# HybridPointing for Touch: Switching Between Absolute and Relative Pointing on Large Touch Screens

TERENCE DICKSON, University of Waterloo, Canada
RINA R. WEHBE, Dalhousie University, Canada
FABRICE MATULIC, Preferred Networks Inc., Japan
DANIEL VOGEL, University of Waterloo, Canada

We propose CursorTap, an extension of Forlines et al.'s mixed, absolute and relative "HybridPointing" to large wall-sized multitouch displays. Our technique uses a relative pointing quasimode activated with one hand, while the other hand controls a distant cursor similar to a large touchpad. A controlled experiment compares the technique to standard absolute touch input as a baseline and a whole-display "Drag" technique representing a common alternate approach. Results show CursorTap is fastest for the common usage scenario of reaching distant targets and then returning to nearby targets. Overall, median selection times across distances are similar with CursorTap, but linearly increase with the other techniques. As further validation, a second study explores how people use CursorTap in a two-person game. The results found just over half of the participants choose to use CursorTap for half of the primary interactions where "enemies" are eliminated using a tap, drag, or lasso "tool".

CCS Concepts: • **Human-centered computing** → **Human computer interaction (HCI)**; **Touch screens**; **Pointing**.

Additional Key Words and Phrases: touch input; large displays; interaction techniques

**ACM Reference Format:**
Terence Dickson, Rina R. Wehbe, Fabrice Matulic, and Daniel Vogel. 2021. HybridPointing for Touch: Switching Between Absolute and Relative Pointing on Large Touch Screens. *Proc. ACM Hum.-Comput. Interact.* 5, ISS, Article 495 (November 2021), 22 pages. https://doi.org/10.1145/3488540

## 1 INTRODUCTION

Touch input almost always uses *absolute direct input*, where the finger manipulates graphical objects by touching them directly. While this is well suited to personal devices like phones and tablets, it can be problematic for very large displays. Not all parts of a large display may be within reach, so the user may have to stretch their arms, twist their waist, stand on their toes, or walk to interact with content outside a comfortable "arms-length" range [32]. With multiple users, people may have to reach in front of each other, which feels intrusive [2] and might occlude parts of the display

This paper is based on a Dickson's Master's thesis [10].

Authors' addresses: Terence Dickson, University of Waterloo, Canada; Rina R. Wehbe, rina.wehbe@dal.ca, Dalhousie University, Canada; Fabrice Matulic, fmatulic@preferred.jp, Preferred Networks Inc., Japan; Daniel Vogel, dvogel@uwaterloo.ca, University of Waterloo, Canada.


2573-0142/2021/11-ART495 $15.00
https://doi.org/10.1145/3488540

from other users. In addition, there are issues related to parallax [30], front-projection shadows, and friction when dragging that can reduce the effectiveness of absolute direct touch input.

Forlines et al. [11] proposed using *relative indirect input* with a pen on a large wall-sized display to address many of the issues identified above. They recognized that users could simply switch between normal absolute direct pointing and relative indirect pointing using a cursor, so they designed a *HybridPointing* technique for pens. Since pens have limited input capabilities, they use a method they call a trailing widget to perform the mode switch. A quick and accurate movement of the pen "traps" a nearby floating button which switches pen input to controlling a cursor with relative indirect input until it is lifted away from the display.

The Forlines et al.'s method has several desirable design qualities compared to other distant pointing methods. First, it is *target independent* since it does not need application-level access to analyze target sizes and locations, unlike Vacuum [6] and Drag-and-Pop [5]. Second, it is *minimal* since it does not require a persistent interface with multiple controls for modes and actions, unlike WallPad [12] and Frisbee [16]. Third, it *preserves full spatial context* since the distant target to select is not transformed or mirrored, unlike all four examples noted above. Fourth, it *requires only basic sensing of a single surface* unlike methods that also require precise 3D tracking like TractorBeam [24] and Pointable [4], additional touch devices like ARCPad [20], or a combination of several input modalities like Pfeuffer et al.'s gaze, pen, and touch method [25].

We extend Forlines et al.'s idea of mixed absolute and relative HybridPointing to very large multitouch displays following the same four design principles. The Forlines et al. method used a mode switch designed for the single input point of a pen by exploiting accurate pen hover sensing. Hover sensing is not common with non-pen touch screens and multitouch enables a larger input space of multiple finger contacts that can simplify the absolute-to-relative mode switch. We propose a bimanual interaction technique called *CursorTap*, where one hand triggers a kinaesthetically held quasimode [26, 29] for relative pointing and the other hand controls a distant cursor in the manner of a large touchpad. The quasimode is activated with three fingers to preserve common single and two-finger panning and zooming operations. A controlled experiment compares CursorTap to standard absolute touch input as a baseline, and a simple *Drag* relative pointing technique, where all display content may be temporarily dragged close to the user. Our results show CursorTap is fastest when accessing distant targets and returning again to nearby targets, a common large display usage scenario. Overall, observed selection times across small to large distances is relatively constant for CursorTap, but linearly increases for the absolute baseline and Drag relative pointing method. As a further validation, we also conduct a study exploring if people would use CursorTap in a more open setting. We developed a two-person game played on a large display that provides opportunities to use CursorTap to eliminate waves of slowing moving "enemies" using tap, drag, or lasso "tools". Our results found 8 out of 14 players freely chose to use CursorTap to eliminate at least 5% of the enemies, and within these 8 players, CursorTap was used 51.5% of the time.

We make two contributions: (1) a hybrid interaction technique to switch between absolute and relative touch input on a wall-sized display; (2) validation of the technique performance in a controlled setting and a evidence that many people will use it in open-ended tasks.

## 2 RELATED WORK

We survey work comparing absolute and relative input, touch interaction techniques, as well as multi-user considerations. Our focus is on touch input on large vertical displays, but other devices and input are discussed if relevant.

### 2.1 Comparisons between Absolute and Relative Input

Forlines et al. [11] introduced a HybridPointing mode-switching technique that allows both absolute and relative pointing using a large pen-based wall interface, and it relies on pen hover for relative to absolute switching. We follow their experiment protocol, with modifications made to accommodate differences in our technique and the touch screen context. We exploit multitouch for switching rather than hover sensing since it is not typically available for touch.

Many relative targeting techniques have been developed for mobile devices for use on the device itself or to control a cursor on a large display. Kim et al. [17] designed interaction techniques for smartphones and other small touchscreens that bring targets closer to the user's thumb while holding the device in one hand. Their thumb-lever technique multiplies the user's thumb movements, acting like a simplified version of our *CursorTap* technique, but without support for interaction. Their mode-switching gestures are tailored for small smartphones. Though little performance improvements were observed across different mode-switching operations and interaction techniques relative to the baseline, they found users preferred a relative method. ARC-PAD uses the touchscreen of a mobile phone to control a cursor on a large display [20]. It combines absolute positioning with direct tap and relative targeting using dragging motions. Since taps cannot be used for clicks, confirming actions are performed with a separate button. Nancel et al. use subareas of the touch screen of a handheld device and relative acceleration to enable high-precision pointing on a wall display [22]. One variation also uses head movements for initial coarse but rapid pointing, while the mobile device allows for finer-grained local targeting near the destination with touch. The techniques are reported to perform as well or better than distant pointing methods. Ikematsu and Siio's Carbon Copy technique [14] temporarily switches a laptop trackpad to absolute mode for a pen to input short content like a signature or a mathematical formula. A trackpad is much smaller than a large display. Furthermore, targeting performance was not evaluated. Our CursorTap technique does not rely on pens or mobile devices and uses the touch surface of the wall display for direct finger-based input.

### 2.2 Interaction Techniques for Large Touch Displays

We review work using pen, mid-air, and touch input.

*2.2.1 Distant Reaching using Pen Input.* Several works using pen input are associated with a sitting posture on a horizontal surface. Reetz et al. [28] propose a method for sending targets along a large table surface. A quick flicking action creates an intuitive motion, but with low accuracy. Pen hover input moves the object after it is flicked away, supplementing accuracy. TractorBeam [24] is a pen-based distant input method for a table, where the pen can be used in the air to point at a remote target like ray-casting. This enables interactions at a distance and pulling targets toward the user to be manipulated using absolute pen input, but precise 3D input implies more than *basic sensing*. Nacenta et al. [21] compare several pen-based techniques to transfer objects from a tablet to a tabletop display and find that a miniature representation of the target area (a "radar") is most effective. Their use of a persistent dedicated interface element means the technique is not *minimal*, and since it uses an external tablet, it requires more than *basic sensing of a single wall-sized display*.

Some pen-based methods were designed and evaluated on large displays. Baudisch et al.'s Drag-and-Pop and Drag-and-Pick [5] enable dragging an object (such as a file icon) to a distant actionable target (such as a folder). By creating local copies of the distant targets, the need to interact at a distance is eliminated, and they observed improvements in selection time for long target distances. The method is developed for a pen sensed with large single-point resistive touch display. Collomb et al. created extensions of these techniques that use local dragging and rubber bands to place objects at distant locations [9]. The Vacuum [6] creates local proxies of distant objects within a

user-specified arc to facilitate multiple sequential selections. Large and distant targets are reported to be faster to select using the technique. Like Forlines et al., the method uses pen hover information which is not possible on most large touch screens. Each of these techniques manipulate the position of individual targets, meaning they are not *target-independent* and would require lower level access to the application interface layer. Our multitouch technique has the same goal as Forlines et al. did with a pen, a generally applicable *target-independent* distant pointing method.

*2.2.2 Reaching by Combining Touch with Mid-air and Other Modalities.* Given the standing posture and large display context, previous research has explored how distant targets can be acquired using mid-air gestures [31, 34], sometimes in combination with touch. Jakobsen et al. [15] analyzed a combination of touch and mid-air gestures to select targets on large touch screens. They found participants choose mid-air gestures to select distant targets instead of reaching out to use absolute touch, but using the distant gestures were typically slower. Pointable [4] is a long-distance interaction method for touch screens using ray-casting, similar to the pen-based TractorBeam. By tracking the 3D position of the finger, the user can switch between ray-cast pointing and direct touch. The long-distance method did not show a clear performance improvement over direct touch. Talaria combines touch to interact with local targets and mid-air pointing to target distant locations on a large display [27]. The technique is used to rapidly move objects across long distances on the screen. Gunslinger [19] provides an implementation of mid-air gestures performed at the user's sides. Using the technique to select distant targets on a large touch screen was proposed, but not tested with touch, and only when the user was far from the display. Pfeuffer et al. [25] propose zooming techniques that combining the standard two-finger "pinch/spread" gesture with gaze or pen input to specifies the zoom pivot point. This can be used to access far targets by transforming the entire interface. Using additional input modalities does not fulfill our goal of *basic sensing* for methods that require only multitouch. However, our Drag baseline technique explores this general idea in a restricted form using multitouch and interface translation only.

*2.2.3 Reaching Using Touch Input.* There are prior works that propose methods for acquiring distant targets on large touch screens, without additional hardware. Su et al. [33] propose a touch-based technique where the user may "jump" from target-to-target by inputting a movement direction. This is a target-dependent technique specifically tailored to situations with very small, dense targets, which may not generalize. The technique is not tested on large touch screens, and is also not compared to a baseline of absolute touch. WritLarge [38] is a design study exploring a range of interaction techniques for managing content on interactive whiteboards by using combinations of multitouch and pen input. It includes a Zoom-Catcher method related to reaching, where a 2-finger touch gesture scopes and selects distant content. However, it is a multi-object selection method, not general target selection, and it is designed to support pen input. I-Grabber is a touch-based technique using a reach extender metaphor to grab distant objects [1] and Lever Cursor is a reaching technique for tabletops supporting both absolute and relative modes [39]. Both techniques use two fingers, so classic pinch-spread gestures cannot be used together. Furthermore, I-Grabber was not evaluated and Lever Cursor was not shown to improve on either relative or absolute targeting. The small display area used in the evaluation limited target distance to 100cm, which in our evaluation is the smallest distance we test.

Wehbe et al. [36] examine patterns of cooperative and competitive gameplay on a large touch display. Their system includes a hybrid pointing technique, but it is not described in detail nor is it the focus of their work. Our second study uses the same game design but our focus is on evaluating the CursorTap interaction technique, not collaborative behaviours between participants.

WallPad [12] addresses long-distance interaction on large touch interfaces by allowing large virtual touchpads to be created. A distant cursor is manipulated by dragging only inside this defined

rectangle, and pressing a large rendered button for distant "clicks." To activate the touchpad, a two-handed gesture is used where one finger "slides off" the end of the other finger. A featured use case is creating local copies of remote interface elements, where remote content is rendered inside the touchpad area. Rendering and using a literal touchpad has the advantage of familiarity, but it introduces input constraints given the limited area for controlling a remote cursor and fixed button area for clicking. Furthermore, the slide-off activation gesture appears complex and unintuitive compared to standard multitouch gestures. Finally, the method was not evaluated in a controlled study, nor compared to any baseline.

Frisbee [16] shares some similarities to Wallpad in that it also creates a local rendering of a distant area inside a user interface with buttons and on-screen controls, and it enables distant interaction through direct touch on this local copy. It also features a magnification method. A study found the technique reduced selection time for distant targets compared to using absolute input. The Frisbee interface is always persistent on-screen, and can be moved only using specific touchscreen buttons.

Compared to techniques like WallPad and Frisbee, which rely on proxy widgets such as virtual touchpads or lenses to interact with remote content, our CursorTap technique is simpler, more general-purpose, and addresses multitouch input. We focus on an efficient method to switch between absolute and relative input, as Forlines et al. did with a pen.

## 3 RELATIVE TOUCH TECHNIQUES

In this section we describe the interaction design of the CursorTap technique and the Drag relative baseline technique. We also provide details about the large display system.

### 3.1 CursorTap

CursorTap is a HybridPointing technique where the user can switch from absolute to relative input. By default, the user uses absolute input, where finger taps and drags manipulate graphical objects on the display. To switch to relative input mode, the user touches the display with at least three fingers of one hand to mark a clear distinction from single and two-finger interactions that are commonly used for panning and zooming. Once activated, touching the screen with a finger of the other hand creates a cursor at the touch point, and subsequent movement of that finger controls the movement of the cursor (Figure 1). This is a simple form of Guiard's kinematic chain model for bimanual interaction [13], where one hand sets the context for actions performed with the other hand. With CursorTap, the context is the input mode. Unlike Guiard's model, our system permits either hand to make the mode activation posture, though we expect users would most often use their non-dominant hand so that the dominant hand is available for cursor control.

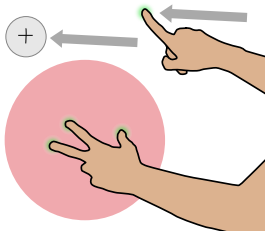

Fig. 1. The CursorTap HybridPointing technique is a relative pointing mode activated when three or more fingers of one hand touch the display. When active, the other hand controls a distant cursor similar to a large touchpad, with up-and-down taps for selection. The pink area illustrates the "dead zone" for the fingers of the activating hand.

The cursor exists for as long as at least one finger of either hand remains on the screen, making this a "quasimode" [26] with kinaesthetic feedback known to reduce mode errors [29]. This allows for a *clutching* movement, where the user keeps one hand stationary against the screen and repeatedly drags and lifts the finger of the other hand to move the cursor. When all fingers are lifted from the screen, the cursor vanishes and the mode returns to absolute direct input.

To avoid spurious taps caused by the user's palm or other fingers, we create a dead zone encompassing the fingers that are not used to move the cursor. This dead zone is visible to the user, and any finger movement within this zone will not contribute to cursor movement, nor will any taps within this zone cause the cursor to register a tap.

Unlike the devices supported by the pen-based HybridPointing design, most touch screens do not have a method for tracking the position of the user's finger when it is not in contact with the screen. Thus, our design simply allows the user to tap while the cursor is active (i.e. while the stationary fingers are still touching the surface) in order to click at the location of the cursor. This method means users can simply lift all their fingers to return to absolute input, while still being able to clutch and tap without leaving the relative input mode.

When moving the cursor in relative mode, we apply a pointer acceleration transfer function similar to how cursor control works in a desktop system [7]. This function dynamically changes the *CD gain* (the ratio of cursor movement speed to input control speed) as a function of input speed (i.e. finger speed in our case). We followed Nancel et al.'s general guidelines [23] for tuning the function so that at low finger speeds, the cursor speed is approximately 1:1, but with high speed finger movements, the cursor can be placed at any location on the display with minimal clutching. The function is a sigmoid curve with $v$ representing the speed of the user's finger:

$$\max\left(L \cdot \text{clamp}\left(\frac{v}{v_1}\right)\left(\frac{1}{1+e^{-k(v-v_0)}}\right)^{T}, x\right) \tag{1}$$

The clamp function restricts its input to the range $[0, 1]$. The parameters used were $L = 240$, $k = 0.2$, $T = 0.8$, $x_0 = 20$, $x_1 = 4$.

The cursor itself is represented simply as a cross-hair. The cross-hair is made slightly larger as it moves away from the position where it was originally created, making it easier to see when the user is interacting with distant objects. Furthermore, we draw a circle filled with a semi-translucent colour around the cross-hair which also increases in size as the cursor moves further from its initial position. This halo is several times larger than the cross-hair, and makes it much easier to find the cursor if the user loses control. The position of the cursor is not constrained to the onscreen area, and a simple triangle pointing in the direction of the cursor off screen is drawn if it is too close to the border. Unlike the HybridPointing technique, we expect the input state to always be immediately obvious to users; absolute input only occurs when the user has no fingers against the screen, and no cursor is drawn, while relative input occurs only when the user has at least one finger against the screen and a cursor is visible.

### 3.2 Drag

Many touch interfaces provide an input method for interacting with conceptual surfaces that are larger than the screen itself. By touching and dragging on the screen, the surface is panned (or "scrolled") horizontally and vertically to allow users to view and interact with objects that were previously outside the displayed area.

We implemented the same style of interaction for a large display. The user places two or more fingers on the screen to initiate a drag, and continuing to move the fingers causes the screen to translate (Figure 2). Dragging stops when all fingers are lifted. Compared to CursorTap, this can

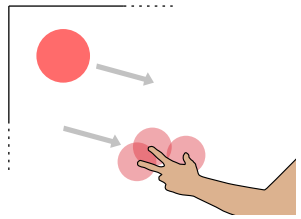

Fig. 2. Drag Technique. The pink areas illustrate the individual dead zones. The red target moves in the direction of the user's fingers, as does a visual representation of the logical border of the screen.

enable a more literal manifestation of the kinematic chain model. Like Guiard's original experiments, dragging the display with one hand manipulates the location of content for the other hand to perform actions on. In practice, the user can choose to drag with one hand and tap with the another, or use a single hand to drag and tap sequentially.

To reduce fatigue when dragging distant content on a very large display, the movement velocity is modified using the exact same acceleration function as CursorTap (Eqn. 1), with the input being the average velocity of the user's fingers. Without this velocity re-scaling, the user would repeatedly clutch to drag the far end of the screen all the way to their position.Like the CursorTap method, we create dead zones around the fingers used to drag the screen. These dead zones are visible to the user, with one dead zone for each finger, to support dragging with more than one hand. Any touch within any of these dead zones will not register as a tap on a target. A visual representation of the logical border of the screen is drawn, so the user is aware of how far the display area has been dragged.

This simple method also has the four desirable design qualities as our method, though it makes trade-offs in how they are achieved. Panning is a *target independent* and *minimal* method that *preserves full spatial context* and *requires only basic sensing*. It is likely more intuitive than CursorTap since it is a common operation used in current touch interfaces, and since all content is dragged to the user, there is no issue viewing the targets and simple direct touch is used for selection. However, dragging poses an issue for multi-user environments in large displays; translating the entire screen will invariably disrupt any other active users of the screen.

### 3.3 Large Display System

We used a 413 × 117 cm wall-sized touch screen made of eight 1920 × 1080 px screens each 103 × 58 cm, arranged in a matrix of four columns and two rows. The entire matrix of screens is covered with a sheet of clear acrylic and bordered by a PQ Labs commercial optical touch frame for detecting input. There are visible 22 mm wide interior bezels between the tiled displays, but the acrylic sheet and single touch frame means that input is not affected. Our system includes bezel areas as touch input. The techniques and both experiments that follow were implemented using the Java-based Processing framework[1].

*3.3.1 Touch Sensor Filtering.* An optical touch frame of this size can introduce spurious touch points and position noise. These occur most often when there are several nearby touches, as is the case with the multi-finger quasimodes used in the CursorTap and Drag techniques. Spurious finger lifts and movements make a quasimode based on the number of finger contacts unreliable. To compensate, we developed our a filtering layer when handling raw touch input events form the touch frame driver. For consistency, this filtering layer was active for all interaction techniques. The filtering method is described in Appendix A.

[1]Source code available at https://github.com/exii-uw/hybrid-touch

## 4 EXPERIMENT 1: CONTROLLED EXPERIMENT

The goal of our first experiment is to compare the performance of CursorTap to straightforward absolute touch input and the Drag relative technique. The experiment design is similar to the comparison used by Forlines et al. [11].

### 4.1 Participants

We recruited 24 participants for the study. The participants consisted largely of university students from a broad selection of majors, 10 female and 14 male. Participants received $15, the experiment was 1 hour on average.

### 4.2 Task

We designed a standard 2D target acquisition task, which required participants to point and click on a series of targets positioned around the screen. The targets were drawn as red circles on a black background. When participants successfully clicked a target, it would turn white and vanish and another target would appear elsewhere on the screen. When the participant missed a target, the event was logged and the target glowed red. Participants had to select each target before continuing in order to prevent them from "racing" through the experiment carelessly. A line to the next target was shown briefly to make it easier to find the target when it appeared. If the Drag technique caused a target to no longer be visible on-screen, a red arrow pointing to the off-screen position of the target was drawn.

In order to maximize the use of our screen space, we required participants to stand at one side of the screen when using relative input techniques. The standing zone was marked by two parallel lines 1.03m apart on the floor (Figure 3). Participants were told to keep their feet within the assigned zone, but could stretch their body outside if they wished. This simulates a larger touch screen, where a user stands at a primary location to perform local manipulation, but uses relative input to select targets far away.

The targets themselves were arranged at the corners of a rectangle, with target presentation order starting at the upper-left then moving down, up, down, right, left, right, up, down, up, left, right, left, arriving back at the starting position. This pattern and the rectangle size created four target distances as an independent variable (see Figure 3 and "Design" below). The rectangle sizes also prevent any target from appearing across bezels between individual screens: note previous work has shown the effect of bezels is minimal [35].

Participants were informed that the targets would follow this pattern. The right and left movements of the target were reversed when the participant began on the right side of the screen, with the target beginning in the upper-right corner, such that the first four targets were always in front of the user and within reach. The corners of the rectangle that did not currently house the active target were drawn using a faint grey, so that participants would be able to quickly anticipate the next position of the target.

The motivation behind this design is to reduce the amount of time spent by participants attempting to find the next target, which would impact the measurement of pure targeting performance. Thus, our design differs from that of Forlines et al., which did not give an indication of where new targets would appear, and positioned targets randomly. Furthermore, in our experiments, only purely horizontal and vertical target movements were tested, as the aspect ratio of the apparatus was very wide, so that near 45° diagonal movements would still be achievable without resorting to distance interaction techniques.

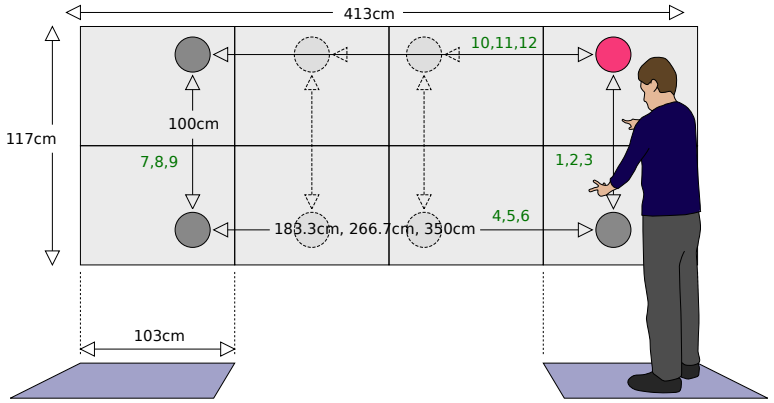


Fig. 3. Task setup showing target positions for the three long distances and selection pattern starting from upper left; the lower two light-blue squares represent the two standing zones on both sides of the display.

### 4.3 Baseline Conditions and Research Questions

While Forlines et al. compared the performance of the hybrid method against both absolute and relative input methods, the use of our cursor method in a purely relative trial is not particularly relevant, as the technique is designed to be used alongside absolute input in ordinary use. Instead, we compared to two baselines: ABSOLUTE, where participants directly tap targets to select them and were not required to stand inside one of the zones; and DRAG, the other relative input baseline technique described above.

These baselines enable us to test the theoretical costs and benefits of CursorTap. ABSOLUTE is more intuitive and simple, and by definition, the target must be within arms-reach at time of selection. This should make it fast and accurate for targets that are already near, but the user has to move to distant targets which takes additional time and could reduce accuracy since body stability may be reduced when rushing or reaching. DRAG is essentially an inverted CursorTap without the cursor: the same transfer function is used, but instead of sending a cursor to a target, the user drags the target to their position. Final target selection should therefore have all the advantages of absolute, perhaps with higher accuracy since the user never has to reach and their body remains stable. The potential issue compared to CursorTap is that DRAG is likely visually disruptive, and it would not be compatible with multiple users.

These baselines enable us to answer key research questions:

*RQ1: Is the additional cognitive and physical overhead of CursorTap a reasonable trade-off compared to physically moving to the target with ABSOLUTE?* A key advantage with CursorTap is that releasing the mode instantly returns to absolute mode to select a nearby target. This is essentially "beating Fitts' law" [3] since the physical distance from the distant target to the nearby one is covered instantly. Our task design includes reciprocal distant and near target selections separated by large distances, so we should see some amortized savings.

*RQ2: Does the reduced visibility when selecting distant targets with CursorTap result in very high error rates compared to DRAG or ABSOLUTE?* With DRAG, the target is always near and easily seen at time of selection, so it should have the lowest error rates. With ABSOLUTE, the user will have to reach for and move to some targets, which may increase error rates due to instability, and it is unclear which has the greater cost.

### 4.4 Design

We used a repeated-measures design with within-subject factors, as opposed to the between-subjects factor of input mapping used in the HybridPointing study. Our independent variables were METHOD (ABSOLUTE, DRAG, CURSORTAP), target WIDTH (5.1 cm, 10.2 cm) and target DISTANCE (100 cm, 183.3 cm, 266.7 cm, 350 cm). The combination of width and distance creates IDs ranging from 3.4 to 5.9.

Forlines et al. had a target distance of 400 cm, but this was larger than our display, so we distributed target distances equally from 100 cm to 350 cm. Note that the "short" vertical target distance of 100 cm occurs more often since it is a transition between the three primary "long" horizontal distances.

Each participant performed 6 blocks of target selections with each METHOD. 12 timed trials were made for each combination of target WIDTH and DISTANCE, aside from the shortest target distance.

Half the participants started on the right zone of the display (see Figure 3) the other half on the left. When standing in the left zone, the left side of the rectangle always coincided with the middle of the leftmost column of monitors. The right side corresponded to one of the three distances for the remote targets. The conditions were mirrored when participants were standing on the right side. People switched sides for each block. Combined with the six possible permutations of the order of *input method*, this resulted in twelve total conditions across the 24 participants. Participants were permitted to take breaks in between each block.

The target widths and rectangle sizes that determine target distances were presented in a random ordering. This was kept consistent between participants, such that the first block of each trial always used the same ordering of widths and distances, regardless of which interaction method was used first.

In summary: 3 METHODS × 6 blocks × 2 WIDTHS × (3 long DISTANCES × 6 selection + 18 short DISTANCE selections ) = 1296 selections per participant.

### 4.5 Results

Repeated measures ANOVA were conducted for all measures[2] and when significant effects were discovered, post hoc pairwise t-tests with Holm correction were performed. Because measures for preference exhibited non-normality, they were transformed using Aligned Rank Transform [37]. Trials were aggregated by participant and analyzed factor. Time data were aggregated using the median to account for skewed distribution.

For each participant, target times more than 3 standard deviations from the mean of the participant's time were excluded as outliers[3]. In total, 405 trials (1.3%) were removed.

*4.5.1 Learning Effect.* Overall, BLOCK had a main effect on selection time ($F_{5,115} = 9.41, p < .0001$, $\eta_p^2 = .045$), and on error rate ($F_{5,115} = 2.92, p = .015, \eta_p^2 = .048$). For time, post hoc tests found blocks 1, 2 were significantly slower than blocks 3 to 6 (all $p < .02$), suggesting an initial learning effect. For error, the only difference was between block 2 and 5, but combined block error rates are all below 4%. In subsequent analysis, we use only blocks 3 to 6 as more representative of practised performance. This is similar to Forlines et al., who removed the first 4 blocks to account for learning.

*4.5.2 Error Rate.* If a target was missed one or more times, the trial was marked as an error. High velocity movements occasionally registered false taps as errors without affecting user behaviour in that trial. We corrected for 1,649 of these by identifying taps more than one target diameter away from the outer edge of the target. The error rate is the ratio of error trials to all trials.

[2]When the assumption of sphericity was violated, we corrected the degrees of freedom using Greenhouse-Geisser (Greenhouse-Geisser's $\epsilon < 0.75$) or Huynh-Feldt (Greenhouse-Geisser's $\epsilon \geq 0.75$).

[3]A standard outlier removal method in HCI pointing studies, for example Forlines et al. also used this method.

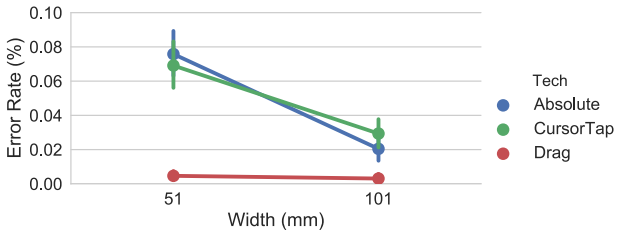


Fig. 4. Error rate for TECHNIQUE by WIDTH (error bars 95% CI).

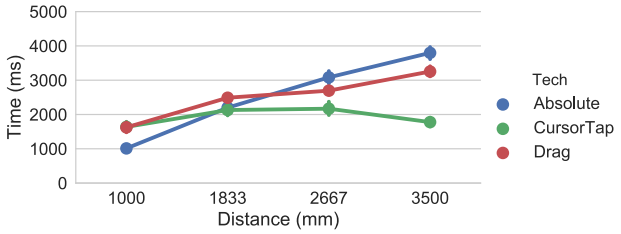


Fig. 5. Target time for TECHNIQUE by DISTANCE (error bars 95% CI).

There was a main effect of TECHNIQUE on error rate ($F_{2,46} = 31.45, p < .0001, \eta_p^2 = 0.41$). Post hoc tests show the error rate for DRAG (0.5%) is lower than both ABSOLUTE (4.7%) and CURSORTAP (5.4%) (all $p < .0001$). The main effect provides a high-level summary, but the effect of technique on width is most relevant. There was an interaction between TECHNIQUE and WIDTH on error rate ($F_{2,46} = 19.83, p = .0001, \eta_p^2 = 0.12$) (see Figure 4). For CURSORTAP and ABSOLUTE, rates increased from under 4% for the larger width to nearly 8% for the smaller width (both $p < .0001$), but there was no significant difference across widths for DRAG. There was no significant difference between CURSORTAP and ABSOLUTE at either width.

These represent acceptable error rates, but one may consider 8% somewhat high for 51mm targets. We think this may be due to some difficulty judging distant cursor and target positions with CURSORTAP because of the reduced viewing angle and reaching and tapping at the end of a large physical body motion with ABSOLUTE. The rate for DRAG is remarkably low. We believe the benefit of always bringing the target directly in front of the body is the reason. Note that Forlines et al. observed overall similar error rates between 4.2% and 6.8%, but no breakdown by width was reported.

*4.5.3 Selection Time.* Overall, when all tested conditions are combined, TECHNIQUE had a significant main effect on time ($F_{2,46} = 24.45, p = .0001, \eta_p^2 = 0.31$) with post hoc tests showing CURSORTAP was 500 ms faster than ABSOLUTE and DRAG (both $p < .0001$). This is an encouraging general result for CursorTap performance, but a significant interaction of TECHNIQUE and DISTANCE provides more detail ($F_{3.78,87.03} = 73.99, p = .0001, \eta_p^2 = 0.46$) (see Figure 5). The variance of these measures is quite small: note the very small 95% CI error bars in this graph.

At the shortest DISTANCE of 1000mm, the time for ABSOLUTE (1011ms) is significantly lower than both CURSORTAP (1638ms) and DRAG (1624ms) (all $p < .0001$). But at a DISTANCE of 1833mm, times for ABSOLUTE and CURSORTAP are no longer significantly different (both near 2200ms) and DRAG is slightly, but significantly, slower (2488ms) (all $p < .01$). At the second longest DISTANCE of 2667mm, the time for CURSORTAP (2169ms) becomes significantly lower than both ABSOLUTE (3082ms) and DRAG (2695ms) (all $p < .0001$). Finally, the time for CURSORTAP (1777ms) diverges even more at the

longest distance of 3500mm, becoming approximately half of the times for ABSOLUTE (3797ms) and DRAG (3253ms).

The results show CURSORTAP times remain nearly constant as the distance increases, but both ABSOLUTE and DRAG increase with distance (Figure 5). An approximately linear increase for ABSOLUTE was also noted in Forlines et al., but the nearly flat response of CURSORTAP suggests a stronger benefit for touch hybrid pointing than pen hybrid pointing.

*4.5.4 Subjective Ratings.* At the end of the experiment, participants were asked to consider accuracy, speed, and fatigue and provide a numerical rating for each technique (Figure 1). Ratings used a real numbered, continuous scale from 1 (most preferred) to 7 (least preferred). Decimal ratings such as 4.5 were permitted.

Table 1. Average subject ratings (with standard error of the mean). Scale is 1 to 7, lower scores are better.

| | Fatigue | Speed | Accuracy | Mean |
|---|---|---|---|---|
| ABSOLUTE | 4.6 ±0.36 | 4.0 ±0.36 | 2.3 ±0.37 | 3.6 ±0.24 |
| DRAG | 2.3 ±0.21 | 2.7 ±0.34 | 2.6 ±0.30 | 2.5 ±0.16 |
| CURSORTAP | 3.5 ±0.38 | 4.3 ±0.30 | 4.7 ±0.32 | 4.2 ±0.20 |
| Mean | 3.2 ±0.22 | 3.5 ±0.21 | 3.7 ±0.23 | 3.5 ±0.12 |

Differences between techniques across all metrics were significant (*all* $p < .005$). For fatigue, DRAG was rated as least fatiguing, CURSORTAP second, and ABSOLUTE most fatiguing. In terms of accuracy and speed, both ABSOLUTE and DRAG were rated higher than CURSORTAP. So although there was some subjective preference for CursorTap over absolute pointing, CursorTap was not rated overall as faster or comparably accurate. When asked about the ratings, several participants explained that the fatigue was more mental than physical with CursorTap due to the novelty of the method. Regarding speed, participants commented that although CursorTap felt fast for multiple distant targets, the extra overhead to send the cursor initially out to a distant target felt slower than simply walking.

### 4.6 Experiment Discussion

When selecting medium and distant targets with CursorTap, there is an overall time advantage, with only a modest increase in error, compared to Drag and Absolute baseline techniques. Regarding research question RQ1, our results suggest the amortized time overhead to use CursorTap is less than the overhead required to physically move to and from distant targets with pure absolute pointing. However, there was some overhead when selecting the "short" distant targets with both CursorTap and Drag. We believe this may be caused by a moment of hesitation to decide if a relative switch or drag was needed. Regarding RQ2, we did observe very low error rates for Drag. Error rates with CursorTap and using the absolute technique were higher, but similar, suggesting that the cost of reduced visual angle and visual acuity with CursorTap is similar to the cost of fatigue or instability due to physical movement with absolute. There also is an interesting observation concerning only Drag and default absolute pointing. Our results suggest that the time cost for dragging the display to select a target is similar to the time required to physically move to a target. However, participants rated Drag significantly lower for fatigue, suggesting some advantage.

*Examining "flat" CursorTap performance.* The overall time performance for CursorTap is the result of a near instant return from distant targets for a high proportion of trials. After using relative mode to select a distant target, the participant can immediately exit the quasimode by lifting their hand and immediately select a nearby target using direct touch in absolute mode. Since target acquisition distances are measured between two successive targets and not between the user and

the next target, the immediate return allows CursorTap to virtually cover a long distance in an instant and "beat Fitts' law", as hypothesized in RQ1.

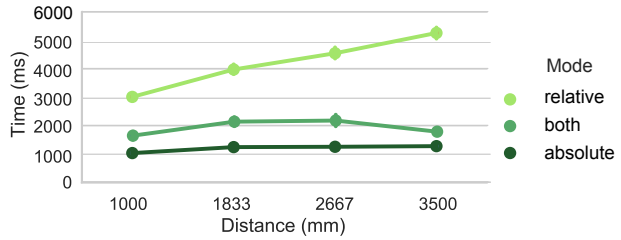


Fig. 6. Target time for CURSORTAP by DISTANCE and pointing mode used (error bars 95% CI).

Table 2. CURSORTAP trials by pointing mode by DISTANCE: error free trials for blocks 3 - 6 as percentage with counts in parenthesis.

| | DISTANCE (mm) | | | | |
|---|---|---|---|---|---|
| *mode* | 1000 | 1833 | 2667 | 3500 | *all* |
| relative | 49% (1575) | 49% (545) | 49% (534) | 46% (476) | 49% (3130) |
| absolute | 51% (1645) | 51% (558) | 51% (559) | 54% (561) | 51% (3323) |

Figure 6 and Table 2 help explain the resulting overall "flat" pattern for CursorTap time across distances. Figure 6 shows that selection times using *relative mode* steadily increase with distance, as predicted by Fitts' law[4], but selection time in *absolute mode* remains nearly constant, since many selections are for targets already near to the user. The reason why the overall selection time for CursorTap is flat (*both modes* in Figure 6) is due to how the data is aggregated. Table 2 shows the proportion of trials in each mode by distance. There are proportionally more trials in absolute mode with greater target-to-target distance due to more error trials being removed. Since the median time is used to aggregate trials (by participant, width, distant, and block) the greater proportion of faster absolute mode selections dominates, pulling the amortized time closer to the absolute time.

*Actual Usage and User Preference.* The asymmetric performance when selecting distant targets can apply to tasks where the user wishes to click on a distant button and return back to their primary work area. In theory, using CursorTap saves overall time for this kind of distant-and-back sequence, even though the outward portion of the task incurs additional time. However, this overall potential savings was not enough to positively influence subjective impressions. Participants had a stronger preference for the Drag technique compared to CursorTap, perhaps due to its lower error rate. However, dragging the entire display content quickly becomes problematic in a multi-user setting due to the momentary disruption for other users. To investigate subjective preferences further, we designed a more open-ended study in which we could observe if CursorTap would be willingly used in a less structured task. We do this with multiple people on a shared display, to test if CursorTap, unlike Drag, is indeed compatible with this situation.

## 5 EXPERIMENT 2: OPEN-ENDED TASK

The goal of the second experiment is to explore if people choose to use CursorTap in a more open-ended task when sharing a large display with another person. We use a game setting to create frequent opportunities for potential technique usage and to control for cooperative and competitive behaviour between the two players. Our game enables us to test a range of interactions that include

[4]Additional Fitts' modelling of relative trials in Section 5.4 of Dickson's Master's thesis [10].

object dragging and lasso selection, not only target selection as in experiment 1. We discuss the generalizability and limitations for using a game context below. The same apparatus as experiment 1 was used with additional software written to run the game. Note this apparatus and game was also used in Wehbe et al.'s study [36], where the social mechanisms people use to establish and maintain physical and digital territories when collaborating or competing are examined. In contrast, our study focuses entirely on usage patterns of the CursorTap technique.

### 5.1 Participants

We recruited 14 participants, mainly university students, but with a broad range of majors. They participated in pairs, consisting of strangers, acquaintances, friends, and couples. Note 4 people also participated in experiment 1, but the two studies were conducted several months apart. Remuneration was $10 for what was on average a one hour session.

### 5.2 Game Task

The game, illustrated in Figure 7 and demonstrated in the accompanying video, works as follows: Each player is assigned one side of the display and is not permitted to touch inside the other player's side. The goal is to defend planet Earth (located at the centre of the display) from waves of "enemies" that move towards the Earth starting from the left and right edges of the display. Players eliminate enemies using three different kinds of "tools", each activated at a defined spot in a central "neutral zone" column. Three colours are used for enemies and these tools: each tool can only eliminate enemies with a matching colour. The tools are shared between players, meaning each tool can be activated by only one player at a time.

The game uses a scoring system without any victory or loss condition. One point is awarded for each eliminated enemy, but one point is deducted for each enemy that reaches the earth. The game can work in two ways: as a cooperative game where players have a single shared score; and as a competitive game where each player has their own score.

*5.2.1 Tools Representing Different Interactions.* A player activates a tool by dragging it off of a defined location in the central neutral zone. Each tool activation area is a 150 mm diameter circle. Importantly, the three tools used to eliminate enemies may all be controlled using either absolute or relative input. This means the player can choose to use default absolute input (i.e. *directly* tapping or dragging the tool with a finger) or the relative input quasimode (i.e. *indirectly* tapping or dragging to control a remote tool like a cursor). These tools are designed to represent common types of direct manipulations:

- A green "shield" tool represents *dragging*. The player drags the tool to a location and any green enemy that collides with the tool is eliminated. Dragging is performed directly with a finger in absolute input mode or with the tool behaving like a cursor in relative mode.
- A red "cannon" tool represents *tapping*. The player drags the tool to a location and then taps to eliminate any blue enemy that is near the tool. The tap is performed on the tool itself in absolute mode or at a remote location when in relative mode.
- A blue "black hole" tool represents *lasso selection*. A blue trail is left behind the tool as it is dragged. When the trail intersects itself, any blue enemies within the enclosed area are eliminated. The tool is either dragged with the finger in absolute input mode or like a cursor in relative mode.

The game also included a special "magnet" tool to steal the tool currently used by the other player. This was only controlled using absolute input, so it is not a focus of our analysis.

*5.2.2 Enemies as Interaction Targets.* Enemies appeared in waves, approaching the central Earth from random positions starting along the left and right sides. Each enemy was 40 mm in diameter

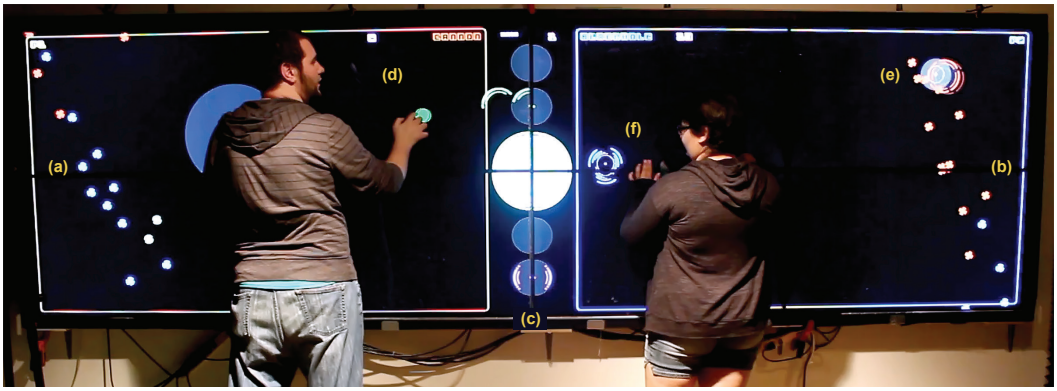


Fig. 7. Game elements and layout: (a) wave of enemies approaching the Earth from left; (b) wave of enemies approaching the Earth from right; (c) centre "neutral zone" column containing the three enemy elimination tools, the magnet too, and the planet Earth at centre; (d) player 1 using CursorTap to control the red *tapping* tool with relative input; (e) player 1's tool at a distant location on the display; (f) player 2 about to use absolute input to control the blue "lasso" tool.

and travelled at a constant speed of 28 mm/s. Each side of the wave consisted of 18 enemies: 12 of a primary type, 4 of a secondary type, and 2 of a third type. The primary type of the left side was the secondary type of the right side, and vice-versa. The third type was the same for both sides. This design was meant to encourage participants to change tools, create opportunities for players to use CursorTap's relative mode to eliminate enemies on the other player's side, and to introduce some competition for each tool. Each wave ended when all the enemies on both sides had been either eliminated or had collided with the Earth. The next wave began immediately after.

Note that unlike Experiment 1, the enemies are dynamic targets without controlled distances from initial tool positions. This was necessary to create an open-ended task in the context of a game. The variety of tools and the spatial distribution of enemy targets creates opportunities for a variety of pointing and interaction strategies, which may or may not include using the CursorTap technique.

### 5.3 Design

The gameplay and CursorTap technique were explained to participants at the beginning of the session. There was also a short practice session for them to familiarize themselves with the interactions (less than 5 minutes on average). We chose this structure since the study goal is to examine willingness to use CursorTap, not its discoverability or learnability.

Then, the game was run twice for each pair, once with cooperative scoring and once with competitive scoring. We use this as a CONDITION factor with two levels: COOPERATIVE and COMPETITIVE. The order was counter-balanced across participant pairs.

Within each scoring condition, there were two BLOCKS of play, each with six waves of enemies. Since each wave had 18 enemies on each side, there were 216 potential targets per block, and 864 potential targets per experiment session.

For analysis, we also define two factors: MODE, which is whether default absolute mode, or the CursorTap relative mode is used to eliminate an enemy; and SIDE, which is whether an enemy target was eliminated on the same or opposite side of the display relative to the player.

The primary dependent variables are: *Elimination Time*, the time to eliminate an enemy from the time it first appeared during a wave; and *CursorTap Usage*, the proportion of enemies eliminated when the relative input quasimode was active. In addition, participants rated the accuracy, speed,

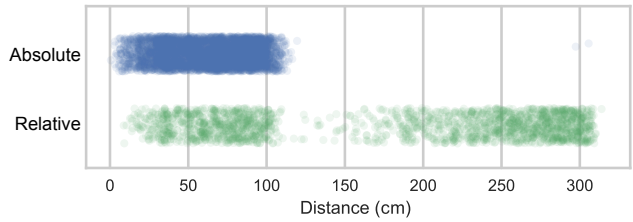


Fig. 8. Density plot showing distance from enemy to centre of player side at moment of elimination.

and fatigue when using default absolute input and the relative input quasimode for each of the three enemy elimination tools.

### 5.4 Results

The same analysis method as experiment 1 was used: repeated measures ANOVA, t-tests with Holm correction, and ART [37] for subjective measures that exhibited non-normality.

*5.4.1 Learning Effect.* We examine median *Elimination Time* calculated per participant per block to establish whether participants improved their gameplay from block 1 to block 2 in each gameplay condition. There was no significant main effect of BLOCK nor any significant interaction involving BLOCK. This suggests participants maintained a consistent level of gameplay throughout the session.

*5.4.2 CursorTap Usage.* Overall, 8 of the 14 participants used the CursorTap relative input quasimode to eliminate enemies at least 5% of the time. The 6 participants who used the quasimode less than 5% of the time said their reluctance was primarily due to the sometimes noisy touch input, which made relative input less stable than absolute input.

Among the 8 participants who used relative input more often, they used it to eliminate 51.5% of enemy targets overall. Broken down by tool type, 41.5% used relative input with the *tapping* tool, 50.6% with the *dragging* tool, and 62.6% with the *lasso* tool. These differences are significant, there is a main effect of TOOL on *CursorTap Usage* ($F_{2,14} = 4.80, p < 0.026, \eta_p^2 = 0.114$) with all post-hoc comparisons significant ($p < .001$).

Participants used relative input more often for distant enemies, but they also used it for enemies that were within arms reach. We found 61.4% of eliminations using relative mode were for enemies located on the other player's side of the display. The remaining 38.6% of eliminations with the relative mode technique were used when the enemy was on the player's own side. Examining eliminations by distance from enemy to centre of player's side, we can see that many eliminations with relative input are well within arms-reach when compared to those eliminated with absolute input (Figure 8). Overall, relative mode was used on the participant's own side for 19.8% of all eliminations.

Finally, we did not find evidence for different relative mode usage between competitive and cooperative gameplay: CONDITION had no significant effect on *CursorTap Usage* ($F_{1,7} = 0.733, p = 0.42, \eta_p^2 = 0.0196$), nor an interaction with enemy SIDE ($F_{1,7} = 0.00018, p = 0.99, \eta_p^2 = 0.0000$). From observation, participants did not seem to demonstrate different behaviours based on the gameplay condition. This is surprising, as one might have expected them to optimize their behaviour for a higher combined score in the cooperative condition.

*5.4.3 Subjective Ratings.* As in experiment 1, preference scores used a real numbered, continuous scale from 1 (most preferred) to 7 (least preferred) with decimal ratings such as 4.5 permitted. We found no strong preference for either input mode when considering ratings for Fatigue, Speed, or

Accuracy, with no statistically significant difference for ratings by tool, or when all tool ratings are combined into an overall mean (see Table 3).

Table 3. Mean subjective ratings (with SEM) after combining ratings for individual tools. Ratings use a 1 to 7 numeric scale with 1 most preferred (best).

| | Fatigue | Speed | Accuracy | **Mean** |
|---|---|---|---|---|
| Absolute | 3.2 ±0.26 | 3.6 ±0.25 | 3.2 ±0.29 | **3.3 ±0.15** |
| Relative | 3.0 ±0.25 | 3.1 ±0.27 | 3.6 ±0.27 | **3.2 ±0.15** |

### 5.5 Study Discussion

Our observations show some people are willing to use the CursorTap hybrid technique to interact, and that those who use the technique use it regularly. We note that participants used the short tutorial time to become comfortable with CursorTap, but once familiarized, some quickly adopted the quasimode gesture to take advantage of relative pointing during gameplay.

Although not all participants took advantage of CursorTap, our finding that a little over half chose to use it in an open-ended task is encouraging. Activating and using the relative input quasimode is not typical for touch interaction, and the game could be played without it. Participants did not specifically mention that they avoided it due to additional cognitive and physical overhead, and the subjective ratings do not suggest a specific reason related to fatigue, speed, or accuracy. Another possible reason was that for some people, the intensity of the game prevented them from experimenting with CursorTap, though this was not noted by participants. The main reason given for not using CursorTap was related to touch sensing errors, which may have affected some participants more than others due to hand sizes, height, or how they moved their body.

Interestingly, participants used the relative input quasimode to eliminate targets that were within their own side, and would have been within reach to use default absolute input. Although absolute touch is faster for tapping on nearby targets (shown in our first experiment), in the context of the game, clearly some participants found relative input to provide other benefits. One possibility is that they were able to move a tool more quickly by exploiting the pointer acceleration function, and this provided an advantage regardless of the enemy position.

No significant differences were found between cooperative and competitive conditions, suggesting a relative input cursor is not distracting for other people. In other words, "digitally reaching" into another user's area appears to be socially acceptable, an aspect of large display use examined in detail by Azad et al. [2] and Wehbe et al. [36].

## 6 DISCUSSION

Absolute and relative cursor control are suited for certain contexts: absolute for close targets and relative pointing for distant targets. CursorTap is a method that can bridge these two usage contexts when selections involve combinations of far and near targets. Our subjective rating results from experiment 1 suggest there is a cognitive overhead when users have to decide whether to make a switch from absolute to relative, or vice versa. But when the decision to directly tap or use a relative cursor is more clear cut, like with exclusive and clearly separated player zones in our open-ended game task, then some people will choose to use CursorTap. We surmise that CursorTap would prove useful in similar multi-user scenarios with personal spaces directly in front of the owner which can be manipulated using classic direct touch input, and with the possibility to occasionally reach to other users' spaces to share content or access a specific tool (similar to our magnet to steal the other player's tool).

One limitation of touch-based methods to extend cursor reach on large displays such as ours is the reduced visibility of distant content. When the user is standing close to the display to be able to directly interact with it, it becomes more difficult to see, and therefore manipulate, content that is located at the far edges of the screen as the viewing angle is relatively low. We observed this effect in the higher error rates in Experiment 1 for CursorTap relative to Drag. We can mitigate that effect by increasing the cursor size the farther it moves from the user and providing local pointers to distant items, but these are not perfect solutions. We are not suggesting that our technique is suitable for extremely large displays, where users would likely prefer to step back in order to have a better view and maybe use remote pointing [34], or a proxy input device, for example, a mobile phone [20, 22].

### 6.1 Possible Extensions

A promising extension of the CursorTap method that could warrant deeper investigation is a hybrid of absolute and cursor-based input, where users can use the cursor to select a broad group of controls, creating a local copy of those controls at their current position. This might be analogous to how WallPad [12] and Frisbee [16] focus on bringing a copy of remote content to the user. This target could be made much larger, and would therefore be significantly easier to select, allowing for users to interact with the controls inside the group more comfortably with absolute input. This could conceivably combine the speed and multi-user benefits of the long-distance hybrid method with the accuracy and familiarity of interacting with more complex controls using direct touch input directly in front of them.

Another possible extension is to support other cursor-based tools or operations by using more than one finger of the moving hand. CursorTap uses three fingers to set the mode with one hand and a single finger of the other hand to move a cursor to perform clicks or selections, but one could imagine the other hand being able to further control, for example, a moving magic lens [18] with two fingers and a selection rectangle with three fingers. However, dragging with additional fingers may be more difficult if there is friction with the screen. Alternative methods to switch between relative and absolute methods are also conceivable, for instance using hand-contact shape or pressure instead of three fingers, if supported by the touch sensor, or single-hand techniques using differentiated multitouch gestures.

## 7 CONCLUSION

We presented CursorTap, a bimanual, multitouch HybridPointing technique for large displays that supports absolute and relative pointing. Our combined results show that CursorTap is effective for reaching distant targets, and outperforms standard absolute touch input or dragging the entire display window, for medium and far targets. Although our participants did not report a strong subjective preference, we showed in our second study that a good proportion of people freely choose to use our technique even if it is not strictly necessary. Few interaction technique papers move beyond a controlled experiment, and attempt to investigate usage in an open-ended choice-based task. We believe the results of our second study strengthens our methodology by validating the potential for CursorTap in real applications, not necessarily only games. Based on our results, we believe CursorTap can be effectively used in large-display contexts where both spatially close and distant access to content is required, for example, collaborative applications for visualization and data exploration.

## ACKNOWLEDGMENTS

This work made possible by: NSERC Discovery Grant 2018-05187; Canada Foundation for Innovation Infrastructure Fund 33151 “Facility for Fully Interactive Physio-digital Spaces”; and Ontario Early

Researcher Award ER16-12-184. We wish to acknowledge our ISS reviewers who provided critical feedback that motivated significant improvements to this paper.

## REFERENCES


[1] Martha Abednego, Joong-Ho Lee, Won Moon, and Ji-Hyung Park. 2009. I-Grabber: Expanding Physical Reach in a Large-Display Tabletop Environment through the Use of a Virtual Grabber. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces* (Banff, Alberta, Canada) *(ITS '09)*. Association for Computing Machinery, New York, NY, USA, 61–64. https://doi.org/10.1145/1731903.1731917

[2] Alec Azad, Jaime Ruiz, Daniel Vogel, Mark Hancock, and Edward Lank. 2012. Territoriality and Behaviour on and Around Large Vertical Publicly-shared Displays. In *Proceedings of the Designing Interactive Systems Conference* (Newcastle Upon Tyne, United Kingdom) *(DIS '12)*. ACM, New York, NY, USA, 468–477. https://doi.org/10.1145/2317956.2318025

[3] Ravin Balakrishnan. 2004. "Beating" Fitts' law: virtual enhancements for pointing facilitation. *International Journal of Human-Computer Studies* 61, 6 (2004), 857–874.

[4] Amartya Banerjee, Jesse Burstyn, Audrey Girouard, and Roel Vertegaal. 2011. Pointable: An In-air Pointing Technique to Manipulate Out-of-reach Targets on Tabletops. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces* (Kobe, Japan) *(ITS '11)*. ACM, New York, NY, USA, 11–20. https://doi.org/10.1145/2076354.2076357

[5] Patrick Baudisch, Edward Cutrell, Dan Robbins, Mary Czerwinski, Peter Tandler, Benjamin Bederson, Alex Zierlinger, and others. 2003. Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch-and pen-operated systems. In *Proceedings of INTERACT*, Vol. 3. 57–64.

[6] Anastasia Bezerianos and Ravin Balakrishnan. 2005. *The Vacuum: Facilitating the Manipulation of Distant Objects*. Association for Computing Machinery, New York, NY, USA, 361–370. https://doi.org/10.1145/1054972.1055023

[7] Géry Casiez and Nicolas Roussel. 2011. No More Bricolage! Methods and Tools to Characterize, Replicate and Compare Pointing Transfer Functions. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 603–614. https://doi.org/10.1145/2047196.2047276

[8] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1 € Filter: A Simple Speed-based Low-pass Filter for Noisy Input in Interactive Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. ACM, New York, NY, USA, 2527–2530. https://doi.org/10.1145/2207676.2208639

[9] Maxime Collomb, Mountaz Hascoët, Patrick Baudisch, and Brian Lee. 2005. Improving Drag-and-Drop on Wall-Size Displays. In *Proceedings of Graphics Interface 2005* (Victoria, British Columbia) *(GI '05)*. Canadian Human-Computer Communications Society, Waterloo, CAN, 25–32.

[10] Terence Dickson. 2017. *HybridPointing Touch: A Technique to Switch Between Absolute and Relative Pointing on Large Touch Screens*. Master's thesis. University of Waterloo, Canada. http://hdl.handle.net/10012/12593

[11] Clifton Forlines, Daniel Vogel, and Ravin Balakrishnan. 2006. 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* (Montreux, Switzerland) *(UIST '06)*. Association for Computing Machinery, New York, NY, USA, 211–220. https://doi.org/10.1145/1166253.1166286

[12] Jérémie Gilliot, Géry Casiez, and Nicolas Roussel. 2014. Direct and Indirect Multi-touch Interaction on a Wall Display. In *Proceedings of the 26th Conference on L'Interaction Homme-Machine* (Villeneuve d'Ascq, France) *(IHM '14)*. ACM, New York, NY, USA, 147–152. https://doi.org/10.1145/2670444.2670445

[13] Yves Guiard. 1987. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of motor behavior* 19, 4 (1987), 486–517.

[14] Kaori Ikematsu and Itiro Siio. 2016. Carbon Copy Metaphor: Mode Switching Technique for Trackpad-Based Manipulations. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces* (Niagara Falls, Ontario, Canada) *(ISS '16)*. Association for Computing Machinery, New York, NY, USA, 439–444. https://doi.org/10.1145/2992154.2996795

[15] Mikkel R. Jakobsen, Yvonne Jansen, Sebastian Boring, and Kasper Hornbæk. 2015. *Should I Stay or Should I Go? Selecting Between Touch and Mid-Air Gestures for Large-Display Interaction*. Springer International Publishing, Cham, 455–473. https://doi.org/10.1007/978-3-319-22698-9_31

[16] Azam Khan, George Fitzmaurice, Don Almeida, Nicolas Burtnyk, and Gordon Kurtenbach. 2004. A Remote Control Interface for Large Displays. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology* (Santa Fe, NM, USA) *(UIST '04)*. ACM, New York, NY, USA, 127–136. https://doi.org/10.1145/1029632.1029655

[17] Sunjun Kim, Jihyun Yu, and Geehyuk Lee. 2012. Interaction techniques for unreachable objects on the touchscreen. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*. ACM, 295–298. http://dl.acm.org/citation.cfm?id=2414585

[18] Ulrike Kister, Patrick Reipschläger, and Raimund Dachselt. 2014. Multi-Touch Manipulation of Magic Lenses for Information Visualization. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces* (Dresden, Germany) *(ITS '14)*. Association for Computing Machinery, New York, NY, USA, 431–434. https://doi.org/10.1145/2669485.2669528

[19] Mingyu Liu, Mathieu Nancel, and Daniel Vogel. 2015. Gunslinger: Subtle arms-down mid-air interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, 63–71. http://dl.acm.org/citation.cfm?id=2807489

[20] David C. McCallum and Pourang Irani. 2009. 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* (Victoria, BC, Canada) *(UIST '09)*. Association for Computing Machinery, New York, NY, USA, 153–156. https://doi.org/10.1145/1622176.1622205

[21] Miguel A. Nacenta, Dzmitry Aliakseyeu, Sriram Subramanian, and Carl Gutwin. 2005. *A Comparison of Techniques for Multi-Display Reaching*. Association for Computing Machinery, New York, NY, USA, 371–380. https://doi.org/10.1145/1054972.1055024

[22] Mathieu Nancel, Olivier Chapuis, Emmanuel Pietriga, Xing-Dong Yang, Pourang P. Irani, and Michel Beaudouin-Lafon. 2013. *High-Precision Pointing on Large Wall Displays Using Small Handheld Devices*. Association for Computing Machinery, New York, NY, USA, 831–840. https://doi.org/10.1145/2470654.2470773

[23] Mathieu Nancel, Emmanuel Pietriga, Olivier Chapuis, and Michel Beaudouin-Lafon. 2015. Mid-air pointing on ultra-walls. *ACM Transactions on Computer-Human Interaction (TOCHI)* 22, 5 (2015), 21.

[24] J. Karen Parker, Regan L. Mandryk, and Kori M. Inkpen. 2005. TractorBeam: seamless integration of local and remote pointing for tabletop displays. In *Proceedings of Graphics interface 2005*. Canadian Human-Computer Communications Society, 33–40. http://dl.acm.org/citation.cfm?id=1089515

[25] Ken Pfeuffer, Jason Alexander, and Hans Gellersen. 2016. *Partially-Indirect Bimanual Input with Gaze, Pen, and Touch for Pan, Zoom, and Ink Interaction*. Association for Computing Machinery, New York, NY, USA, 2845–2856. https://doi.org/10.1145/2858036.2858201

[26] Jef Raskin. 2000. *The humane interface: new directions for designing interactive systems*. Addison-Wesley Professional.

[27] Hanae Rateau, Yosra Rekik, Laurent Grisoni, and Joaquim Jorge. 2016. Talaria: Continuous Drag & Drop on a Wall Display. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces* (Niagara Falls, Ontario, Canada) *(ISS '16)*. Association for Computing Machinery, New York, NY, USA, 199–204. https://doi.org/10.1145/2992154.2992164

[28] Adrian Reetz, Carl Gutwin, Tadeusz Stach, Miguel Nacenta, and Sriram Subramanian. 2006. Superflick: a natural and efficient technique for long-distance object placement on digital tables. In *Proceedings of Graphics interface 2006*. Canadian Information Processing Society, 163–170. http://dl.acm.org/citation.cfm?id=1143106

[29] Abigail J Sellen, Gordon P Kurtenbach, and William AS Buxton. 1992. The prevention of mode errors through sensory feedback. *Human-computer interaction* 7, 2 (1992), 141–164.

[30] Ben Shneiderman. 1991. Touch screens now offer compelling uses. *IEEE software* 8, 2 (1991), 93–94.

[31] Garth Shoemaker, Anthony Tang, and Kellogg S. Booth. 2007. Shadow Reaching: A New Perspective on Interaction for Large Displays. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA) *(UIST '07)*. ACM, New York, NY, USA, 53–56. https://doi.org/10.1145/1294211.1294221

[32] Garth Shoemaker, Takayuki Tsukitani, Yoshifumi Kitamura, and Kellogg S. Booth. 2010. Body-Centric Interaction Techniques for Very Large Wall Displays. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries* (Reykjavik, Iceland) *(NordiCHI '10)*. Association for Computing Machinery, New York, NY, USA, 463–472. https://doi.org/10.1145/1868914.1868967

[33] Qingkun Su, Oscar Kin-Chung Au, Pengfei Xu, Hongbo Fu, and Chiew-Lan Tai. 2016. 2D-Dragger: unified touch-based target acquisition with constant effective width. ACM Press, 170–179. https://doi.org/10.1145/2935334.2935339

[34] Daniel Vogel and Ravin Balakrishnan. 2005. Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays *(UIST '05)*. Association for Computing Machinery, New York, NY, USA, 33–42. https://doi.org/10.1145/1095034.1095041

[35] James R Wallace, Daniel Vogel, and Edward Lank. 2014. The effect of interior bezel presence and width on magnitude judgement. In *Proceedings of Graphics Interface 2014*. Canadian Information Processing Society, 175–182.

[36] Rina R. Wehbe, Terence Dickson, Anastasia Kuzminykh, Lennart E. Nacke, and Edward Lank. 2020. *Personal Space in Play: Physical and Digital Boundaries in Large-Display Cooperative and Competitive Games*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376319

[37] Jacob O. Wobbrock, Leah Findlater, Darren Gergle, and James J. Higgins. 2011. The Aligned Rank Transform for Nonparametric Factorial Analyses Using Only Anova Procedures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. 143–146.

[38] Haijun Xia, Ken Hinckley, Michel Pahud, Xiao Tu, and Bill Buxton. 2017. *WritLarge: Ink Unleashed by Unified Scope, Action, & Zoom.* Association for Computing Machinery, New York, NY, USA, 3227–3240. https://doi.org/10.1145/3025453.3025664

[39] Jibin Yin and Hua Liu. 2010. Lever Cursor: A bimanual selection technique for multitouch tabletop. In *The 2010 IEEE International Conference on Information and Automation.* 105–109. https://doi.org/10.1109/ICINFA.2010.5512341

## APPENDIX A: TOUCH FILTERING LAYER

For replication and information we provide details for the additional touch filtering we implemented in our system. Filtering was performed by introducing an intermediate layer between the driver's reported touch events and the touch events that were presented to the applications built using our system. This separate layer presents its own touch data structures, allowing full control over the position and number of active touches.

The filtering layer works in multiple stages:

1. *All touches coming from the driver that are too close to the edge of the screen are ignored.* Targets in the task are intentionally not placed near the edge of the screen, and the sensors occasionally detected spurious touches stuck to the edges. The radius used was 64 pixels, or 3.5 cm.
2. *All new touches coming from the driver are not presented to the program for a short period of time.* If a touch from the driver is lifted within 20 ms, it is considered a spurious touch and is discarded without ever being presented to the rest of the program. If the touch is within 64 pixels, or 3.5cm, of an existing non-spurious touch, the touch is "absorbed" into the existing touch (see below). If the touch is within 64 px, or 3.5 cm, of a non-spurious touch which was lifted within the last 40 ms, the new touch is considered to be a continuation of the recently-lifted touch, and no touch-up or touch-down events are sent to the rest of the program.
3. *All lifted touches coming from the driver do not send a touch-up event to the program for a short period of time.* The 40 ms pause mentioned above allows the touch to be continued, ignoring spurious gaps in touch data caused by the user's finger slightly exiting the frame's sensors. If the 40 ms delay passes without the lifted touch being continued, then a touch-up event is sent to the rest of the program as usual.
4. *When a touch from the driver is absorbed into an existing touch, the movement of that finger is ignored for the purposes of moving the touch.* Only the first finger that triggered the touch actually controls how the touch moves. However, if the absorbed finger moves far enough away from the initial finger's current position, the absorbed finger will split off into a new touch, which is treated identically to a touch for which the driver just sent a touch-down event, as in `enum:touchdown`. If the initial finger for the touch is lifted, then the next absorbed finger takes over as the finger that controls the touch. When all absorbed fingers are lifted, we wait to send a touch-up event as in `enum:touchup`
5. *A touch tap event is triggered if a non-spurious touch is touched down and touched up within a short period of time, across a small distance.* A "tap" consists of a non-spurious touch that exists for at most 400 ms. Because of the small delays introduced by the filtering process, this means that short, brisk swipes might be seen as "taps", so to help discriminate, any touch that moves more than 64 px, or 3.5 cm, total during its lifetime will not trigger a tap.
6. *The movement of a non-spurious touch is lowpass-filtered.* Having a finger that is controlling cursor motion or screen motion jump around can be very frustrating when trying to select small or distant targets. We apply the one-euro filter [8] to the coordinates of the touch movements

that are presented to the rest of the program. We use the parameters 1.0 for *minimum cutoff*, 0.007 for *beta*, and 1.0 for *derivative cutoff*.

This filter layer was active during every interaction mode in both experiments. The parameters were identical for each interaction mode, except for one: as the CursorTap technique is particularly sensitive to spurious touch-taps on the finger that moves the cursor, and there is generally only one finger moving the cursor, we expand the radius in which a touch can be considered a continuation of a recently lifted touch. The radius is set to the maximum of the default (64 px, or 3.5 cm) and 1.1 times the distance moved between the finger's current position and the finger's last position, such that if the finger disappears from the frame then reappears while moving at a constant velocity, it will continue the previous touch instead of triggering a touch-down and a touch-up. This expansion happens only for the finger that is moving the cursor.

Received July 2021; revised September 2021; accepted September 2021