Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays

Daniel Vogel, Ravin Balakrishnan
Department of Computer Science
University of Toronto
dvogel | ravin @dgp.toronto.edu

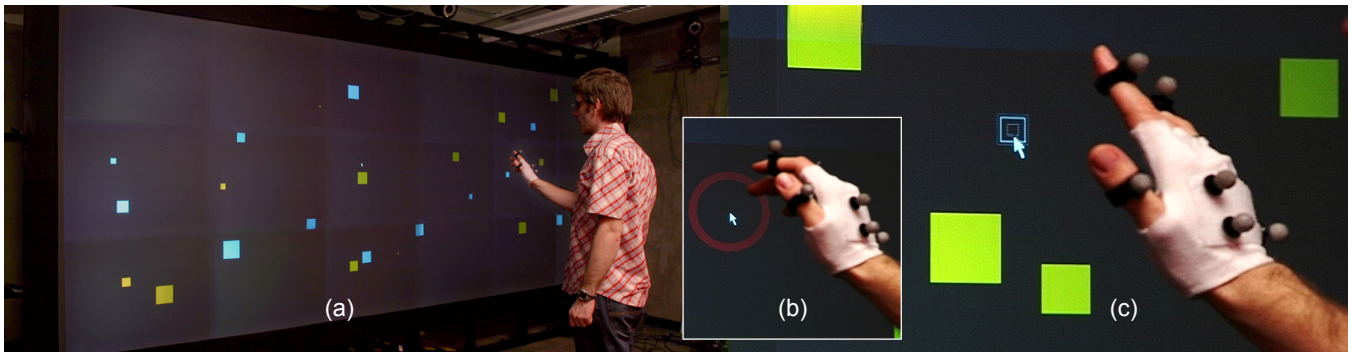

Figure 1. (a) very large (5m x 1.8m), high resolution (6144 x 2304 pixels) display; (b) visualization showing ambiguous posture threshold warning; (c) the hand controls pointer position and makes "click" selection with finger or thumb.

ABSTRACT

We explore the design space of freehand pointing and clicking interaction with very large high resolution displays from a distance. Three techniques for gestural pointing and two for clicking are developed and evaluated. In addition, we present subtle auditory and visual feedback techniques to compensate for the lack of kinesthetic feedback in freehand interaction, and to promote learning and use of appropriate postures.

Categories and Subject Descriptors: H.5.2 [**User Interfaces**]: Interaction styles; I.3.6 **[Methodology and Techniques]**: Interaction techniques.

General Terms: Design, Experimentation, Human Factors

Additional Keywords and Phrases: very large displays, freehand gestures, whole hand interaction, pointing

INTRODUCTION

As displays increase in size and resolution while decreasing in price we will soon have entire walls providing high resolution visual output. These very large, high resolution displays will allow users to work up close with detailed information and also enable them to step back and manipulate the contents of the entire display space.

There are some tasks that are best performed from a distance: for example, sorting slides/photos/pages spread over the large display, or presenting a large drawing to a group while navigating/panning/highlighting. Because of their size and architectural context, these displays can be used in a more casual manner similar to a large physical whiteboard or paste up design space. There are also circumstances where users cannot easily approach the display and can interact only from a distance. Consider a central control room used to monitor large systems like a railway, or a large display mounted out of reach in a public place like an airport.

Direct manipulation through pointing and clicking remains by far the dominant interaction paradigm in conventional user interfaces. Although alternatives like gesture-based interfaces have been explored, the self-revealing nature, simplicity, and flexibility of the point and click metaphor is hard to beat. When a display surface can sense touch, selecting items by tapping with your finger or a pen is immediately appealing, as it mimics real world interaction.

But what happens when we are farther away from the display? Proposed solutions to distant point and click interaction include using 3D input devices such as a flying mouse or hand-held isometric input [12, 32], and laser pointer-style devices [18, 20, 21]. However, relying on a hand-held isometric or isotonic device can make the transition from distant to close interaction awkward. Although laser pointers can become "touch pens" when used on the display surface, with "on again, off again" casual interaction, a physical device must be acquired and released, and may even become misplaced.

Our work investigates potential techniques for pointing and clicking from a distance using only the human hand. This eliminates issues with acquiring a physical input device, and transitions very fluidly to up close touch screen interaction. Although we use a commercial motion tracking system with reflective markers on the hand for developing and evaluating these techniques, computer vision is approaching robust, real time tracking of bare hand postures and movement in 3D space [19], thus making bare hand interaction a realistic possibility in the near future.

UIST'05, October 23–27, 2005, Seattle, Washington, USA.

DESIGN CHARACTERISTICS

Commonly available point and click input devices make assumptions about the user's spatial relationship to the display. They assume the user is always near the display for pen and touch screen input, or near a stationary horizontal surface when using a mouse. With a very large, high resolution display users will be manipulating detailed information up close, and also stepping back to see and manipulate the entire display from a distance. This physical movement creates an interesting problem when determining what input technique to use, as we can no longer assume a reasonably fixed spatial relationship between the user and display. The pointing device needs to move with us and be instantly available to perform selections. Whatever it is, we not only need to carry it, but may also need to hold it up without the luxury of a desk to rest on. Thus, we are led to a set of desirable characteristics for pointing and selection devices suitable for very large, high resolution displays:

1) Accuracy: The device must be able to reliably select small targets both from a distance and up close. Although the user interface could be tailored to show only large targets and make distant selection easier, this is not a realistic design because it squanders the potential benefits of high resolution up close interaction with the display. Also, considering that most states require 20/40 corrected vision to receive a driver's license, most people can read an 11.6 mm symbol from a distance of 4 m[1] [17]. Thus even from a distance of 4m, a target size of 16 mm is quite reasonable.

2) Acquisition Speed: The more casual "on again, off again" use of this type of large display means users will not necessarily be interacting with it for extended periods of time and may perform other activities in between interactions with the display. Zhai [32] argues that a major problem with 3D input devices operated from a distance is device acquisition. A cursor controlled by such an input device does not "stay put" when the device is physically released, unlike a conventional mouse. Thus, the acquisition and release of the pointing device should require minimal effort and be instantaneous.

3) Pointing and Selection Speed: In spite of the large pixel count and physical size of the display, when used from a distance the device should be able to move to any location on the display quickly with minimal or no clutching. The selection (click) itself should also be easily executed.

4) Comfortable Use: As with any device, it should be easily understood and simple to operate. Since we can no longer assume proximity to the display or a desk, the device may be operated with a single hand, possibly in free space. This may introduce fatigue and strain if not carefully designed. Hinckley et al. [9] caution that ergonomics for spatial control are quite different than typing, and an emphasis should be placed on techniques that avoid or reduce fatigue.

5) Smooth Transition Between Interaction Distances: The pointing device should smoothly transition from up close interaction, to interaction at a distance from the display. This implies that the way in which the device is operated should be consistent regardless of its distance from the display. However, this consistency should also preserve direct touch interaction when up close to the display since the affordances of direct touching are so strong.

PREVIOUS WORK

Hand-held Indirect Pointing Devices

Proposed solutions to distant point and click interaction include using 3D input devices such as isotonic flying mice or hand-held isometric input [12, 32]. An isometric input device doesn't require the movement of the device itself in space, which makes it much less tiring than a freely held isotonic device. However, a hand-held isometric or isotonic device makes the transition from distant to up close interaction awkward because the device has no direct mapping when used on a touch-enabled surface.

Laser Pointer-Style Devices

Various researchers have explored the use of laser pointers as input devices for very large screen interaction [14, 15, 18, 20, 21, 23]. Laser pointer-style devices have the advantage that they can become "touch pens" when used directly on the display surface (i.e., the ray emanating from the device nears zero length but is still usable). However, these ray casting devices are notoriously inaccurate at a distance due to hand jitter. Myers et al. [18] compared laser pointers to other devices in pointing tasks. They found laser pointers performed the worst, with at best 4 pixel selection accuracy even after predictive filtering, but had good results with their Semantic Snarfing technique. Oh and Stuerzlinger [20] designed a computer controlled laser pointer, but their experiments showed error rates around 40% when selecting relatively large 40 pixel diameter targets. Olsen and Nielsen [21] cleverly designed laser pointer interaction techniques optimized to avoid hand jitter issues as much as possible. Wilson and Pham [31] use relative movement of a wand to control a motor actuated laser beam, and discussed the merits of relative vs. absolute control. Parker et al. [22] found that laser-pointer like devices were faster than direct touch for large targets on a table top display and developed a hybrid technique called TractorBeam.

Eye Tracking

Researchers have investigated eye gaze to control a cursor and make selections in conventional interfaces [8] and on large displays [2, 27]. However, problems with finding a suitable selection "click" mechanism coupled with involuntary saccade movements make it difficult to use eye gaze effectively for precision pointing and selection. The EyeWindows system [6] more appropriately uses eye gaze for the relatively coarse pointing task of focusing a GUI window. The MAGIC pointing technique [33] also appropriately uses eye tracking for coarse contextual pointing combined with a regular pointing device for precision tasks within the context set by the eye gaze.

[1] The min decipherable symbol height h given distance d: $h = 2\ d \tan(\Theta/2)$, Θ = 5' of arc for 20/20 vision. h = 5.81mm, or 11.64mm for 20/40 vision [17].

Body and Hand Tracking

Krueger [11] explored using the entire body as an input device to control playful visualizations on a large display, while advances in computer vision bring us closer to robust, real time recognition of our body, hand, and finger positions in 2D and 3D space [19]. Vogel and Balakrishnan [28] use body position for coarse grained 1D pointing to provide context for fine-grained actions, but 2D control is difficult with this approach. Nickel and Stiefelhagen [19] explored computer vision methods to find pointing direction including head to hand line of sight, forearm orientation, and head orientation. They found head orientation to be important for determining pointing direction.

Direct Hand Pointing

Zhai [32] discussed using a hand for 6 DOF tasks and found problems with rotation due to limited mobility of the wrist. Hinckley et al. [9] surveyed different techniques for using hands for spatial input, mostly concentrating on 6 DOF tasks applied to virtual or augmented environments, but they only discuss physical buttons mounted on gloves for selection or clutching. Corradini and Cohen [4] created a free space finger painting system by tracking the position of a finger within a user defined absolute coordinate frame. They use speech together with pointing gestures to issue commands controlling the painting parameters. Bolt's classic "put-that-there" system [1] combines direct pointing using a 6 DOF magnetic tracker with voice commands to disambiguate context.

Virtual Environments

Using the hand to select objects is common in virtual environments (VE). Poupyrev and Ichikawa [26] gave an overview of several hand based object manipulation techniques including finger ray casting, although they use a button for selection. In an experimental evaluation, they found ray casting to be fast if accuracy is less important. An earlier study by Bowman and Hodges [3] found that naturalness is not always a necessary component of an effective technique, for instance although the go-go reaching technique is most natural, users preferred ray casting since it was the least effort. Pierce et al. presented a two-handed Voodoo doll technique to manipulate distant objects [25], and, most relevant to our work, a family of image plane selection techniques [24]. These allow selecting objects in a 3D world by considering the 2D view plane of the user. Some, such as the "Head Crusher," use finger gestures for a selection mechanism.

Selection With the Hand

Most work discussed so far uses a physical button, dwell time, or voice command for selection. Grossman et al. [7] use a thumb trigger gesture for selection in their volumetric display interaction techniques. Their prototype uses a commercial high-precision motion tracking system, which suggests one reason why selection gestures performed with the fingers have been ignored until recently: their movement is simply too subtle to be recognized accurately by other tracking technologies.

POINTING AND CLICKING USING ONLY THE HAND

Touching a screen with the finger is an effective way to interact when up close, so perhaps a bare hand can also be used to point when away from the display. This eliminates the problem of carrying a device, and provides a natural transition from distant to up close touch screen interaction.

Kendon's social anthropology research posits that we use seven different gestures to point when communicating [10]. These pointing ("deictic") gestures are classified by the context of what is being pointed at. Relevant to our work is the gesture to indicate a specific thing, done with the index finger extended and palm facing down. One can imagine a laser beam emanating from the tip of the finger along the vector of the finger's direction, resulting in a pointing technique that it is arguably natural and conceptually simple. Our survey of previous work suggests that this "ray casting" technique is the most obvious for distant pointing. As discussed earlier, this style of pointing is fast for selecting large targets, but prone to errors due to hand jitter. A nice quality of ray casting is that it is consistent with touch screen input – when the finger touches the display, the ray has zero length and it behaves like a touch screen.

If we observe the motion of the hand when using a mouse, another candidate technique emerges. If the mouse is thought to be invisible, the motion of the hand alone can be used for cursor positioning. Since the very large display is a vertical plane, hand motion should also be in a parallel vertical plane (rather than horizontal) to remain consistent with touch screen interaction close to the display.

Clicking and Clutching Without a Button

A classic problem in device-free interaction is how to signal a selection or a clutch in the absence of any buttons. One solution, which has been adopted by many eye tracking systems [8] is to use a cursor dwell time threshold as a click event [31]. Although simple, this introduces a fixed, constant lag, and interactions may suffer from the "Midas Touch effect" [8] in very dense environments. Another approach is to use speech to signal a selection [1] but this is excessive for simply capturing click down and up actions.

We explored different hand gestures and postures to clutch and click. Since the hand is also pointing, the click or clutch action should be designed to minimize hand movement side effects, which can be tricky due to the interconnectedness of tendons and ligaments in the hand. We also felt that the posture or gesture used to deactivate the clutch should have some tension, similar to the natural tension that is required when lifting the mouse to clutch.

When depressing a physical button or tapping a display surface, we receive instant kinesthetic feedback confirming that the click has been triggered. Wang and MacKenzie [29] found that performance degraded significantly when there was no physical surface to touch when manipulating virtual objects with the hand. Thus, with free space hand gestures, we need to investigate other sensory replacements in an attempt to mitigate the effects of lost kinesthetic feedback.

System Protoype

To prototype and explore different techniques we use a Vicon (www.vicon.com) motion tracking system to get accurate and fast position information for the hand. We place passive reflective markers on the thumb, index finger, ring finger, back of the palm, and wrist (Figure 1c). The system can uniquely identify each marker and stream its sub-millimetre 3D coordinates at up to 120 Hz to other applications. While the inconvenience of using markers and a specialized motion tracking system does detract from the implementation simplicity of our prototype, this technology allows us to explore advanced interaction techniques today, *before* marker-free tracking becomes widely available. As such, this hardware should be viewed simply as an enabling technology for our prototype, rather than one that would be used in a future real implementation of our interface ideas. We also use a 5m wide, 1.8m high, back projected display (Figure 1a), with imagery generated by 18 1024x768 resolution projectors in a 6x3 tiling for an effective resolution of 6144x2304 pixels. A cluster of 18 PCs drive the projectors, with Chromium (chromium.sourceforge.net) providing distributed graphics rendering over the cluster. Although this display is not currently enabled for touch input, we also used a 50" touch-enabled plasma display to observe how the techniques transition from up close to distant usage. Our custom software was written in C++ and OpenGL and is capable of easily performing position and gesture recognition at 45 FPS, producing a maximum lag time of only 22 ms between a movement and screen update.

CLICKING TECHNIQUES

We created two clicking techniques, one using the index finger and the other using the thumb. Since our design goal is to create a selection technique wholly compatible with current "point and click" user interfaces, our techniques support click down and click up events – allowing single clicks, double clicks, and drags. We use visual and auditory feedback to replace the lost kinesthetic feedback of a physical button push or display tap [31]. To visually indicate when a click down has occurred, we show a short animated progression of a medium sized square shrinking and disappearing at the position where the event occurred. At the same time, a distinctive clicking sound is played characterized by a waveform envelope with a long attack and short release reminiscent of the "in" sound made by pet training "clickers" (Figure 2b). In a similar way, a click up is visualized by a small square expanding from the event point and a slightly different clicking sound with a slightly higher pitch, short attack, and long release (Figure 2d). If a click up event is not registered within 1000 ms, a small square is shown attached to the tail of the cursor arrow (Figure 2c) indicating a prolonged click down state used for dragging. This not only acts as a visual replacement for the kinesthetic tension normally experienced when holding a button down, but we also found that this aided new users in learning the tolerances of the click up style.

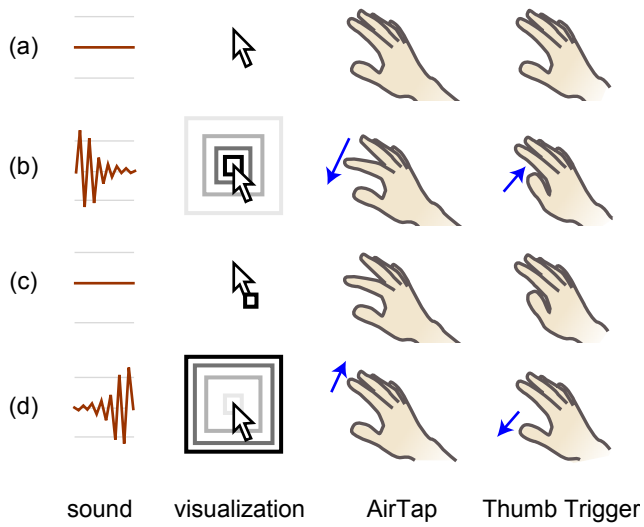

Figure 2: Clicking Techniques, Visualizations, and Sound. AirTap is a "down and up" gesture of the index finger. ThumbTrigger is an "in and out" gesture of the thumb. Sound and visualization are used to make the clicking feel more physical in the absence of a surface or button: (a) default state (no clicking); (b) click down gesture displays an animated series of rectangles scaling down towards the click point and plays a distinctive click down sound; (c) holding the finger or thumb in a click down state displays a small square on the tail of the pointer; (c) click up gesture displays an animated series of rectangles scaling out from the click point and plays a click up sound.

AirTap

The AirTap click technique is similar to how we move our index finger when clicking a mouse button or tapping a touch screen. We found two main challenges when designing this technique. The first challenge is that there is no physical object to constrain the downward movement of the finger to a definite start or stop position. To deal with this, our click down recognition algorithm uses relative features of the downward finger motion, specifically velocity and acceleration, in addition to absolute position and movement axis. The second challenge is the ambiguity and idiosyncrasy of this style of finger movement. Other hand gestures and even involuntary finger movement tend to resemble this type of clicking action, and individuals tend to move their finger in distinctive, but different, ways. We adopted a simple calibration scheme that tuned the recognition parameters to a particular individual's clicking style, which narrowed the space of recognized clicks and reduced false positives in click recognition. To calibrate, we record 5 seconds of index finger movement as the user clicks. From this data we find the principle movement axis, m, by fitting a Gaussian to the finger positions to get the dominant Eigenvector. We find a threshold position, P_{up}, to trigger a click up, and threshold distances moved in 200ms, D_{down} and D_{cancel}, to begin a click down or cancel respectively. We also use a fixed pause velocity threshold, v_{pause} of 80mm/s.

Let the current finger position be P with velocity v and distance traveled in 200ms be D (v and D are both measured along the principle movement axis m). Figure 3 shows the state machine for this recognition algorithm.

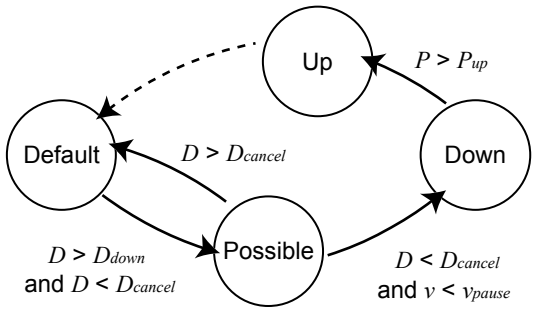

Figure 3. Recognition state machine. The current finger position is P with velocity v and distance traveled in 200ms, D. v and D are both measured along the principle movement axis m.

ThumbTrigger

Similar to Grossman et al. [7], we implemented a thumb trigger style click where the thumb is moved in and out towards the index finger side of the hand (Figure 2). Theoretically, this click style has a distinct advantage over the AirTap since the thumb can touch the side of hand and provide kinesthetic feedback when clicking, and also provides an absolute down position. Based on this, our first implementation used only distance thresholds; once the thumb moved past a certain point on the way towards the palm, a click down was triggered, and likewise when travelling away during a click up. However, early tests revealed that users found "clicking" their thumb against the side of their hand uncomfortable and tiring. Therefore, we adopted a recognition algorithm which uses a blend of relative features and absolute features, similar to AirTap.

Adjusting for Intended Click Point

When performing either of these click gestures, the interconnected nature of the hand's physiology causes some involuntary finger and hand movement. To combat this, we adjust the position of the click down or up event to be the intended position, taken to be the point where the cursor was pointing when the click gesture began.

POINTING TECHNQIUES

We designed three pointing techniques: absolute position finger ray casting, relative pointing with clutching, and a hybrid technique using ray casting for quick absolute coarse pointing combined with relative pointing when more precision is desired. These techniques depend on simple hand postures to signify a clutch or ray cast point which we will discus now.

Detecting and Teaching Hand Postures

We detect whether individual tracked fingers are pointing out or curled into the palm based on threshold distances between the current finger marker position and the closed or open positions. For example, open and closed hand positions vary between individuals, so we calibrate through a simple 10 second process. Asymmetrical thresholds are used to move the finger to an open or closed state; this eliminates "thrashing" when near an ambiguous position. To encourage users to adopt definite hand postures, we created an *ambiguous posture visualization* (Figure 4, Figure 1b). A red circle appears and becomes more intense when the posture nears an ambiguous position. We found this to be an effective teaching signal, and it appeared to be effective in training users to adopt clean postures.

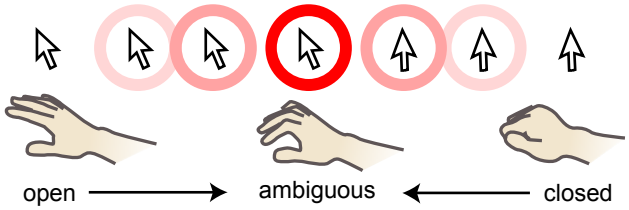

Figure 4. Ambiguous Posture Visualization. Increasing intensity of a red disk indicates an ambiguous posture.

RayCasting

Drawing from previous work in interaction design and bolstered by social anthropological research on human pointing [10] we implemented a finger ray casting pointing technique. The cursor is placed at the point where a ray emanating from the index finger intersects with the display (Figure 5). Note that using the index finger in this way rules out combining the AirTap clicking gesture with Ray Casting. As we discussed earlier, there is a known problem of "jittery" cursor movement with laser pointers due to natural hand tremors, and this is exacerbated by the use of the finger to define the ray. We experimented with using the inner surface of the palm to define the ray, but, although this reduced the jittery nature somewhat, it is not how we actually point in the real world [10] and, more importantly, forced more extreme movements of the forearm, increasing fatigue. Researchers have explored various filters to steady the cursor in the case of laser pointing like Kalman filtering [20], a two stage mean filter based on angular velocity [31] and more elaborate models [14, 15]. Initially, we used a simple recursive low pass filter on the 2D cursor position, but results from our pilot evaluation led us to improve on it. We designed a *dynamic* recursive low pass filter for 3D marker positions that define the ray. This interpolates between cutoffs of .25Hz and 5Hz (with a 90Hz sample rate) based on marker velocity v. If $v < 10$mm/s then the low cutoff is used, if $v > 200$ mm/s then we use the high cutoff, and for values in between we use a linear interpolation of cutoff based on v. We found this effectively removed jitter when the pointer was still, yet introduced almost no lag, which can be detrimental to performance [30].

Relative Pointing with Clutching

In this technique we use the motion of the hand projected on a vertical plane for cursor positioning (Figure 6a). We first considered using an *absolute* mapping. By calibrating the scale of the hand's vertical coordinate frame to a comfortable range of movement, we found that cursor control was surprisingly accurate and fast. However, a problem emerges as the user moves: our design constraint of tracking only a single hand prevented us from translating the coordinate frame according to body movement.

When using whole hand spatial input, Hinckley et al. [9] emphasize the importance of using relative rather than absolute mappings. Mapping absolute hand positions directly to a parameter can be too abstract and they suggest moving relative to a physical prop instead. Since our design goals preclude using props, we adopted a relative technique where an absolute start position is chosen on a vertical plane using a clutching mechanism and subsequent left-right and up-down motion is relative to that. Hinckley et al. refer to this as a *ratcheting recalibration mechanism* [9].

We use the neutral "safe hand" posture for pointing (Figure 6a) and a clenched fist ("Grip Clutch") to disengage the hand from the pointer and recalibrate the position (Figure 6b). We experimented with tense postures for pointing but found that clicking with the fingers or thumb is difficult if the hand is under tension. Also, since pointing actions are typically of longer duration than clutching actions, it makes sense to use a tense posture for clutching rather than pointing. To inform the user that the cursor has been disengaged from hand movement, we animate the standard arrow pointer icon to rotate so it appears to dangle from the arrow point. When the clutch is deactivated, the pointer rotates back to its customary angle. This subtle visualization suggests that when the hand is connected to the cursor the pointer is "held up by the hand" and when disengaged it swings down to a rest position. The ambiguous posture visualization (Figure 4) helps the user adopt clear clutch or no clutch poses.

To increase the range of cursor movement, we implemented the same variable control-display (CD) gain function used for pointer control in Windows XP. This adjusts the CD gain according to a non-linear function of velocity [16]. Once the hand and pointer are reasonably calibrated through clutching, we found this function worked well. We selected a scale factor of 0.7 which made it easy to traverse thousands of pixels from one side of the display to the other, yet did not introduce any loss in accuracy when selecting small targets.

Hybrid RayToRelative Pointing

Our third technique uses ray casting as a way to recalibrate the hand position while simultaneously repositioning the cursor near the desired target. This eliminates the cognitive load of the Grip Clutch's backwards ratcheting movement, and takes advantage of ray casting's ability to do rapid coarse grain pointing. Direct cursor control is accomplished using the same relative hand movement technique (Figure 7a) discussed in the previous section but when the hand pose changes to a finger point, the cursor is replaced with a circle positioned on the display where the ray emanating from the finger intersects with the display (Figure 7b). The circle can be rapidly repositioned with very little hand movement and the cursor positioned at the centre when the hand returns to the neutral safe hand position. Returning to the neutral posture causes the index finger tendons to contract and jerk the circle upwards, positioning the cursor too high. To eliminate this, we adjust the cursor position to be the intended position, taken to be the point where the return to open hand is initiated.

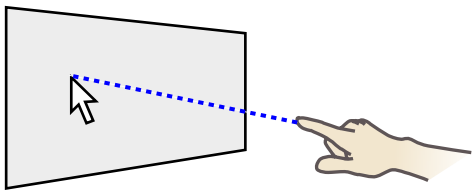

Figure 5. RayCasting. A ray extends from the tip of the finger and the cursor is positioned where it intersects with the large display surface.

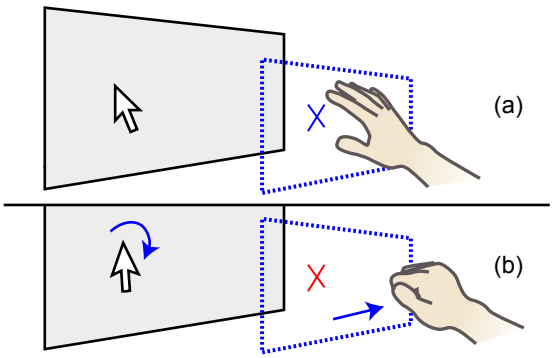

Figure 6. Relative Pointing with Clutching. (a) The open hand is used for relative cursor control, and (b) a clenched fist ("Grip Clutch") is used for clutching. When the clutch is engaged, the cursor arrow swings to a dangling position.

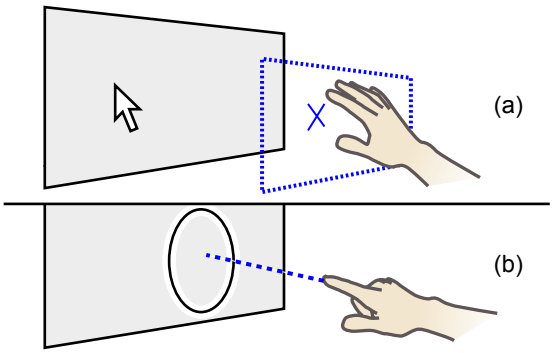

Figure 7 Hybrid RayToRelative Pointing. (a) The open hand is used for relative cursor control, and (b) recalibrating (or clutching) is performed with an absolute ray cast pointing gesture. When ray cast pointing, the cursor transforms to a large circle to suggest the selection of an approximate area.

PILOT EVALUATION

The purpose of our initial evaluation was to compare and refine the ThumbTrigger and AirTap click gestures when used with each of our three pointing techniques (except for the AirTap and RayCasting combination for obvious reasons). We used a Fitts' [5, 13] style task requiring sequential clicks on different sized circular targets (10mm, 30mm, 90mm) with varying distances (3200mm, 1600mm, 800mm). We did not see a difference in trial performance time or error rate between the two different click gestures. Since AirTap is more consistent with touch screen interaction, and since the ThumbTrigger's theoretical advantage of kinesthetic feedback did not materialize in these exploratory tests, we elected to use AirTap with the relative pointing technique in further studies. We found surprisingly high error rates for the RayCasting technique in

the small and medium target sizes varying from 20% to 80%, more than twice that of the other techniques. In addition, users found RayCasting to be excruciatingly difficult and tiring due to the extra effort when attempting to select small targets with poor pointing precision. This prompted us to improve the RayCasting pointer position filter from a simple low-pass filter to a dynamic low-pass filter as discussed previously.

EXPERIMENTAL EVALUATION

Goals

Our goal is to compare task completion time, error rate, recalibration activations, recalibration frequency and total recalibration time between the three pointing techniques: RayCasting, Relative, and RayToRelative. We expect the RayCasting technique to be the fastest since it requires no recalibration, but will have a higher error rate with small targets. Theoretically, the RayToRelative technique has a time advantage over the Relative technique since the recalibration mechanism simultaneously moves the cursor closer to the target. However, the overhead of switching between ray casting and relative movement, and the implicit target acquisition task during recalibration (i.e., the ray casting action must coarsely point to the approximate region of interest) may introduce too much delay.

Participants

Twelve participants, 4 women and 8 men ranging in age from 20 to 36 years, participated. None of the participants had experience with pointing tasks on large displays.

Apparatus

The experiment used the Vicon motion tracking system with passive markers attached to the hand, and the very large high resolution display as discussed previously. The participants stood at a stationary, central position 4 m away from the display.

Task and Stimuli

From our pilot evaluation, we found that participants were willing to move their hand to the extreme extents of their range of motion, adopting awkward poses to avoid the time penalty associated with recalibration. This behaviour is not characteristic of a real usage scenario where users would adopt a more energy conserving posture, balancing speed and effort. To encourage a more relaxed posture, we artificially constrained participant's movement by introducing boundaries where hand tracking appears to fail. We revealed proximity to a boundary with a blue disk surrounding the cursor which fades in to full opacity and displays an 'X' when the boundary is reached.

In the experiment, each set of trials begins with the cursor and first target hidden until the user holds their hand in a experimentally controlled start location for 2 seconds. This simulates the transition from some arbitrary hand-based task to a pointing task. The cursor appears at an experimentally manipulated distance from a circular white target. To select the target, the participant may have to clutch and recalibrate to control the cursor in a comfortable manner. We refer to this first trial task of starting the cursor movement and selecting the first target as the *Transition Task*.

After the first target is selected successfully, the next target within a series of three targets appears. The participant must successfully select this target before the next appears, and so on. We call this the *Sequence Task*, which differs from the *Transition Task* in that the user is already controlling the cursor with the given technique from the time the target appears. This simulates the real interface situation where users might want to make several selections in a row before relaxing their hand and cursor. We also chose a sequence design to ensure that participants use the techniques in an ecologically sound manner, optimizing for both speed and comfort. In contrast, if we had used a single target per trial, participants could have optimized for speed while ignoring momentary discomfort for the short duration of each single task. The net result is that our design ensures to the extent possible that participants recalibrate their hand to cursor relationship when required, simulating real usage scenarios.

Design

A repeated measures within-participant factorial design was used. The independent variables were *Technique* (*Relative*, *RayToRelative*, and *RayCasting*), distance between targets D (D_L = 4020mm, D_M = 2680mm, D_S = 1340mm) and target width W (W_L = 144mm, W_M = 48mm, W_S = 16mm), where the subscripts $_L$, $_M$, and $_S$ are used to denote *large*, *medium*, and *small* respectively. The Fitts' index of difficulty (ID) of our selection tasks range from 3.37 to 7.98 bits.h

Presentation of the three techniques to the 12 participants was fully counter-balanced, resulting in 6 different presentation order groups. For each technique, participants had a 5 minute learning session and 1 block of practice trials. Then participants were asked to perform 3 blocks of recorded trials. For each block of trials, participants performed 6 *series* for each of the 3 W conditions. A *series* consisted of one selection in the *Transition Task* followed by 3 selections in the *Sequence Task*. Target distances within the 6 series were presented randomly, with each of D_L, D_M, and D_S appearing an equal number of times across the 6 series. W was presented in a Latin square ordering across the three blocks. Participants had to successfully select each target before the next target would appear, ensuring that they did not *race through* the experiment by clicking anywhere just to finish quickly. Participants were allowed breaks between series.

In summary, the experimental design was:

12 participants x

3 techniques x

3 blocks x

3 target widths x

6 sets of *Transition Task* (1 target) followed by *Sequence Task* (3 targets)

= 1944 series (1944 targets selected in *Transition Task,* and 5832 targets selected in *Sequence Task*)

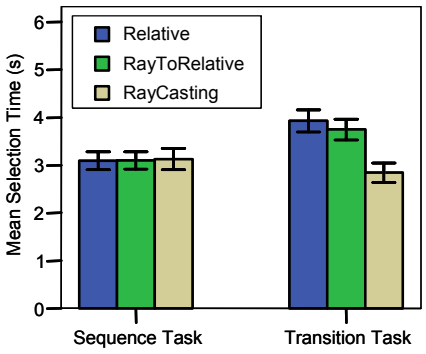

Figure 8. Mean selection times for *Sequence Task* and *Transition Task*.

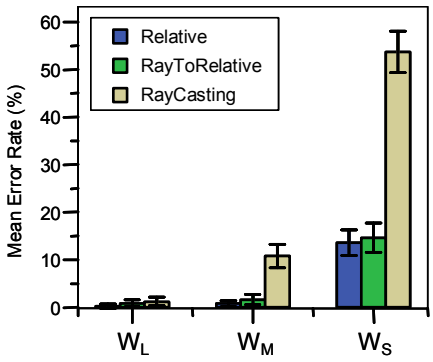

Figure 9. Mean error rate by width for *Sequence Task*.

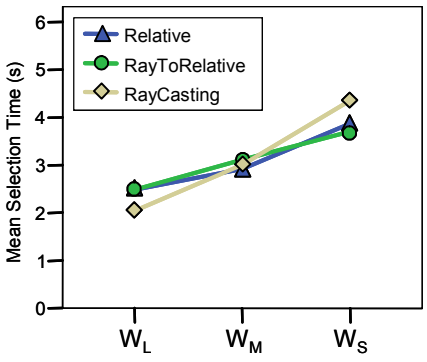

Figure 10. Technique and width interaction on selection time for *Sequence Task*.

Selection Time and Error Analysis

Results for Sequence Task

Selection time for the *Sequence Task* was defined as the time it took to move from the previous target and successfully select the next target. Targets that were not selected on the first attempt were marked as errors, and not included in the timing analysis. Repeated measures analysis of variance showed that the order of presentation of the three techniques had no significant effect on selection time or error rate, indicating that a within-participants design is appropriate.

There was no significant main effect for technique on selection time, with means of 3088, 3097, and 3128 ms for the *Relative*, *RayToRelative*, and *RayCasting* techniques respectively (Figure 8). Since our analysis below found minimal clutching in the *Sequence task*, this indicates that the relative hand movement technique used by both *Relative* and *RayToRelative* is equivalent in speed to the absolute *RayCasting* pointing technique.

There was a significant interaction between technique and target width ($F_{4,24} = 33.945$, $p < .001$). Post hoc multiple means comparison tests showed that *RayCasting* was significantly slower than *Relative* and *RayToRelative* for W_S (by 485 and 654 ms respectively, both $p < .05$) but faster for W_L (by 447 and 449 ms respectively, both $p < .05$) (Figure 10). This suggests that although there is an overhead in the relative hand movement techniques, the small controlled movements required for selecting small targets are faster for the relative hand movement techniques when compared to *RayCasting*. With *RayCasting,* the small corrective movements required by small targets become more difficult and time consuming. There was no significant interaction found between technique and target distance.

There was a significant effect for technique on selection error rate ($F_{2,12} = 109.212$, $p < 0.001$). Multiple means comparison tests found *RayCasting* to be significantly more error prone with a mean error rate of 22.5% compared to 3.5% and 5.7% for *Relative* and *RayToRelative* respectively Note that error rates for *Relative* and *RayToRelative* are within the typical range for Fitts' pointing tasks. Not surprisingly, a significant interaction was found between technique and width ($F_{4,24} = 122.413$, $p < 0.001$). Multiple means comparison tests showed *RayCasting* had significantly higher errors rates of 56% for W_S and 10.5% for W_M (both $p < .05$). In comparison, the error rates for *Relative* was 9.6% and 1.1% and *RayToRelative* was 15.4% and .9% for W_S and W_M respectively. W_L error rates were less than 1.1% for all techniques (Figure 9).

Results for Transition Task

Selection time for the *Transition Task* was the length of time to successfully click on the first target after the cursor appeared. Like the *Sequence Task*, selection errors were not included in the timing analysis. There were no significant technique order effects for selection time or error rate.

There was a significant main effect for technique on selection time ($F_{2,12} = 20.193$, $p < .001$). *RayCasting*, with a mean time of 2843ms, was significantly faster than *Relative* and *RayToRelative*, with mean times of 3926 and 3744ms respectively (Figure 8). Since there was no significant difference in mean selection time for the *Sequence Task*, this indicates that recalibration time is a significant factor in pointing performance.

There was a significant interaction between technique and distance on selection time ($F_{4,24} = 14.692$, $p < 0.001$). Multiple means comparison tests found *RayCasting* to be significantly faster than *Relative* and *RayToRelative* for D_L and D_M (all $p < .01$), but not D_S. The lower frequency of recalibrations for D_L prevents the recalibration overhead time from slowing the two relative positioning techniques significantly. This reaffirms our *Sequence Task* results: the relative techniques are equivalent in speed to *RayCasting* when there is no recalibration. The remaining results for time and error revealed similar trends to the *Sequence Task.*

Recalibration Frequency Analysis

We designed the *Transition Task* to simulate a situation with a high likelihood of a recalibration step with the *Relative* and *RayToRelative* techniques (*RayCasting* requires no recalibration). Indeed, we found that a recalibration step was used in the *Transition Task* 63.1% of the time with *Relative* and 63.9% with *RayToRelative*; this difference in recalibration frequency was not significantly different.There was a significant main effect for target distance on recalibration frequency ($F_{2,12} = 8.357$, $p < .001$) with means of 91.3%, 73.9%, and 25.5% for D_L, D_M, and D_S. The longer the distance the more often users had to recalibrate, clearly indicating that recalibration is an important factor on large displays. In the *Sequence Task* participants recalibrated only

35.8% and 31.5% with *Relative* and *RayToRelative* respectively; no significant difference was found between techniques. Once reasonable calibration has been achieved, subsequent selection tasks require fewer calibration steps.

Recalibration Time and Activation Analysis

We examined metrics related to how the *Relative* and *RayToRelative* recalibration mechanisms were used. Our analysis included only trials in which a recalibration occurred across *Transition* and *Selection* tasks including only distances D_L and D_M (since D_S had a low frequency of recalibration). Recalibration time is the time spent recalibrating in a single trial, by clutching with the *Relative* technique, or by switching to and from ray casting with *RayToRelative*. Recalibration activations are the number of times a user recalibrated in a single trial. There were no significant effects for technique presentation order on recalibration time or activations indicating that a within-subjects analysis is appropriate.

There was a significant main effect for technique on recalibration time ($F_{1,6} = 7.495$, $p = .034$), with means of 901 and 1536 ms for *Relative* and *RayToRelative* respectively. Recall that we found no significant difference in overall selection time between these techniques. The difference in recalibration time between the two techniques is interesting, as it clearly indicates that the overhead of switching between ray casting and relative pointing with *RayToRelative*, despite the smooth transition design, is higher than any advantage gained in bringing the cursor closer to the target in the ray cast action. In contrast, the clutching in the *Relative* technique only serves to reset the position of the cursor relative to the hand and does not move the cursor closer to the target, but the results clearly indicate that this is not a major drawback to this technique.

There was a significant main effect for technique on recalibration activations ($F_{1,6} = 16.680$, $p = .006$), with means of 1.133 and 1.027 for *Relative* and *RayToRelative* respectively. Post hoc multiple means comparison tests of distance on technique found a significant difference in *RayToRelative* recalibration activations of 1.053 for D_L and 1.000 for D_M, but no significant difference for *Relative*. Considering also the lower recalibration time, this reinforced our observations that with the *Relative* technique users performed recalibration more frequently in fast, short, consecutive strides independent of distance. However with the *RayToRelative* technique, users recalibrated their position less often, especially with medium distances, indicating that the coarse positioning aspect of the technique streamlined the interaction somewhat.

User Feedback

At the conclusion the experiment, we asked each participant to rank the techniques for speed, accuracy, and ease-of-use. 8 felt *Relative* was fastest and 7 found it the most accurate with *RayToRelative* making up the balance. For ease-of-use, 6 selected *Relative*, 5 *RayToRelative*, and 1 *RayCasting*. We found most participants liked *Relative* because the clutching action was similar to that of lifting a mouse to clutch.

DISCUSSION and CONCLUSIONS

We found that although *RayCasting* was faster in tasks where clutching would have been required or when selecting large targets, its high error rates prevent it from being a practical technique. We found no major significant effect between the *Relative* and *RayToRelative* techniques in terms of selection time or error rate. The relatively low error rates for these two techniques is reassuring, indicating that they are equally usable for selection of small (16mm) targets while standing 4m away from the display.

The two relative hand pointing techniques differed with regards to the number of recalibration activations and recalibration time. Interestingly, the longer recalibration times for the *RayToRelative* technique did not impact its overall selection time as compared to the *Relative* technique, indicating that the time overhead due to the ray casting portion of the technique was compensated for by a reduction in subsequent relative movement.

We note that our techniques currently only support actions equivalent to those of a single button mouse or touch screen. It would be interesting to explore using the thumb and index finger together to "left" and "right" click.

Our design of the ambiguous clutch visualization was observed qualitatively to be an effective "teaching input" that helped users adopt more definite postures. This idea could be applied to click gesture recognition. When a click-like finger movement just missed being classified as a click gesture, visual feedback could be shown with a suggestion how to adjust finger movement so it is classified as a click the next time (make it faster, slower, longer, etc).

It would be interesting to consider the other six pointing gestures from Kendon's study [10], as well as additional body movements, in the design of future pointing techniques. For example, using a second hand or eye gaze or head position to accelerate pointing by selecting a region of the display within which the dominant hand can perform fine grained pointing and clicking.

In summary, this research has made several contributions: we have motivated the need for facile pointing and clicking techniques for interacting with large displays from a distance; identified desirable characteristics for such techniques; and developed and evaluated new pointing and clutching techniques that leverage the simplicity and inherent human ability to point with a hand. Of particular interest are the subtle nuances in the design of our techniques. We use visual and auditory feedback in our clicking techniques to compensate for the lack of kinesthetic feedback typically present when clicking a physical button; provide unobtrusive but yet effective visualizations to subtly alert the user when postures and gestures are about to become ambiguous; and used various heuristics to tune the parameters of the clicking mechanisms such that they behave as users implicitly expect. Finally, our evaluations demonstrate the usability of relative hand base pointing techniques with error rates in the same low range one typically sees with status-quo devices like mice.

ACKNOWLEDGEMENTS

We thank John Hancock, Anastasia Bezerianos, Géry Casiez, and our experiment participants.

REFERENCES

1. Bolt, R. (1980). Put-that-there: Voice and gesture at the graphics interface. *Computer Graphics*, 14(3). p. 262-270.
2. Bolt, R. (1981). Gaze-orchestrated dynamic windows. *Computer Graphics*, 15(3). p. 109-119.
3. Bowman, D. and Hodges, L. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. *ACM Symposium on Interactive 3D Graphics*. p. 35-38.
4. Corradini, A. and Cohen, P. (2002). Multimodal speech-gesture interface for hands-free painting on virtual paper using partial recurrent neural networks for gesture recognition. *International Joint Conference on Neural Networks*. p. 2293-2298.
5. Fitts, P.M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47. p. 381-391.
6. Fono, D. and Vertegaal, R. (2005). EyeWindows: Evaluation of eye-controlled zooming windows for focus selection. *ACM CHI Conference*. p. 151-160.
7. Grossman, T., Wigdor, D., and Balakrishnan, R. (2004). Multi finger gestural interaction with 3D volumetric displays. *ACM UIST Symposium*. p. 61-70.
8. Hansen, J., Andersen, A., and Roed, P. (1995). Eye-gaze control of multimedia systems. *ACM Symposium on Eye Tracking Research & Applications*. p. 115-122.
9. Hinckley, K., Pausch, R., Goble, J.C., and Kassell, N. (1994). A survey of design issues in spatial input. *ACM UIST Symposium*. p. 213-222.
10. Kendon, A. (2004). Gesture: visible action as utterance. 2004: Cambridge University Press.
11. Krueger, M. (1991). *VIDEOPLACE and the interface of the future*, in *The art of human computer interface design*, B. Laurel, Editor. Addison Wesley. p. 417-422.
12. MacKenzie, I. and Jusoh, D. (2001). An evaluation of two input devices for remote pointing. *Eighth IFIP Working Conference on Engineering for Human-Computer Interaction*. p. 235-249.
13. MacKenzie, S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7. p. 91-139.
14. Matveyev, S. and Göbel, M. (2003). The Optical Tweezers: multiple-point interaction technique. *Virtual Reality Software and Technology*. p. 184-188.
15. Matveyev, S., Göbel, M., and Frolov, P. (2003). Laser pointer interaction with hand tremor elimination. *HCI International*. p. 376-740.
16. Microsoft. (2005). Pointer ballistics for Windows XP. Accessed on 14 Feb 2005, www.microsoft.com /whdc/device/input/pointer-bal.mspx.
17. Millodot, M. (1997). *Dictionary of Optometry and Visual Science*. Butterworth-Heinemann. p. 8, 44.
18. Myers, B., Bhatnagar, R., Nichols, J., Peck, C.H., Kong, D., Miller, R., and Long, C. (2002). Interacting at a distance: measuring the performance of laser pointers and other devices. *ACM CHI Conference*. p. 33-40.
19. Nickel, K. and Stiefelhagen, R. (2003). Pointing gesture recognition based on 3D-tracking of face, hands and head orientation. *International Conference on Multimodal Interfaces*. p. 140-146.
20. Oh, J.-Y. and Stuerzlinger, W. (2002). Laser pointers as collaborative pointing devices. *Graphics Interface*. p. 141-149.
21. Olsen, D.R. and Nielsen, T. (2001). Laser pointer interaction. *ACM CHI Conference*. p. 17-22.
22. Parker, J.K., Mandryk, R.L., and Inkpen, K.M. (2005). TractorBeam: Seamless integration of remote and local pointing for tabletop displays. *Graphics Interface*. p. 33-40.
23. Peck, C. (2001). Useful parameters for the design of laser pointer interaction techniques. *Extended Abstracts of the ACM CHI Conference*. p. 461-462.
24. Pierce, J., Forsberg, A., Conway, M., Hong, S., and Zeleznik, R. (1997). Image plane interaction techniques in 3D immersive environments. *ACM Symposium on Interactive 3D Graphics*. p. 39-43.
25. Pierce, J., Stearns, B., and Pausch, R. (1999). Two handed manipulation of voodoo dolls in virtual environments. *ACM Symposium on Interactive 3D Graphics*. p. 141-145.
26. Poupyrev, I. and Ichikawa, T. (1999). Manipulating objects in virtual worlds: categorization and empirical evaluation of interaction techniques. *Journal of Visual Languages and Computing*, 10. p. 19-35.
27. Skaburskis, A., Vertegaal, R., and Shell, J. (2004). Auramirror: Reflections on attention. *ACM Symposium on Eye Tracking Research & Applications*. p. 101-108.
28. Vogel, D. and Balakrishnan, R. (2004). Interactive public ambient displays: Transitioning from implicit to explicit, public to personal, interaction with multiple users. *ACM UIST Symposium*. p. 137-146.
29. Wang, Y. and MacKenzie, C. (2000). The role of contextual haptic and visual constraints on object manipulation in virtual environments. *ACM CHI Conference*. p. 532-539.
30. Ware, C. and Balakrishnan, R. (1994). Reaching for objects in VR displays: Lag and frame rate. *ACM Tranactions on Computer-Human Interaction*, 1(4). p. 331-356.
31. Wilson, A. and Pham, H. (2003). Pointing in intelligent environments with the World Cursor. *INTERACT 2003*.
32. Zhai, S. (1998). User performance in relation to 3D input device design. *Computer Graphics*, 32(4). p. 50-54.
33. Zhai, S., Morimoto, C., and Ihde, S. (1999). Manual and gaze input cascaded (MAGIC) pointing. *ACM CHI Conference*. p. 246-253.