LOP-cursor: Fast and precise interaction with tiled displays using one hand and levels of precision

Conference Paper · March 2012

DOI: 10.1109/3DUI.2012.6184196

3 authors:

Henrique Galvan Debarba
École Polytechnique Fédérale de Lausanne
14 PUBLICATIONS **22** CITATIONS
SEE PROFILE

Luciana Nedel
Universidade Federal do Rio Grande do Sul
158 PUBLICATIONS **856** CITATIONS
SEE PROFILE

Anderson Maciel
Universidade Federal do Rio Grande do Sul
75 PUBLICATIONS **355** CITATIONS
SEE PROFILE

LOP-cursor: Fast and Precise Interaction with Tiled Displays Using One Hand and Levels of Precision

Henrique Debarba* Luciana Nedel† Anderson Maciel‡

Instituto de Informatica (INF)
Universidade Federal do Rio Grande do Sul (UFRGS)

Abstract

We present levels of precision (LOP) cursor, a metaphor for high precision pointing and simultaneous cursor controlling using commodity mobile devices. The LOP-cursor uses a two levels of precision representation that can be combined to access low and high resolution of input. It provides a constrained area of high resolution input and a broader area of lower input resolution, offering the possibility of working with a two legs cursor using only one hand. LOP-cursor is designed for interaction with large high resolution displays, e.g. display walls, and distributed screens/computers scenarios. This paper presents the design of the cursor, the implementation of a prototype, and user evaluation experiments showing that our method allows both, the acquisition of small targets, and fast interaction while using simultaneous cursors in a comfortable manner. Targets smaller than 0.3 cm can be selected by users at distances over 1.5 m from the screen with minimum effort.

Index Terms: H.5.2 [User Interfaces]: Input devices and strategies (e.g., mouse, touchscreen)—

1 Introduction

With the increasing availability of larger displays – both in size and resolution – desktop conventional interaction is no longer an efficient option in a variety of use cases [15, 20]. Such larger displays are giving rise to new working scenarios in which users are not sitting in front of the screen nor they have a table upon which to pose their mice. A widespread example of this situation are the multi-display screens in operation centers, public spaces and scientific facilities. While they offer great visualization possibilities, efficient interaction with such displays is still a major challenge.

The literature is rich in direct pointing based techniques, many of which do not necessarily rely on conventional mouse-and-keyboard input (next section comments on a few examples). Although those usually offer fast and intuitive input for pointing tasks while interacting with large and distant displays, precision of pointing is limited due to hand jittering. Moreover, most interaction approaches used today in this context do not offer the same resolution for interaction input as the resolution offered by the displays. This problem is related to input device limitations but also to the cursor metaphor in use.

On the other hand, small multi-use devices as cell phones and, more generally, smartphones with sophisticated high-resolution small screens are more ubiquitous than specific devices. Those mobile devices are complete devices, with a number of sensors (GPS, accelerometers, gyroscope, magnetometer, multi-touch screens, cameras, etc.) and networking capabilities (3G, wi-fi, bluetooth).

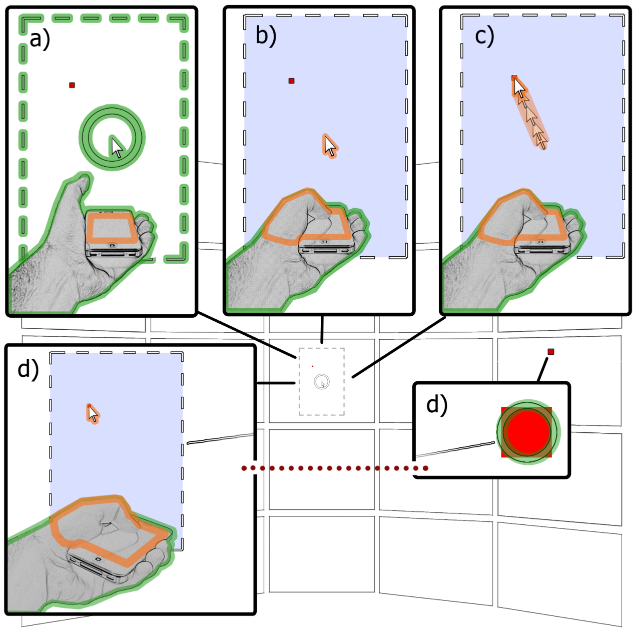

Figure 1: LOP-cursor usage to select and move objects on a tiled display: (a) the user hovers the target neighborhoods with an arrow pointer performing a 3D gesture with a mobile device in the hand; (b) touching the device touchscreen locks the position of a rectangular control canvas; (c) fine tune and precise selecting the target object with the arrow in the control canvas is achieved moving the thumb on touchscreen; (d) holding the control canvas locked, the user can move the arrow cursor by touchscreen on the mobile device while moving their hand to point at a second place with a secondary cursor (ring cursor) to define a destination target to the selected object. Green marks indicate actions made using gestures, while orange marks point out actions done on the touchscreen. The image background illustrate the LOP-cursor graphic representation on a tiled display.

Different combinations of sensor sets enable a number of positioning and orientation possibilities for interaction with external facilities by means of personal mobile devices. While literature covers quite well the use of camera and touch based mobile device interaction techniques [2, 8, 4, 5, 9, 17, 13], sensors of movement and their possible combinations are less explored [19, 10, 7].

In this paper we introduce the LOP-cursor, a multilevel and two-legged cursor for interaction with large high-resolution displays based on a combination of 3D and 2D interaction metaphors. By freely walking in front of the display area, the user changes their position and distance to the screen, emulating pan and scale of the visualization in relation to himself.

LOP is a cursor metaphor in which the accurate cursor positioning is based on two levels of precision. The two levels are necessary

*e-mail: hgdebarba@inf.ufrgs.br
†e-mail:nedel@inf.ufrgs.br
‡e-mail:amaciel@inf.ufrgs.br

to address the lack of precision observed in many single level direct pointing techniques. With LOP-cursor, the user first points the device towards the target on the screen using the laser pointing 3D metaphor. If higher precision is needed, the user can fine tune the cursor position by sliding a finger on the device's touchscreen, at this point using a 2D interaction technique. Figure 1(a to c) illustrates a walkthrough on positioning the cursor as a combination of the two levels.

The fine tuning by touchscreen freezes a control canvas through all the duration of the touch, but does not stop the direct 3D pointing feature, which results in two simultaneous cursors under the control of the user. This design allows two simultaneous pointing locations (the two legs of the cursor) to quickly perform composed tasks as, for instance, selection and move. Figure 1(a, b and d) presents the walkthrough on the simultaneous use of both legs of the cursor. Ninja Cursors [11] is the only other approach that allows simultaneous control over multiple cursors. However, it only replicates the mouse movement across cursors aiming to reduce the index of difficulty of selection tasks, while we aim at allowing simultaneous pointing at two specific places at the same time.

The contributions of this paper are twofold:

- An approach for high resolution interaction with large high resolution displays based on two levels of precision using 3D and 2D interaction metaphors

- A two-legged cursor concept defining two simultaneous positions using only one hand

The remaining of the paper is organized as follows. Section 2 summarizes related works on the use of mobile devices to interact with large displays as well as input techniques that consider levels of precision. Section 3 presents the decisions and details of the LOP-cursor design. Section 4 describes the current state of hardware technology and software implementation. In Section 5 we present the evaluation of the LOP-cursor comparing it against other selection techniques. Finally, in Section 6 we present and discuss the results achieved and in Section 7 we highlight our findings and suggest future developments.

2 Related Work

2.1 Mobile device control of large displays

Ballagas assumed that the smartphone is the first really pervasive computational device [1], being an essential part of contemporaneous life and an always on pocket device. Therefore, its use as an input/output device is quite obvious. Most relevant works using mobile devices to control external large displays are based on optical analysis, usually relying on optical flow [2, 8], pattern recognition [2, 4, 5, 8, 9, 17], and external optical tracking of the device [16]. Other works also use mobile devices sensors on less specific tasks [19, 10, 7].

Ballagas et al. are precursors on mobile phone interaction with large displays, introducing the *Sweep* and *Point and Shoot* techniques [2]. Sweep uses the optical flow to move a cursor on the screen, with a central button as a clutching activator. Point and Shoot uses a quick blink of bi-dimensional tags on the controlling screen synchronized with the camera capture order, allowing reconstruction of camera pointing center. Jiang et al. [9] use the two last on screen cursor positions to define a coordinate system, allowing the cell phone to calculate the new position the cursor should assume.

Pears and Olivier introduced a technique for registration of mobile phone and external displays using four square markers [17]. Registering allows direct mapping of every pixel at the large display on the mobile phone screen, and thus, direct control is provided. Boring et al. developed the Touch Projector, extending interaction with video to mobile devices [4]. Touch Projector uses a polygon comparing algorithm to identify the screen where the user is aiming, thus allowing the control over multiple and spread displays. Touch Projector also suggested improvements to the video interaction concept, based on mobile device specific constraints and needs. Later, Boring et al. extended Touch Projector to interact with media facades [5]. LightSense [16] uses a mobile phone with a back LED over a semitransparent table and track its two dimensional position using an external optical tracking system. The LED diffusion over the table is also used to distinguish across ten levels of distance between table and device.

None of the presented techniques deal with the precision issue at the level we are proposing. Presented pointing techniques are all based on camera tracking and/or ray casting with low cost approaches, thus resulting in less precision of pointing. The motion sensors embedded in the newer smartphones allow for a number of gesture-based input modalities. Specific pointing devices (Wiimote, gyro mice...) make extensive use of these sensors, but generic smartphones have not been widely explored for pointing tasks. WYSIWYF [19] uses accelerometer readings and prongs contact with a smart board to position and orient a volume cutting plane. Touching the smart board with the mobile device creates a virtual plane that starts from the contact points between the mobile device and the smart board which have the same orientation of the device. Katzakis and Hori [10] used mobile device accelerometers and digital compass to orient 3D objects applying a direct mapping of orientation across them. On a comparative test, the mobile device approach performed better than mouse and touchpen input. Other works used movement sensors for tasks like distinguishing devices on a multi-touch display wall using tilt correlation [7].

2.2 Levels of precision input

Switching between absolute and relative modes of input, in some sense, allows the use of more than one level of precision for interaction. Absolute pointing is used for quick traveling long distances, while relative movements of the cursor are used for fine tuning, thus also providing fast and high precision pointing.

Vogel and Balakrishnan [20] explored natural input using the naked hand. The proposed technique allows switching across absolute and relative input performing two different hand poses. The user can point to a region at the large screen performing an absolute hand pose, and then, change the hand pose to switch to the relative mode. Similar in concept, Forlines et al. also implemented an analogous for pen interaction with large and high resolution touch surfaces [6]. Users could switch from absolute to relative input mode to achieve targets out of their arms reach area. Both techniques rely on high cost aparatus and a prepared interaction environment (VICON tracking system).

Nancel et al. [14] also proposed a technique using the VICON system. They combine a VICON-based ray casting mode with a precise relative pointing sliding the finger on an Apple iPod Touch device touchscreen. The touch surface is divided in two areas: an upper zone for tracking, and a lower zone for clicking. Touching the upper zone switches to precise mode. Authors claim it is possible to select objects with 4 millimeters standing 2 meters away from the display, which is comparable to some results we show in this paper using only smartphone built-in sensors.

The ARC-Pad [13] implements an absolute plus relative cursor controller using a mobile phone touchscreen. The movement of the finger while holding the touch triggers a relative movement, identical to an ordinary touchpad, but a quick tap on the screen triggers a jump of the cursor to the location defined by an absolute mapping relation with the external display. We implemented this approach and further compared with LOP-cursor at the Section 5.1.

3 Design

3.1 The conception of the LOP-cursor

LOP-cursor was conceived for high-precision pointing on high-resolution displays. The technique is based on two levels of precision. In the first level, the user points the device towards the target on the screen, which results in moving a rectangular control canvas containing the cursor arrow to that location (see Figure 1a). If the user succeed on selecting the desired target with the arrow, the task is achieved. Otherwise, a fine tuning can be done by sliding a finger on the device touchscreen, moving the cursor arrow inside the control canvas (see Figures 1b and 1c). Different from most 2 levels of input techniques, LOP-cursor uses absolute mapping in both levels, ray casting for the first level, and a rectangle representing the physical device touchscreen at the large display. During the fine tuning, the position of the control canvas is locked, but the coarse pointing using the ray casting metaphor remains active (see Figure 1d). This feature – the simultaneous use of the two legs of the cursor – inspired the use of LOP-cursor to perform composed tasks, as select-and-move, for instance.

3.2 Cursor States

The three-dimensional interaction space of our cursor metaphor combined with its two legs allows for an increased pointing capability. To ensure user control while keeping flexibility in this context, we defined three states of interaction for the cursor: (1) free-pointing state; (2) hold-control-canvas + free-pointing state; (3) pin-control-canvas + free-pointing state. The states are detailed below and depicted in Figure 2.

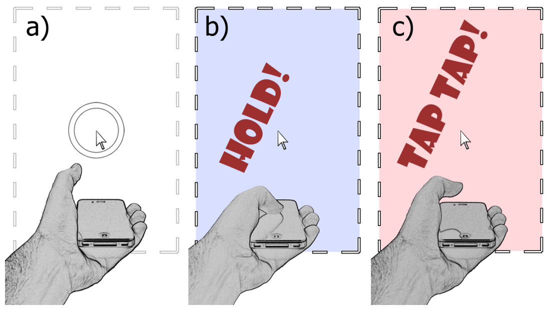

Figure 2: LOP-cursor states: (a) *free-pointing*; (b) holding a touch leads to *hold-control-canvas + free-pointing*, (c) a double tap activate and deactivate *pin-control-canvas + free-pointing*.

3.2.1 Free-pointing

This is a one-legged cursor mode (Figure 2a). While on free-pointing state, cursor positioning is allowed on a direct-pointing only manner. In this state, our metaphor is very similar and, in average, not worse than a ray casting based technique, allowing large targets to be easily acquired. Small differences in performance can appear depending on the way position and direction are gathered, and difficulties related to small targets remain an issue.

3.2.2 Hold-control-canvas + free-pointing

In this state, the two legs of the cursor are active and can be defined (Figure 2b). The cursor enters this state when a touch on the mobile device touchscreen is hold. While in hold, the control canvas keeps the initial touch pointing position as its anchor, so that the user can fine tune pointer positioning by sliding the finger on the touchscreen. Unlike most touchpads, here touch is mapped with an absolute relation between the devices touchscreen and the control canvas.

While the leg-1 of the cursor is finely controlled inside the very stable control canvas, wrist movements can still displace and rotate the device, giving rise to the leg-2 of the cursor. Leg-2 is then a cursor pointer controlled by free-pointing that can be used as a lower precision secondary and simultaneous cursor. Interaction scenarios for leg-2 are described later in this paper. Touch take-off results in the release of the control canvas. On releasing, the control canvas silhouette assumes the current position of leg-2, and the two-legged cursor reunifies, returning to the free-pointing only state.

3.2.3 Pin-control-canvas + free-pointing

Starting from a free-pointing state, a double tap on the device touchscreen pins down the control canvas (Figure 2c). Pinning position is defined by the pointing position at the start of the first tap, which minimizes the occurrence of pinning on an undesired position due to device displacement while tapping.

While pinned, the touch take-off does not trigger a state change. The state is maintained until the user performs another double tap. This allows for a number of touchscreen actions to be performed. For example, a single tap can be used for selection. Meanwhile, sliding a finger is used to fine pointing inside the control-canvas analogously to the hold-control-canvas state above. This state allows more stable use of simultaneous cursors.

3.3 Graphic representation

Four complementary shapes are used to represent the LOP-cursor which are combined to maximize usability. They are described here in association with the terms already used above to introduce the cursor states.

3.3.1 Arrow: the leg-1 pointer

To benefit from the high precision positioning capability of LOP-cursor, a traditional point selector over an area selector was preferred for our main cursor design. We used the well known arrow shape to represent this cursor, which minimizes ambiguities. We believe that using the arrow to highlight the leg-1, which is the ultimate pointer in LOP-cursor, helps users to immediately differentiate it from the other shapes we used.

The arrow opacity depends on the size of the nearest target to the cursor, the cursor size itself, and the distance between the target and the arrow center. Using adaptive opacity avoids the occlusion of very small targets by the arrow and allows users to keep track of the target whenever it is required. Figure 3 illustrates the arrow and its opacity function.

3.3.2 Rectangle: the control canvas

A rectangle is used as the representation of the mobile device touchscreen on the display. It defines a control canvas within which the arrow is. An absolute mapping is used between the device's rectangular touchscreen and the control canvas, e.g. a touch at a location near a corner of the touchscreen results in placing the arrow at the same corner of the control canvas. This absolute approach makes the rectangle a natural choice for the virtual representation of the touchscreen. More important, this approach allows for the control canvas representation to contain an input resolution equivalent to that of the device's touchscreen sensing resolution (480 x 320 in our implementation). The rectangle has also the same aspect ratio of the touchscreen, making the correspondence explicit to the user.

To allow user control over the leg1 cursor precision, *pinch* and *stretch* gestures resize the control canvas proportionally to its current size. In our implementation, the default input scale is 1:2, where each pixel on the touchscreen is represented by a 2x2 area of pixels at the large screen. For instance, to achieve higher precision, the user can resize the control canvas to a 10:1 precision, and thus, a 10x10 area of pixels on the touchscreen would be mapped to only 1 pixel of the large screen.

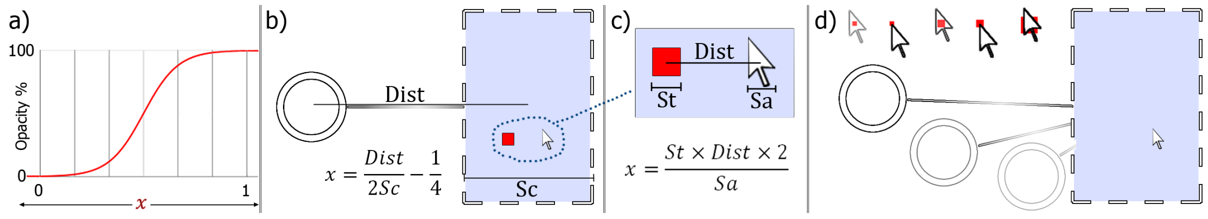
a)
100
Opacity %
0
0
1
x
b)
Dist
$x = \frac{Dist}{2Sc} - \frac{1}{4}$
Sc
c)
Dist
St
Sa
$x = \frac{St \times Dist \times 2}{Sa}$
d)

Figure 3: A logistic function maps the arrow and ring opacity according to the x variable value (a); x for the ring depends on distance between the center of the ring and the center of the rectangle $Dist$, and the width of the rectangle Sc (b); x for the arrow depends on distance between center of the target and center of the arrow $Dist$, and width of arrow Sa and target St (c); arrow and ring opacity sample results for a variety of target sizes and positions, and rectangle and ring distances (d).

3.3.3 Ring: the leg-2 pointer

We chose a relatively large ring shape instead of an arrow for leg-2 because it is subject to jittering and precision cannot be guaranteed.

Leg-2 indicates the physical pointing direction of the device. It always follows the device's orientation, enabling the user to keep track of where they are pointing to. This is particularly relevant when the control canvas is in hold or pinned and the two legs are separate. In this occasions, when the focus of the user is on leg-1, involuntary hand movements eventually take the ring to locations far from the control canvas and the leg-1 arrow. After release, the control canvas jumps back to follow the device's orientation and consequently the location of the leg-2 ring pointer. Experiments show that while this jump does not seem to upset the user, it can cause disorientation when the ring pointer is not displayed.

To avoid distraction caused by continuous movement of the ring near the arrow, we used an adaptive opacity factor. The ring opacity is proportional to its distance from, and size of, the control canvas. The nearer the ring is from the control area, more transparent it is. Figure 3 illustrates the concepts above.

3.3.4 Line: a bridge between leg-1 and leg-2

A line is used for connecting the control canvas (rectangle) and the leg-2 (ring). This aids the user to keep track of the relative position of legs 1 and 2 and to warn where they are currently pointing to. The line is specially useful when the two legs are separated by a large distance. For example, while simultaneously working with the two cursor legs, the user will need to quickly switch attention between them. The line guides searching direction, while the color intensity guides on the distance between legs.

3.4 Transitions between states

While in free-pointing state, both the arrow and ring pointers follow the mobile device pointing direction. The arrow is kept at the center of the ring, and a semi-transparent rectangle with dashed borders is shown to keep the user aware of the size of the control canvas. As the control canvas and the ring cursor are superposed and move together, the line is not visible in this mode.

When switching from free-pointing to hold-control-canvas + free-pointing, or from free-pointing to pin-control-canvas + free-pointing, a transition effect is played to guide user attention across states. The effect renders the rectangle with more opacity, and ring transparency is controlled as described in Figure 3, giving a hint of switching modes. The inverse effect is played when returning to the free-pointing-only state. To inform the user about the current state, *holding* fills the rectangle with a blue transparent tone, and *pinning* fills the rectangle with a red transparent tone.

In preliminary tests, due to absolute pointing within the control canvas, the arrow cursor often jumped when entering in hold-control-canvas + free-pointing state. This caused a discontinuity that led the users to report some disorientation. To overcome this issue, in the final design of the LOP-cursor the arrow is kept in place and the rectangle (canvas) is placed accordingly to compensate the distance of the touch from the center. Although this causes a small motion discontinuity for the canvas (it is small because the users tend to touch at the center of the device touchscreen), it was preferred than a discontinuity of the arrow as the user focus is on the arrow.

4 PROTOTYPE IMPLEMENTATION

We implemented the LOP-cursor in the context of an experimental interface. Our prototype consists of interaction challenges that cover many of the interaction needs when working with large high resolution displays. The challenges are simple selection, drag and classification tasks with low cognitive load. Real world applications, which can also involve 3D environments, are not addressed here.

4.1 Hardware and Software

Input and output hardware in our implementation consist respectively of a smarphone and a tiled display wall.

Our display wall is a 16 LCD monitors tiled-display, disposed on a 4 x 4 matrix. Each monitor has 1,680 x 1,050 pixels and 22 inches diagonal. The total pixel count is 6,720 x 4,200 = 28,224,000 pixels ($\approx$28 megapixels). The display wall is controlled by four PC Core2Quad, with two NVIDIA GTX 285 each. See Figure 4 for an overview photograph of the prototype in operation.

The smartphone used in our main implementation is an Apple iPhone 4. An iPod Touch (fourth generation) was also used for preliminary and comparative testing. As the smartphones used do not present a general purpose embedded back button, we adapted a Microsoft Wireless Mobile 3500 Mouse, which offered a reliable wireless communication.

4.2 Orientation acquisition

In our prototype, we adapted the strategy proposed by Madgwick [12] which combines gyroscope, accelerometer and digital compass information to gather a more robust orientation. This orientation acquisition method relies on the *gyroscope* to provide instantaneous orientation changes (providing 3 axes angular rate of change), and gradual adjustments using two distinct vectors related to absolute frames of reference, the *accelerometer* and the *magnetometer* (respectively related to gravity and magnetic north pole). In practice, we correct gyroscope cumulative rotational error in two axis using the accelerometer (roll and pitch) and in a third axis using the magnetometer (yaw).

Magnetometer readings are less precise and more error prone than *accelerometer* readings, suffering from magnetic interference from near metal structures. Raw readings from the iPhone oscillate within the range of 10% in our environment. Although it does not seem too much, it is crucial to understand that, as a ray is casted using device orientation, any variation is greatly magnified. Because of that, two additional steps were implemented to allow long

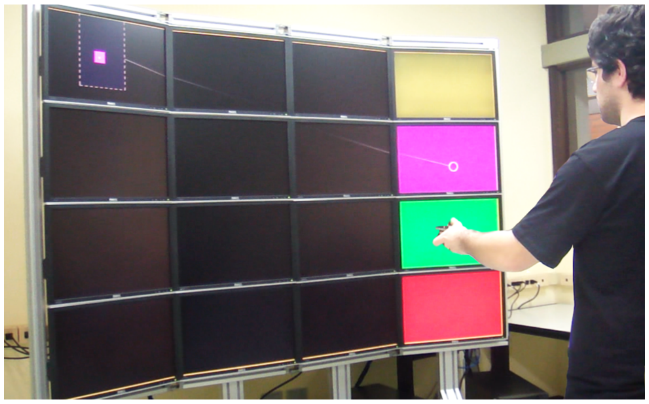

Figure 4: Prototype overview. This photograph depicts a user interacting with the two legs of LOP-cursor. Notice that while leg-1 selects a square on the left, the leg-2 ring indicates the location on the right to where the square will be moved.

term use of the *smartphone* as a pointing device. In the first step, to force the *magnetometer* to only adjust yaw drift error, the tridimensional vector provided is projected into a plane orthogonal to the accelerometer vector. In the second, to ignore high frequency *magnetometer* reading variations, instead of using a filter – which would result in poorer interaction due to delay – we are using a redundancy controller based on the gyroscope updates. More specifically, in this step we ignore small variations from the magnetometer readings whenever the gyroscope readings cannot confirm that a rotation actually occurred around that axis.

4.3 Position calibration

As smartphones and other *every-pocket* mobile devices are not equipped with position tracking, we used only orientation to define pointing and assumed a constant position of the user while testing our prototype. To define this position and reconstruct rotational zero position of the device, our system uses a calibration step. Notice that this step is not necessary when position tracking is used.

For magnetometer calibration, the device must be placed face up pointing orthogonal to the plane defined by the display. A calibration command registers an orientation offset. Next, the calculation of approximate user position (P_3) is achieved registering two orientations of the device. The user is asked to aim the device at a blue point P_1 at one corner of the screen, and then at a red point P_2 at the opposite corner. This calibration method assumes that the real distance between blue and red dots is known, and that the display is perpendicular to the world Z axis.

The two registered orientations are used to retrieve two normalized vectors $\vec{v}_1$ and $\vec{v}_2$. A third vector $\vec{v}_3$ is calculated as $\vec{v}_3 = P_2 - P_1$. The angles α_1 between $\vec{v}_1$ and $\vec{v}_2$, α_2 between $-\vec{v}_1$ and $\hat{v}_3$, and α_3 between $-\vec{v}_2$ and $-\hat{v}_3$ are computed by dot product. The length of the segments given by $\vec{v}_1$ and $\vec{v}_2$, respectively, are calculated as

$$l_1 = \frac{|\vec{v}_3| sin(\alpha_2)}{sin(\alpha_1)}, l_2 = \frac{|\vec{v}_3| sin(\alpha_3)}{sin(\alpha_1)}. \quad (1)$$

Then, we estimated device positions $P'_3 = P_1 + l_1(\vec{v}_1)^{-1}$ and $P''_3 = P_2 + l_2(\vec{v}_2)^{-1}$. A mid point between P'_3 and P''_3 is used as the final P_3.

Limitations of this calibration approach is that significant changes in the position of the user will result in the need for a new calibration, or at least some cursor offset control. Also, the rotational *zero* will depend on which of the joints (shoulder, elbow, wrist) the user performs the rotations.

5 EVALUATION

We conducted two sets of user tests, both using the prototype presented in Section 4: a comparative evaluation, and a deeper exploration of the LOP-cursor capabilities.In all tests, target start and goal positions were constrained to never intercept a monitor bezel since, as stated by [3], this can be detrimental to some user interaction aspects.

5.1 Comparative evaluation

5.1.1 Design

To validate the LOP-cursor technique, we conducted a comparative evaluation on a selection task. LOP-cursor was compared to an *ARC-pad* [13] implementation, and a Ray Casting using only device orientation (*ORayCasting*), a technique equivalent to limiting LOP-cursor to the use of the lower precision level of pointing. To remove subjective bias on choosing the level of precision, the LOP-cursor implementation used for evaluation constrained users to always use the two levels of precision for selections. We call this implementation (*CLOP-cursor*). CLOP is actually the same as the original LOP but free-pointing is not effective for selection triggering. Thus, with the CLOP-cursor, users had to always perform steps *a* through *c* as described in Figure 1.

Independent variables are: *Technique*: ARC-Pad, LOP-Cursor and ORayCasting. Target *Size*: 0.5cm, 1cm, 2cm, 4cm. Target *Distance*: 25cm, 50cm, 100cm. Dependent variables are: *Time* and *Error rate*. We used a *within-subject* design. Technique exposure was counter-balanced, while size and distance of target were randomly presented. Pointing to a target and triggering a selection counted as a *Trial*. There were a total of 10 *Trials* for each independent variable combination. Each technique evaluation was divided in 5 *Blocks*, where a *Block = Sizes x Distances x 2 Trials*, giving a total of 24 *Trials* per *Block*. The first 2 *Blocks* were used for practicing, and thus are not considered on further analysis. Participants were allowed to take a non-mandatory break between blocks.

After the comparative evaluation, another selection test where the complete *LOP-cursor* is used was taken. In this case, subjects were not constrained to always use the higher level of precision. They could decide when and if they want to use two levels of precision, or only one. This was intended to evaluate for which sizes of target users preferred to use only the ray casting level of precision, or both of them, thus allowing us to infer if participants subjective preference match to the previously applied comparative evaluation best times per size. Six target sizes were used: 0.5cm, 1cm, 2cm, 4cm, 8cm, 16cm. Each test had 5 *Blocks* of 18 *Trials*, 3 trials per size condition. First 2 *Blocks* were used as training, thus being discarded for analysis.

Active targets were drawn in red on a black background. To avoid bias from visual search, the following target was shown in a dark grey tone. There was an additional starting target at each *Block*. During all evaluation, the user was positioned centered to the tiled-display, at a constant distance of 150cm. Users who completed all the evaluation were then asked for their preferred interaction technique (CLOP-cursor, LOP-cursor, ARC-pad or ORayCasting), and allowed to leave general comments and observations about the evaluation. Users were asked to favor precision over speed.

5.1.2 Implementation details

Comparative tests were taken using an *iPod Touch 4th generation*. Orientation without the magnetometer recalibration may drift on *Yaw* over time (see Section 4.2), but the ray casting was too imprecise when using yaw corrections because of the high variability of the magnetometer readings from the iPod. As this would put the ORayCasting technique in disadvantage relating the other techniques tested we disabled yaw corrections. To limit the effect of yaw drifting we preferred to arrange short blocks of trials (24 trials) and frequently correct any drift between blocks.

The ARC-pad paper [13] does not make clear how a selection is triggered, and seems to suggest its use over a desk. Our test design, instead, proposes the users to stand in front of the screen, and thus they had to hold the device with both hands to maintain consistent aspect ratio orientation across the tiled display and the mobile device. Selection was implemented using a *Tap* gesture from the secondary hand while holding arrow position with the primary hand. See Section 2.2 for a brief description of the ARC-pad technique.

Cursor position was filtered using a *dynamic low-pass filter*, interpolating between cutoffs of 0.2Hz and 5Hz, with 60Hz sample rate. Cutoff is defined according to cursor speed: when $<$ 1cm/sec, lower cutoff is used (0.2Hz); when $>$ 50cm/sec, higher cutoff is used (5Hz). For any speed between these, a linearly interpolated cutoff value is used.

5.2 In depth LOP-cursor evaluation

5.2.1 Design

We conducted this deeper evaluation to assess the limits of our technique in terms of precision, and to initiate the study on simultaneous cursors. Experiments consisted on completing three different tasks using LOP-cursor:

Task 1. Pointing and selecting targets. A trial is complete when the target is hit.

Task 2. Pointing, selecting, dragging and docking targets. Users should point and select the target in the same way they do in Task 1: drag it over a grey object with the same shape and size of the target located at a fixed distance of 100 cm, and dock the target over the grey object the most precisely as possible. A trial is complete when the target is released (docked).

Task 3. Using the two legs of the LOP-cursor to select a target and put it into a predefined stock area on the right side of the display (as depicted in Figure 4). The target should be selected with the leg-1 using the two levels of precision allowed by the selection technique, and moved to its respective stock area (with the same color of the target) using leg-2. The selection results on a transfer between the legs. The trial is complete when the user successfully trigger a selection while simultaneously aiming with both cursors, on both target and goal area.

The presented tasks add in complexity, and thus, they were always applied on the same order, task 1 through 3. We used four different *Sizes* of targets (0.3, 0.8, 2 and 4 cm). The initial position of targets were randomly defined. For *Task 3*, targets appeared only at the left-half of the tiled display.

We also found that the 0.3 cm target could introduce delay due to visual search. Then a green target was shown between each trial for tasks 1 and 2. Green target and the trial target(s) appeared at the same time. Having found the trial target(s), the user selected the green target to start the trial.

The procedure consisted on two practice *Blocks* of two *Trials* for each *Size* (total of *eight trials per training block*) and one evaluation *Block* with six targets per *Size* (total of *24 trials*) for each *Task*. The first practice block was used to explain task goal, and to demonstrate the LOP-cursor capabilities that would be used during the task. Users were asked to complete trials as quick as possible, but without sacrificing precision over time. On the second practice block users interacted by themselves, with minor hints of the test conductor when needed. During the evaluation block, subjects were left by themselves, and no help was given. After the evaluation, users filled a System Usability Scale (SUS) questionnaire.

An iPhone 4 disposing of a magnetometerwas used on these tests. This allowed orientation to be gathered as described in Section 4.2, allowing longer blocks to be taken. As this evaluation aimed to assess the limits of our technique, users were constrained to always use the two levels of the LOP-cursor (the so called CLOP-cursor).

6 Results and Discussion

6.1 Comparative evaluation

Eleven participants took place on this evaluation, (mean age 22, 2 women and 2 left handed) all of them Computer Science students (10 undergraduate, 1 master student). Most of them had previous experience with pointing devices (mostly Wii gaming), all of them had at least some experience with a mobile device touchscreen, only two had significant experience with very large displays.

From the comparative test we had 2,376 total valid *Trials*. A trial was considered a false positive (accidental selection triggering) when selection occurred at any of the following cases: *20cm* or farther from the target; less then *200ms* after the previous trial was completed. CLOP-cursor, ARC-pad and ORayCastig presented 2, 6 and 9 false positives respectively. These *Trials* were overlooked on further analysis.

All subjects completed the comparative evaluation, but three of them could not complete the last evaluation (using the non-constrained LOP-cursor). This was due to the long time taken by the comparative evaluation (from 30 to 40 minutes to complete), resulting on schedule and fatigue issues for some subjects. This test had a total of 432 trials, 2 of which false positives.

General mean time for a selection with CLOP-cursor, ORayCasting and ARC-pad were respectively: 2.55, 2.46 and 3.05 *seconds*. One-way ANOVA showed that ARC-pad was significantly slower than CLOP-cursor (F(1.1574)=76.58, $p<0.0001$) and ORayCasting (F(1.1567)=84.81, $p<0.0001$). See Figure 5 for detailed mean time and standard deviation for each target size.

Error rate for ORayCasting is significantly higher than CLOP-cursor and ARC-pad (F(1.1571)=109.08, $p<0.0001$ and F(1.1567)=120.6, $p<0.0001$). Error difference between CLOP-cursor and ARC-pad is not significant (F(1.1574)=0.48, $p<0.49$), even considering only the 1cm targets (F(1.391)=1.93, $p<0.17$). Error rates are presented on Figure 6.

We also found significant learning effect on mean time across blocks for CLOP-cursor (F(2.787)=4, $p<0.019$) and ARC-pad (F(2.783)=3.04, $p<0.0484$). Although a reduction of error rate did occurred for CLOP-cursor (Block1=12, Block2=11 and Block3=5) there was no significance (F(2.787)=1.61, $p<0.2$).

On the LOP-cursor trials (post comparative evaluation) error rate followed a tendency between CLOP-cursor and ORayCasting (see Figure 6). This unconstrained LOP-cursor presented lower error rates for small sizes than the CLOP-cursor, even though they rely on the same fine tuning technique. As this technique was always presented to users as the last one, we believe learning effect did affect error rate, but there was not enough data to statistically prove such variation. Detailed error rate for each level showed consistency with CLOP-cursor and ORayCasting previously applied comparative test for most target sizes (Figure 7).

Subjective user preference between 1 or 2 levels of precision according to target size is presented in Figure 8. For target sizes of 0.5 and 1 cm users always used 2 levels.

After the evaluation, subjects who participated on all procedures were asked for their preferred interaction technique. The LOP-cursor implementation allowing 2 levels of interaction was preferred by seven users, while ARC-pad was preferred by only one subject.

6.2 In depth LOP-cursor evaluation

Eleven undergraduate students in Computer Science participated on this experiment (mean age of 23, 2 females, all right handed). Most subjects had some experience with pointing devices, all of them had at least some experience with mobile device touch screens. Each test took from 25 to 40 minutes to be concluded.

All users were able to complete the tasks, i.e., they were all able to select every 0.3 cm target at the distance of 150 cm with our technique. The chart on Figure 9 shows the mean time for completion of

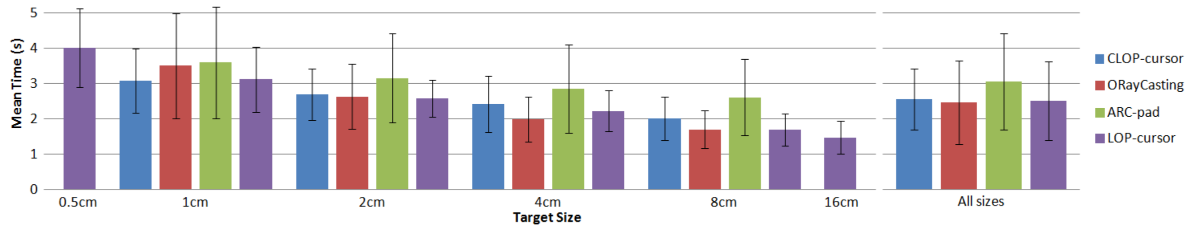

Figure 5: Mean time spent to select targets of different sizes using CLOP-cursor, ORayCasting, ARC-pad and LOP-cursor, and their respective standard deviations.

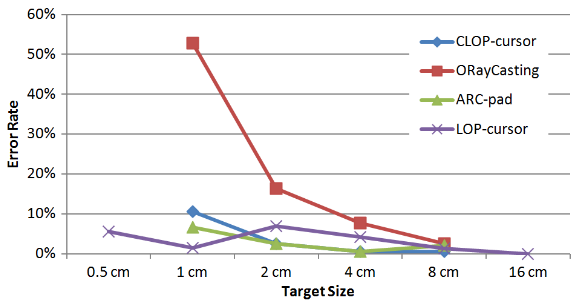

Figure 6: Error rate to select targets of different sizes using CLOP-cursor, ORayCasting, ARC-pad and LOP-cursor.

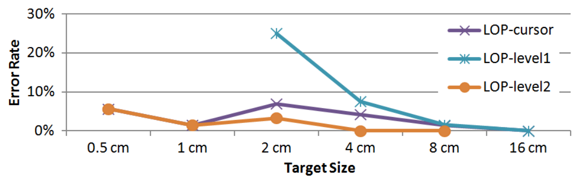

Figure 7: Error rate to select targets of different sizes using LOP-cursor, and individual error rate for each level of precision.

the 3 tasks. Notice that although the larger targets allow for faster task completion, the time difference is always inside the standard deviation, regardless of the huge difference (more than one degree of magnitude) in objects size.

There were 5 false positives (incidental docking) on Task 2, which were discarded for further results. Mean docking distance error is 0.17 cm, being 0.158 cm, 0.162 cm, 0.176 cm and 0.188 cm for target sizes with 0.3 cm, 0.8 cm, 1 cm, and 2 cm, respectively. Docking distance error difference per size showed not to be significant (F(3.255) =1.38, p<0.254). The similarity observed on the mean errors obtained with all target sizes tested indicates that the LOP-cursor is a technique suitable for tasks involving different sizes of objects, with more or less the same accuracy, which is a positive result. The most frequent user complain was related to prototype physical ergonomics (device size and back button position). We noticed that some users slightly moved touch position when triggering a selection command. We believe the ergonomic issue was the main cause of this effect. This was specifically detrimental when selecting the 0.3 cm targets, where subjects needed more than one selection triggering to complete the trial: 42% and 39% for Task 1 and Task 3, respectively. CLOP-cursor scored 77 on the SUS questionnaire, with standard deviation of 12.8.

Task 3 results showed that users could successfuly point at two simultaneous positions. The other existent multiple cursor technique, Ninja Cursors [11], points to only one location at a time. It is difficult to compare performance results between the LOP-cursor and the Ninja Cursors as their evaluation is based on a 2 screens desktop environment, while ours pursuit interaction with much larger and higher resolution displays, away from the desktop and with potential to interaction in 3D environments. Nevertheless we could notice that while the performance of the LOP-cursor is independent of screen target density, the performance of the Ninja Cursors is significantly affected by both the number of cursors and the target density.

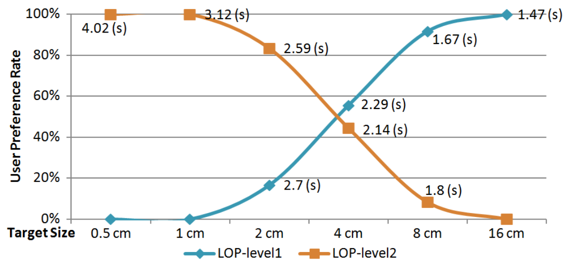

Figure 8: Users preferred level of precision for a variety of target sizes when using LOP-cursor, and their respective trials mean time.

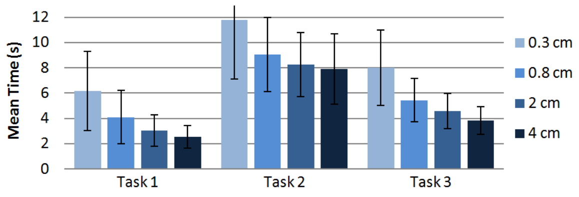

Figure 9: Mean time for completion of each trial of each task.

7 Conclusions and Future Work

While high-resolution displays are a widely explored subject, in this paper we addressed the issues involving high-resolution *interaction* with such displays. As such displays are used away from the desktop, 3D interaction techniques can be helpful even when the application deals with 2D data. We introduced the concept of the LOP-cursor, a two levels pointing metaphor greatly adapted to interact with large high-resolution displays.

We have shown through user studies that ordinary smartphones implementing the LOP-cursor can be a valuable device for interaction with such displays. Our tests demonstrate that using direct pointing to first define the raw location of a cursor, and then a mechanism for fine tuning the raw location to a very precise point, enables the users to quickly and precisely select and drag objects.

The LOP-cursor also introduces a two-legged cursor modality controlled with only one hand. This is a major breakthrough as many everyday user actions with a computer involve quickly defining two locations on the screen to complete the operation (copy files, arrange photographs, etc.).

Concerning the devices, our pointing method requires both the orientation and position of the mobile device. Orientation is retrieved using device built-in movement sensors. In our prototype, position is assumed or given by a simple calibration step. A tracking system for position would perfectly fit and enhance our concept, but as tracking requires more complex equipment, we preferred to keep the solution available for everyone who owns a smartphone.

We continue working on LOP-cursor, investigating the use of simultaneous cursors on real life tasks and alternative screen configurations. Further investigations will be held in order to determine to which level users control two cursors simultaneously or one at a time. In addition, we want to access two-legged cursor cognitive load and determine which two-location tasks may benefit from this metaphor. We plan to apply our concept on distributed computing scenarios, with many computers and screens that do not necessarily respect a position pattern, but rather are distributed across the room. Such scenarios can benefit from LOP-cursor for tasks such as file transfer across computers, where the user can pin control canvas on a high resolution display (computer 1), while controlling the file destination pointing to another display (computer 2, 3, 4 ...) with leg-2 cursor. Future works also include the study of leg-2 serving as a controller for another dimension, such that, its distance from (or circular movement around) the leg-1 can control selected object attributes(scale/depth/orientation).

Although the LOP is based on 3D interaction techniques, the evaluation in this paper focuses on a controlled 2D task. Nevertheless, we see a great potential for using the LOP in 3D environments. One possibility we are investigating is to control the canvas orientation in $\mathbb{R}_3$ using the device's orientation while pinned. We believe in this approach based on works in the literature showing that users quickly map 2D displacements from a given plane to any arbitrary plane, as for instance, the mouse on a table to the arrow cursor on a screen. Non-desktop 3D environments, as a CAVE, could benefit from the bases of LOP-cursor with some incremental development to add these arbitrary planes.

We also plan to investigate how the user distance from the screen and the screen size affects the optimal scale for the control canvas. As stated by Peck et at. [18], users tend to interact in different scales according to their distance from screen. Naturally, when users are near the display, they can see and point with more precision, and thus, a smaller control canvas allowing more precision would also make sense (Section3.3.2).

Acknowledgements

This work was supported by CNPq-Brazil under the projects 309092/2008-6, 483947/2010-5, 483814/2010-5 and 302679/2009-0. Thanks are due to all the volunteers who kindly tested the system.

References

[1] R. Ballagas, J. Borchers, M. Rohs, and J. G. Sheridan. The smart phone: A ubiquitous input device. *IEEE Pervasive Computing*, 5:70–, January 2006.

[2] R. Ballagas, M. Rohs, and J. G. Sheridan. Sweep and point and shoot: phonecam-based interactions for large public displays. In *CHI '05 extended abstracts on Human factors in computing systems*, CHI EA '05, pages 1200–1203, New York, NY, USA, 2005. ACM.

[3] X. Bi, S.-H. Bae, and R. Balakrishnan. Effects of interior bezels of tiled-monitor large displays on visual search, tunnel steering, and target selection. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 65–74, New York, NY, USA, 2010. ACM.

[4] S. Boring, D. Baur, A. Butz, S. Gustafson, and P. Baudisch. Touch projector: mobile interaction through video. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 2287–2296, New York, NY, USA, 2010. ACM.

[5] S. Boring, S. Gehring, A. Wiethoff, A. M. Blöckner, J. Schöning, and A. Butz. Multi-user interaction on media facades through live video on mobile devices. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 2721–2724, New York, NY, USA, 2011. ACM.

[6] C. Forlines, D. Vogel, and R. Balakrishnan. Hybridpointing: fluid switching between absolute and relative pointing with a direct input device. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, UIST '06, pages 211–220, New York, NY, USA, 2006. ACM.

[7] W. Hutama, P. Song, C.-W. Fu, and W. B. Goh. Distinguishing multiple smart-phone interactions on a multi-touch wall display using tilt correlation. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 3315–3318, New York, NY, USA, 2011. ACM.

[8] S. Jeon, J. Hwang, G. J. Kim, and M. Billinghurst. Interaction techniques in large display environments using hand-held devices. In *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '06, pages 100–103, New York, NY, USA, 2006. ACM.

[9] H. Jiang, E. Ofek, N. Moraveji, and Y. Shi. Direct pointer: direct manipulation for large-display interaction using handheld cameras. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 1107–1110, New York, NY, USA, 2006. ACM.

[10] N. Katzakis and M. Hori. Mobile phones as 3-dof controllers: A comparative study. In *Dependable, Autonomic and Secure Computing, 2009. DASC '09. Eighth IEEE International Conference on*, pages 345 –349, dec. 2009.

[11] M. Kobayashi and T. Igarashi. Ninja cursors: using multiple cursors to assist target acquisition on large screens. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 949–958, New York, NY, USA, 2008. ACM.

[12] S. O. H. Madgwick, A. J. L. Harrison, and R. Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. In *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pages 1 –7, 29 2011-july 1 2011.

[13] D. C. McCallum and P. Irani. Arc-pad: absolute+relative cursor positioning for large displays with a mobile touchscreen. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, UIST '09, pages 153–156, New York, NY, USA, 2009. ACM.

[14] M. Nancel, E. Pietriga, and M. Beaudouin-Lafon. Precision Pointing for Ultra-High-Resolution Wall Displays. Research Report RR-7624, INRIA, May 2011.

[15] T. Ni, G. S. Schmidt, O. G. Staadt, M. A. Livingston, R. Ball, and R. May. A survey of large high-resolution display technologies, techniques, and applications. In *Proceedings of the IEEE conference on Virtual Reality*, VR '06, pages 223–236, Washington, DC, USA, 2006. IEEE Computer Society.

[16] A. Olwal. Lightsense: enabling spatially aware handheld interaction devices. In *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*, ISMAR '06, pages 119–122, Washington, DC, USA, 2006. IEEE Computer Society.

[17] N. Pears, D. G. Jackson, and P. Olivier. Smart phone interaction with registered displays. *IEEE Pervasive Computing*, 8:14–21, April 2009.

[18] S. M. Peck, C. North, and D. Bowman. A multiscale interaction technique for large, high-resolution displays. In *Proceedings of the 2009 IEEE Symposium on 3D User Interfaces*, 3DUI '09, pages 31–38, Washington, DC, USA, 2009. IEEE Computer Society.

[19] P. Song, W. B. Goh, C.-W. Fu, Q. Meng, and P.-A. Heng. Wysiwyf: exploring and annotating volume data with a tangible handheld device. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 1333–1342, New York, NY, USA, 2011. ACM.

[20] D. Vogel and R. Balakrishnan. Distant freehand pointing and clicking on very large, high resolution displays. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, pages 33–42, New York, NY, USA, 2005. ACM.