_9=1.03, p=0.33); near, middle, few distractors (t_9=2.80, p=0.02); near, middle, some distractors (t_9=2.18, p=0.06); near, middle, many distractors (t_9=0.99, p=0.005), near, large, some distractors (t_9=2.83, p=0.02); near, large, many distractors (t_9=2.07, p=0.07).



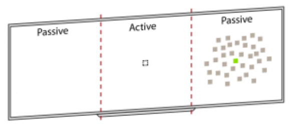


Figure 9.

Distance (left), target size (center), distractor count (right) vs. selection time. Experimental layout (above).

stopping movement after opening the ROI, 16.3% of the time reading/identifying the gesture icon, and 32% of the time drawing the gesture. Stopping movement and reading/identifying the gesture together took 28.4% of the time, very close to the 32% of the time drawing the gesture.

From near to far targets, $T_0 + T_{ROI}$ increased 5.2%, T_S increased by 11.0%, T_R increased by 17.2%, and T_G increased by 12.8%. From large to small targets, $T_0 + T_{ROI}$ increased 5.1%, T_S increased by 5.1%, T_R increased by 10.3%, and T_G increased by 15.6%. It is interesting to note that opening the ROI increased just 5%, while reading and drawing the gesture slowed down. This suggests that opening the ROI was not the source of the increase in selection time, but rather that identifying, reading, and copying the gesture was harder to do when it was smaller/more perspective-warped. Indeed, users commented that small targets were harder to see at the farther distances; we believe this may explain why only small targets were significantly affected (see above).

Performance Stability Across Blocks

There was no significant effect for block number on selection time[1] ($F_{1.19,16.06}$=0.42, p=0.57), or error rate ($F_{2,27}$=0.30, p=0.74). There was also no significant technique × block number interaction for selection time ($F_{2,27}$=0.30, p=0.75), or error rate ($F_{2,27}$=0.27, p=0.77). We attribute the apparent lack of a learning effect to training. In terms of gesture complexity, 23.7% were 1-segment gestures, 35.4% were 2-segment, 16.1% were 2 segment including 1 double-back, 22.0% were 3-segment, 2.4% were 3-segment including 1 double-back, and 0.43% were 3-segment including 2 double-back.

EXPERIMENT 2

In Experiment 1 we determined that Gesture Select can outperform unaided selection for medium-distance and far targets, and that performance is affected by target size, distance and distractor count. However, Experiment 1 may not have been sufficiently representative of real-world tasks in that its selection was 1-D, goal targets were highlighted, and users did not identify targets via a visual icon or label. Thus, the goal of Experiment 2 is to compare performance of Gesture Select to a previous technique, the Vacuum [5], and to unaided direct picking, for 2-D, single selection of labeled targets. We hypothesize Gesture Select will perform similarly to Experiment 1, and outperform the other techniques.

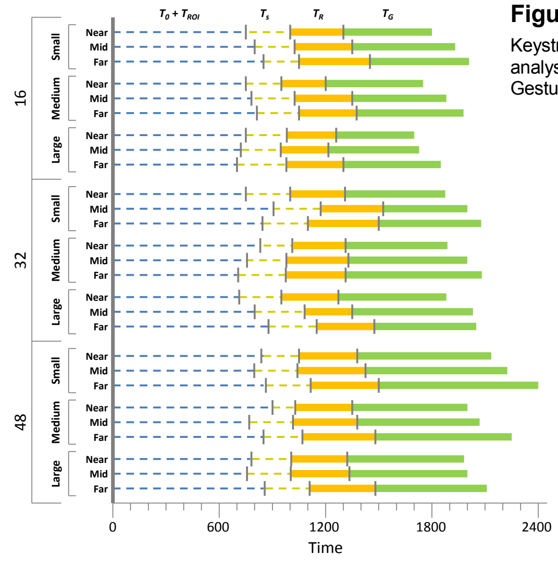


Figure 10.

Keystroke level analysis of Gesture Select.

Since the goal of our technique is similar to The Vacuum, we adapted the experimental design closely from [5]. We did not include Drag-and-Pick as [5] found no significant difference in selection time between Drag-and-Pick and the Vacuum (not including hover dismissal time). Since the Vacuum can scale to a greater number of targets but had very similar performance to Drag-and-Pick, we chose to compare our approach to the Vacuum. We thus implemented a slightly simplified version of the Vacuum for single selection that included no hover dismissal (automatically dismissed after user makes a selection).

Participants and Equipment

We recruited 10 participants from the general population of Brown University (ages 19-27, 4 female, all right-handed). As in Experiment 1, participants used a stylus in their right (dominant) hand and were compensated. Three participants reported owning or using a device that has gestures. The same equipment was used as in Experiment 1 (Fig. 9).

Tasks

Users first tapped a start target and then selected a goal target from amongst distractor targets. As in [5], targets were rendered as gray squares each with a number inscribed; the goal target was gray, like the distractor targets, but was always numbed '1' whereas the distractor targets had other distinct numbers. Since the goal target was not highlighted, users had to identify it via its unique label. The start target let us control the distance to the goal target. Participants could not continue to the next task until correctly selecting the goal target. As in [5], all targets were visible from the trial's beginning. For consistency with [5], a stylus was used for all tasks.

Experimental Design

We used a repeated measures within-subjects full factorial design with controlled variables technique (Direct, Vacuum, Gesture Select), distance (63.3", 78.2"), direction (E, NE, SE, W, SW, NW), and path distractor density (0%, 40%, 80%):

10 participants
x 3 techniques
x 4 blocks (1 training, 3 measured)

x 2 distances
x 6 directions
x 3 path distractor densities
= 4,320 trials completed

The study lasted approximately one hour, and was divided into three parts by technique. For each technique, participants completed 4 blocks, each of which included all combinations of distance, direction, target size, and path distractor density. The first block for each technique was a training block and was not included in the analysis. The condition order was counterbalanced with randomization.

In [13], one target size was used: 6”. We felt that 6” targets, while representative of some applications, are larger than many items users might want to select, e.g. icons or command buttons (see Windows 7 OS typical case analysis, above). Therefore, we used 3” targets, as in Exp. 1. As in [13], we used two target distances (the distance between the start target and the goal target), 63.3” and 78.2” (35.1% and 42.4% of display width). Also as in [13], we used 6 target directions, relative to the start target; excluding the N and S directions, as this requires reaching to uncomfortable positions. For this reason, the diagonals were at angles ±17.8° (for 63” distance) and ±14.3° (for 78.2”) to the horizontal axis; this produced changes in height of 46”.

The start target was placed in the center of the screen. As in [13], users were required to select the start target before selecting the goal target. Also as in [13], all targets were visible at the beginning of the trial, mimicking a scenario in which users are familiar with the interface layout. Emanating from this point, in the six directions, at each distance, goal targets were placed. As in [13] we used two types of distractors: path and area distractors. As in [13], path distractor targets were added between the start and the goal targets, filling in 0%, 40%, or 80% of the space between. We found in Experiment 1 that the number of targets inside the ROI had a significant effect on Gesture Select performance; thus, we used 35 area distractors placed densely around the goal targets, supplementing the path distractors. Users had to complete tasks correctly (see above).

Results

Time

Technique had a significant main effect on task completion time ($F_{2,27}=13.58$, $p<0.0001$), as did distance ($F_{1,18}=63.54$, $p<0.0001$). There was also a significant technique × distance interaction[1] ($F_{1.27,23.71}=22.52$, $p<0.0001$). Path distractor density did not have a significant effect ($F_{2,27}=1.98$, $p=0.17$). There was, however, a significant technique × path distractor density interaction ($F_{4,85}=7.79$, $p<0.0001$). Post-hoc tests were conducted with 2-tailed t-tests, with Holm’s sequential Bonferroni adjustment [36] for multiple comparisons.

Gesture Select significantly outperformed Vacuum for all conditions ($p<0.0025$). Gesture Select also significantly outperformed unaided direct selection for all conditions ($p<0.0025$) with one exception: near + high-density ($t_9=3.26$, $p=0.010$). Consistent with [13], there was no significant difference between Vacuum and Unaided for any condition ($p>0.05$). Gesture Select was affected by distance, increasing 7.55% from mid-far ($t_9=-6.04$, $p<0.001$), as was unaided selection 24.02% ($t_9=-10.06$, $p<0.001$). The Vacuum was not significantly affected by distance ($t_9=0.24$, $p=0.82$).

Interestingly, path distractor density induced no significant difference in selection time for Gesture Select ($p>0.05$). The Vacuum, however, did show an increase in selection time between 0% and 80% of 11.75%, but this was not significant ($t_9=-2.70$, $p=0.025$), and between 40% and 80% distractor densities of 8.60%, also not significant ($t_9=-2.66$, $p=0.026$).

This result shows that in the presence of substantial numbers of distractor targets, in a 2-D selection task where targets are not highlighted, Gesture Select significantly outperforms unaided selection and the Vacuum, and performs comparably to the idealized task in Experiment 1.

Gesture Complexity

A histogram of the gesture difficulty classes of goal targets is shown in Fig. 11. Interestingly, 32.92% were 1 and 2-weight gestures, with 51.67% 3-weight, and less than 15.42% higher weight gestures. This indicates complex gestures with weight greater than 3 were rarely used, despite the large number of targets onscreen (see above), and the fact that most of the gesture set have weight greater than 3 (90.4%). This suggests the region of interest, and the heuristic of assigning simpler gestures to the ROI center may have simplified gestures over randome assignment, as that would have assigned greater numbers of ≥ 4 weight gestures given the distractor target count. The mean selection time for each class is shown in Figure 11 as well (note: weight 5 has few samples). 2-weight gestures were 10.37% slower than 1-weight, and 3-weight were 9.19% slower than 2-weight. Interestingly, 4-weight gestures increased 4.60% from 3, and 5-weight increased 6.06% from 4. Overall, there was an increase of 26.05% from weight 1 to weight 4. This suggests the weight heuristic helped represent the increasing gesture difficulty.

Errors

Technique had a significant main effect on error rate ($F_{2,27}=14.25$, $p<0.001$). Distance and density had no significant effect on error rate ($p>0.05$). There were also no interaction effects between distance, density and technique on error rate ($p>0.05$). Post-hoc multiple means comparisons were conducted, using the same adjustment procedure as above. Unaided direct selection had an error rate of 1.85%, while the Vacuum had an error rate of 6.54%, and Gesture Select 8.24%. There was a significant increase in error rates from unaided direct selection to the Vacuum ($t_9=-3.57$, $p<0.025$), and a significant increase in error rates from unaided direct selection to Gesture Select ($t_9=-6.79$, $p<0.017$). Interestingly, there was no significant difference in error rates between Vacuum and Gesture Select ($t_9=-1.29$, $p=0.23$). We note that Vacuum error rates and selection times are higher than those reported in [13]; this is likely the result of the smaller targets used, both directly in that smaller targets are harder to select, and also as there were more individual targets to search, as the path distractors are density-based.

Subjective Preference

When asked which of the techniques was the fastest, 10 of 10 users chose Gesture Select. When asked which of the tech-

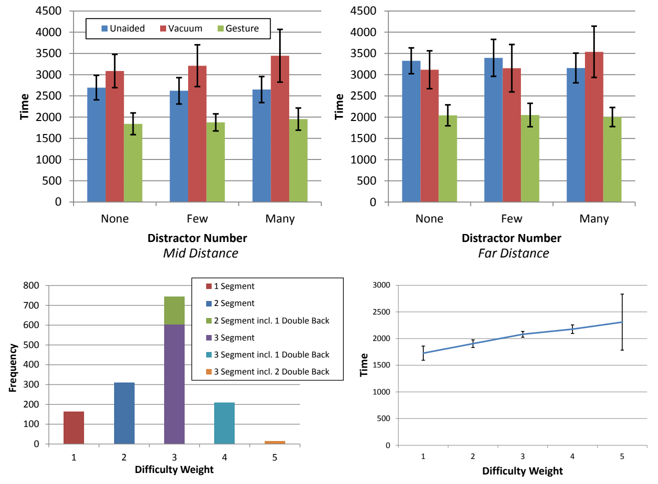


Figure 11. Performance results for medium-distance and far distances (top), histogram of gesture difficulty weight, with gesture complexity breakdown (bottom, left), performance by difficulty weight (bottom, right).

niques was most accurate, 8 of 10 chose unaided selection, 1 chose Vacuum, and 1 chose Gesture Select. This is not surprising, since the large target size made it difficult to miss with unaided selection. When asked which of the techniques was preferred overall, 9 of 10 users chose Gesture Select, and 1 of 10 chose Vacuum. Users felt Gesture Select required "little physical effort" was "really cool" and was "fast and easy to use." The user who preferred Vacuum felt it was useful to be able to see the targets in miniature. Interestingly, despite the fact that 7 users did not use gestural devices, no users mentioned difficulty learning or performing gestures.

Gesture Recognizer Analysis

To determine to what extent the errors for Gesture Select were the result of user error vs. bugs in the recognizer an independent, human, single-blind verification of the recorded gesture performances was conducted. For each participant, 5% of the recordings were sampled randomly, and then recognized using a simple "recipe" set up *a priori*. Beyond visual comparison with a crib sheet, the verifier marked a gesture "unknown" if it was ambiguous, if a mark segment was >80% smaller/larger than other segments, or if there was <45° of angular difference between two consecutive marks.

Overall, there was 97.1% agreement between the human and software recognizer. When looking at trials considered by the system to be successful attempts, there was 100% agreement, indicating false positives were minimal. When looking at trials considered by the system to be failed attempts we found 64.7% agreement, indicating that as much as 35.3% of Gesture Select errors could be false negatives. This suggests user error rate could potentially be lowered with an improved recognizer; it is also possible selection time could be improved as well. Of the gesture errors as a whole, 23.5% were cases in which the user performed the wrong gesture, 41.2% were failures to correctly perform the gesture, and 35.3% were recognition failures.

Performance Stability Across Blocks

There was a significant effect of block number on selection time ($F_{2,27}$=4.73, $p<0.05$), but not for error rates ($F_{2,27}$=1.96, p=0.17). There was also a significant technique × block number interaction for selection time ($F_{4,85}$=6.68, $p<0.05$), but not for error rate ($F_{4,85}$=1.41, p=0.25). The Vacuum improved performance by 257 ms (7%) over the experiment, while the other two techniques changed less than 3%. Informally, we observed that earlier on in the experiment users had a tendency to spend more time adjusting the Vacuum's size, compared with minimal adjustments later on.

DISCUSSION

The controlled nature of the study may limit the generality of the results. We also did not simulate distractions, situational awareness, or collaboration (see below), which might affect performance in ecologically valid situations.

Gesture Select significantly outperformed the Vacuum, as well as unaided direct selection in selection time in Experiment 2, and was preferred overall by 9 of 10 users. It is notable that distance had a 7.55% impact, but path distractors had no significant effect on Gesture Select performance.

Interestingly, despite not involving pointing, Gesture Select is affected by target size and distance. In Experiment 1, we found a 9.84% significant difference between 3" and 6" targets, but no other significant differences. We also found that nearby targets were 5.78% faster to select than medium targets, and 11.56% faster than far targets, but no significant difference between medium and far targets was found.

A keystroke level analysis of Gesture Select performance suggests that small or far targets are harder to see, due to smaller size and perspective warping, thus slowing the gesture identification and copying process. This was consistent with user comments to this effect. This suggests that for a given target size, there may be an effective limit on how far the targets can be from the user and still be legible. A possible solution to this problem for very far targets, would be to apply the clustering approach already used for very small targets, and cluster the targets into larger groups, overlaid with large disclosure icons; executing these gestures could then open a portal.

Performance of Gesture Select was also affected by gesture complexity. We saw a 26.05% increase in selection time from weight 1 to weight 4 gestures, for example. The heuristic of assigning simpler gestures nearer the center of the ROI appeared to benefit users, as a disproportionate number of weight 1, 2 and 3 gestures were used (see above), as there are fewer such gestures. Thus Gesture Select has the interesting property that the number of nearby targets affects performance.

For accuracy, Gesture Select and the Vacuum have a higher error rate than unaided selection. However, there was no significant difference in error rate between the Vacuum and Gesture Select. Gesture Select had an error rate of 6.5% in Exp. 1, and 8.2% in Exp. 2. A single-blind human analysis of the errors in Exp. 2 revealed that as many as 35.3% of these errors were false negatives caused by software recognizer imperfections. We hypothesize that further refinement, or by using simple marks [36], error rates could be reduced. Distance, target size, and distractor count had no significant effect on error rate.

We believe these results are promising, and indicate Gesture Select represents a performance improvement for single selection over the techniques tested.

LIMITATIONS AND FUTURE WORK

There are several limitations in the present work. We did not evaluate Gesture Select in a colocated collaboration scenario; it is possible that the icon overlays could distract other users. In addition, it may be difficult to see disclosure icons if occluded by other users. Given this positive initial result, further work is warranted to evaluate Gesture Select in collaborative scenarios to determine if refinements are required. Another limitation of the technique is that, if more targets are in the region of interest than 260 (the number of available gestures) targets near the edge of the region of interest will not be selectable. However, we expect this problem will be relatively rare, since targets on a display of this size are likely to be large and since we group very small, dense targets together using clustering. The recognizer used in our prototype is imperfect. In a product implementation, a more robust recognizer would be required, or simple marks [36] could be used. Extensions of the core Gesture Select technique were not evaluated formally. Future work is warranted to determine how performance of these extensions compares with prior approaches. In addition, while our technique is usable with direct-touch input as well as pen input, direct-touch input was not tested. Gesture Select is not inherently self-disclosing. A gesture disclosure approach such as [6] could potentially be adapted to address this approachability issue.

CONCLUSION

We have presented Gesture Select, a novel technique for selecting remote targets on large displays. We further presented several extensions to this design for working with remote content for an extended period, and teaching gesture shortcuts for commands. The results of a formal experiment indicate that Gesture Select significantly outperforms direct selection for mid/far targets, and that selection times are affected by target size, and distance within 12% when doubling the distance, and halving target size. Gesture complexity, driven by the number of targets in the ROI had a 15% impact. Further analysis suggests that Gesture Select is principally affected by the extent to which users can easily read the gestures, and the complexity of the gestures in the ROI. The results of a second experiment indicate Gesture Select significantly outperforms unaided direct selection and an existing technique in selection time.

ACKNOWLEDGEMENTS

We wish to thank Andries van Dam for his advice and insight, Alice Liu for her illustrations, and Ronald Dunleavy, Steven P. Reiss, Noah Zimmt and Robert Zeleznik for their assistance. This material is based upon work supported under a National Science Foundation Graduate Research Fellowship and in part by NSF grant CCF-1012056.

REFERENCES

1. Aliakseyeu, D., Nacenta, M., Subramanian, S., and Gutwin, C. Bubble Radar: Efficient Pen-Based Interaction. In *Proc. of AVI'06*.
2. Bau, O. and Mackay, W. OctoPocus: a dynamic guide for learning gesture-based command sets. In Proceedings of UIST'08, 37-46.
3. Baudisch, P., et al. Drag-and-Pop and Drag-and-Pick: techniques for accessing remote screen content on touch-and-pen operated systems. In *Proc. of INTERACT'03*.
4. Bezerianos, A. et al. View and Space Management on Large Displays. *IEEE Computer Graphics and Applications*, 25, 4 (2005).
5. Bezerianos, A. and Balakrishnan, R. The vacuum: facilitating the manipulation of distant objects. In *Proc. of CHI'05*, 361-370.
6. Bragdon, A., Zeleznik, R., et al. GestureBar: improving the approachability of gesture-based interfaces. In *Proc. of CHI'09*.
7. Collomb, M., Hascoët, M., Baudisch, P., and Lee, B. Improving drag-and-drop on wall-size displays. In *Proc. of GI'05*, 25-32.
8. Fekete, J., Elmqvist, N., and Guiard, Y. Motion-pointing: target selection using elliptical motions. In Proc. of CHI'09, 289-298.
9. Fitts, P. M. The information capacity of the human motor system in controlling the amplitude of movement. *J. Exp. Psych.*, 47, 6 '54.
10. Guimbretiere, F. and Winograd, T. FlowMenu: Combining Command, Text, and Data Entry. In *Proc. of UIST'00*, 213-216.
11. Guimbretière, F., Stone, M., and Winograd, T. Fluid interaction with high-resolution wall-size displays. In *Proc. of UIST'01*, 21-30.
12. Harrison, C. and Hudson, S. Providing dynamically changeable physical buttons on a visual display. In *Proc. of CHI'09*, 299-308.
13. Hinckley, K., Baudisch, P., et al. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. *CHI'05*.
14. Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., et al. Stitching: Pen gestures that span multiple displays. In *AVI'04*.
15. Holm, S. A Simple Sequentially Rejective Multiple Test Procedure. *Scandinavian Journal of Statistics*, 6 (1979), 60-65.
16. Humphreys, G. and Hanrahan, P. A distributed graphics system for large tiled displays. In *Proceedings of IEEE VIS'99*, 215-223.
17. Irani, P., Gutwin, C., and Yang, X. Improving selection of off-screen targets with hopping. In *Proceedings of CHI'06* (), 299-308.
18. Kahn, A., Fitzmaurice, G., Almeida, D., et al. A Remote Control Interface for Large Displays. In *UIST'04*.
19. Kobayashi, M. and Igarashi, T. Ninja cursors: using multiple cursors to assist target acquisition on large screens. In *CHI'08*.
20. Kurtenbach, G. and Buxton, W. The limits of expert performance using hierarchic marking menus. In *Proc. of CHI'93*.
21. Malik, S., Ranjan, A., et al. Interacting with Large Displays from a Distance with Vision-Tracked Multi-Finger Gestural Input. *UIST'05*.
22. Mitsubishi. *MegaView Data Wall*. mitsubishi-megaview.com.
23. Mynatt, E., Igarashi, T., Edwards, W., and LaMarca, A. Flatland: new dimensions in office whiteboards. In *Proceedings of CHI'99*.
24. Ni, T., Schmidt, G.S., Staadt, O.G., Livingston, M.A., et al. A Survey of Large High-Resolution Display Technologies, Techniques, and Applications. In *Proc. of IEEE VR'06*.
25. Pederson, E., McCall, K., Moran, T. P., et al. Tivoli: an electronic whiteboard for informal workgroup meetings. In *INTERCHI'93*.
26. Reetz, A., et al. Superflick: a natural and efficient technique for long-distance object placement on digital tables. In *GI'06*.
27. Rekimoto, J. Pick and drop: A direct manipulation technique for multipe computer environments. In *Proc. of UIST'97*, 31-39.
28. Shoemaker, G. and Gutwin, C. Supporting Multi-Point Interaction in Visual Workspaces. In *Proc. of CHI'07*, 999-1008.
29. Smart Technologies. *Large-wall Touch Displays*. smarttech.com.
30. Swaminathan, K. and Sato, S. Interaction for Large Displays. interactions, 4, 1 (1997), 15-24.
31. Tan, D., Meyers, B., et al. WinCuts: manipulating arbitrary window regions for more effective use of screen space. *CHI'04EA*.
32. Tse, E., et al. Speech-filtered bubble ray: improving target acquisition on display walls. In *Proc. of Multimodal interfaces*'07.
33. Vogel, D. and Balakrishnan, R. Distant freehand pointing and clicking on very large, high resolution displays. In *Prc. of UIST'05*.
34. Yatani, K., Partridge, K., et al. Escape: a target selection technique using visually-cued gestures. In Proc. of CHI'08.
35. Zelenik, R. and Miller, T. Fluid inking: augmenting the medium of free-form inking with gestures. In *Proceedings of GI'06*, 155-162.
36. Zhao, S. and Balakrishnan, R. Simple vs. compound mark hierarchical marking menus. In *Proceedings of UIST'04*, 33-42.