The Fat Thumb: Using the Thumb's Contact Size for Single-Handed Mobile Interaction

Sebastian Boring, David Ledo, Xiang 'Anthony' Chen,
Nicolai Marquardt, Anthony Tang, Saul Greenberg
Dept. of Computer Science, University of Calgary, 2500 University Dr. NW, Calgary, AB, T2N 1N4, Canada
[sebastian.boring, dledomai, anthony.xiangchen, nicolai.marquardt, tonyt, saul.greenberg]@ucalgary.ca

ABSTRACT
Modern mobile devices allow a rich set of multi-finger interactions that combine modes into a single fluid act, for example, one finger for panning blending into a two-finger pinch gesture for zooming. Such gestures require the use of both hands: one holding the device while the other is interacting. While on the go, however, only one hand may be available to both hold the device and interact with it. This mostly limits interaction to a single-touch (i.e., the thumb), forcing users to switch between input modes explicitly. In this paper, we contribute the *Fat Thumb interaction technique*, which uses the thumb's *contact size* as a form of simulated pressure. This adds a degree of freedom, which can be used, for example, to integrate panning and zooming into a single interaction. Contact size determines the mode (i.e., panning with a small size, zooming with a large one), while thumb movement performs the selected mode. We discuss nuances of the *Fat Thumb* based on the thumb's limited operational range and motor skills when that hand holds the device. We compared *Fat Thumb* to three alternative techniques, where people had to precisely pan and zoom to a predefined region on a map and found that the *Fat Thumb* technique compared well to existing techniques.

Author Keywords
Mobile device; touch-screen; single-handed interaction

ACM Classification Keywords
H.5.2 [Information interfaces and presentation]: User Interfaces: Input Devices and Strategies, Interaction Styles.

INTRODUCTION
Today, many mobile devices are equipped with multi-touch screens allowing for different content manipulations by using multiple fingers. For example, users can interact with a map by varying the number of touch points (i.e., panning the map with one finger, zooming with the two finger pinch gesture), rather than by selecting dedicated controls (e.g., menus, buttons) to switch between modes. This is especially important for mobile devices, as it reduces the number of controls cluttering the small display.

MobileHCI'12, September 21–24, 2012, San Francisco, CA, USA.

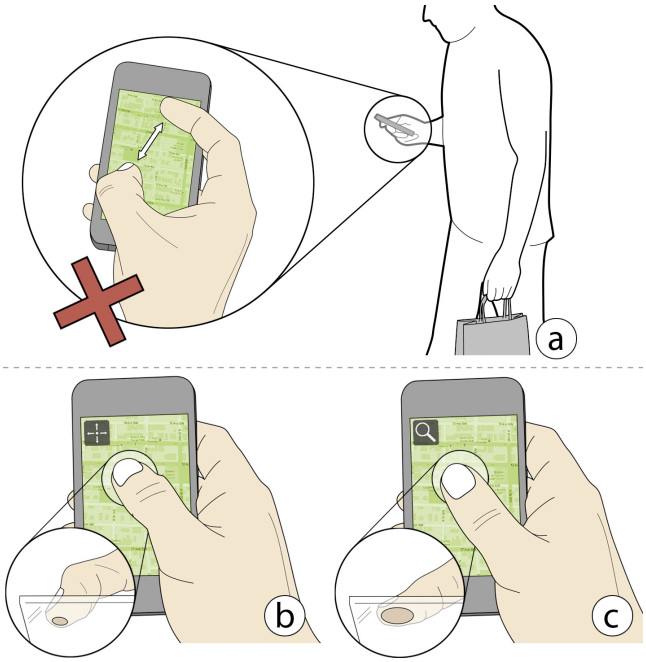

Figure 1. Performing multi-touch actions on a device with just one hand requires awkward hand postures (a); the *Fat Thumb* technique allows fluent single-handed mobile interaction through contact size; e.g., panning (b) and zooming (c) a map.

Yet multi-touch gestures on mobile devices require both hands: one to hold the device, and the other to perform the gestures. This can be problematic, as there are many situations – such as when a person is carrying a bag – when only one hand is available to both hold and interact with the device [14]. Multi-touch actions during such one-handed use become awkward. Figure 1a illustrates such a scenario, where we see how performing the thumb/index finger pinch gesture while holding the device in the same hand requires awkward hand postures. This leads to less precision, more fatigue, and insecure grip of the device (see Figure 1a, left). Alternately, users may perform a sequence of single touch operations to achieve the same result. However, this usually involves an input mode switch between actions: selecting modes via buttons and menus, single-touch gestures recognized as mode-switches [19,22], or employing different hand parts (e.g., touching and tilting the device) [12].

To mitigate this problem, we contribute *Fat Thumb*: a method for one-handed input on a mobile device that caters to the thumb's limited operational range. It uses the thumb's *contact size*, where changing the thumb's *pitch* fluidly allows for

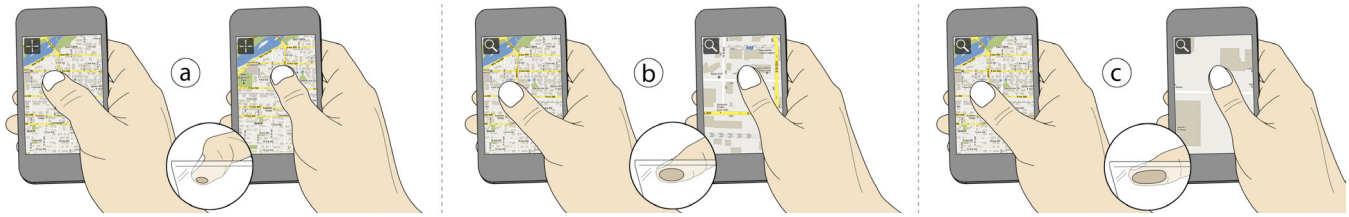

Figure 2. Walkthrough of pan and zoom example: movement with small contact size pans the map (a), increasing contact size switches to zoom (b), and further increasing the contact causes faster zoom operation (c).

transitioning between different modes of interaction (Figure 1b,c). The thumb's *movement* then performs the selected mode. We illustrate and evaluate this technique in a pan-and-zoom task in comparison to exiting single-handed techniques (i.e., a slider, *Tilt-to-Zoom* [12], and *CycloStar* [21]). Our findings demonstrate that the *Fat Thumb* technique compared well to existing one-handed pan-and-zoom techniques, with participants overwhelmingly preferring it.

THE FAT THUMB INTERACTION TECHNIQUE

To illustrate the application of the *Fat Thumb* interaction technique, we use pan and zoom in a map as the driving example throughout this paper [23]. Figure 2 shows a walkthrough of *Fat Thumb* in this scenario (also see the video figure): (a) shows how moving the thumb with a small contact size pans the content. In (b), increasing the contact size switches to the zoom, and moving the thumb in zoom mode around its joint with the palm allows zooming in (right direction) and out (left direction). Finally, in (c) further increasing the contact decreases the control-display (CD) ratio resulting in faster zoom operations.

Fat Thumb has three major differences compared to existing techniques for single-handed mobile device input:

1. It does not require the user to break out of a gesture to switch modes (as with a *Slider*). Instead, it allows for mode switching by adjusting the contact size while moving the thumb. It reduces clutching operations like *Tilt-to-Zoom* [12], thumb fatigue, and avoids moving other hand parts (e.g., the wrist in *Tilt-to-Zoom*).
2. It optimizes for the thumb's limited operational range and motor skills. Compared to other methods, it avoids precise circular strokes as required in *CycloStar* [21], or repositioning the thumb to control a slider.
3. *Fat Thumb* uses the contact size instead of an actual measure of applied pressure. Therefore, the approach causes less friction than pressure-based input systems, even when moving the thumb with a large contact size, which in turn decreases fatigue.

RELATED WORK

Fat Thumb builds on prior reports of the thumb's limitations on mobile devices, as well as prior offerings of single-handed interaction methods designed to mitigate these limitations. We also leverage related work on pressure-based systems, and on using contact shape as an input parameter for touch screen interaction.

The Thumb's Limitation on Mobile Devices

Single-handed interaction with a mobile device presents unique challenges not found in larger interactive touch screens and surfaces [7]. Often, the thumb of the hand holding the device is the only finger that is available for input [13]. Because the hand's key muscles are employed to hold the device [16], the degrees of freedom for input are limited to the thumb's joints (see Figure 3a). Both the *metacarpophalangeal* (MCP) and the *interphalangeal* (IP) joint control the movement of the thumb's tip to and away from the palm. The *carpometacarpal* (CMC) joint enables thumb rotation around the wrist [19], however, that joint is also 'connected' to the device as it secures the grip [27].

Since the thumb's CMC joint cannot be moved without changing the grip, the thumb's length limits its reachability (Figure 3b,c). Considering these kinematic properties (the thumb's movement and its reachability) has strong implications for the ergonomics of an interaction. In a 3 × 3 region matrix, Parhi et al. showed that the center target is the most preferred [26]. Karlson et al. and Henze et al. confirmed this for larger numbers of targets [9,16]. Although lower right regions (when used with the right hand) appear to be within the thumb's reach, fully bending the thumb around its MCP and IP joints requires users to use a different grip of the device [13]. As well, the lower left region is less preferred as it is at the thumb's angular limits [28].

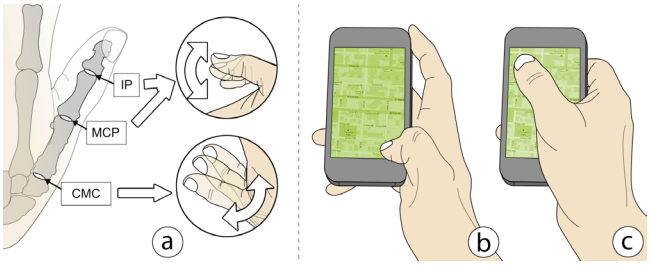

Figure 3. The joints of the thumb (a) limit its movement and reachability on the mobile touch screen. Especially lower right (b) and upper left (c) are hard to reach for right-handed users.

The thumb can perform movements in two directions: rotating around the CMC joint and rotating around both IP and MCP joints. Karlson et al. found that certain thumb motions were ergonomically difficult: moving the thumb diagonally (to and away from the palm) and horizontally requires more effort resulting in longer task times [16]. This suggests that employing the thumb's IP and MCP joints for motion is less suitable than the CMC joint.

Fat Thumb considers these limitations by: (1) not requiring placing the thumb near the device's edges; and (2) restricting its relative movement (in modes that do not require absolute and direct input) around the CMC joint.

Single-Handed Mobile Interaction using Touch

Gestures can be used to overcome limited input capabilities (e.g., only one touch point is sensed) or to increase the expressiveness of an interaction [39]. For example, Olwal et al. presented *Rubbing*, a technique for zooming in and out by performing fast strokes using a single finger [25]. Similar to the *Virtual Scroll Ring* [24], Malacria et al. show that using a circular motion around the point of interest for zooming performs better than *Rubbing* as it requires less high-speed movements [21]. Unlike *Fat Thumb*, these methods were designed for stationary screens (thereby allowing the user to freely position their hand and arm); thus it is unclear how well these work in a mobile context.

Researchers explored designs that account for the thumb's limitations. *ThumbSpace* improves accuracy for small targets that are out of the thumb's reach [18]. *AppLens* and *LaunchTile* use scalable user interfaces to access multiple applications on the same screen [17], similar to *ZoneZoom* on mobile phones featuring a keypad [30]. With *TapTap* users perform a tap to temporarily zoom into a sub-region (with a fixed zoom factor), followed by selecting the target with another tap [32]. With *MicroRolls*, Roudaut et al. demonstrate how users can perform 16 gestures by rolling the finger rather than sliding it, e.g., rolling left has a different meaning than rolling right [33]. *TapSense* allows for using different hand parts (e.g., a knuckle) to select a mode [8]. Hinckley et al.'s *Sensor Synaesthesia* uses the device's accelerometers to zoom in or out while the thumb is not moving [12]. These methods involve an action on a single location to switch the input mode. In contrast, *Fat Thumb* allows for continuous movement of the thumb on the display. Like *MicroRolls*, *Fat Thumb* allows for thumb-rolling, but unlike *MicroRolls*, it can be combined with motion.

Pressure on Mobile Devices

Pressure as input parameter is closely related to the contact size (i.e., more pressure suggests a larger contact size due to flattening of the thumb). Using a stylus for input, Ramos et al. found that a level of six different pressure values is optimal and distinguishable by users [29]. In *GraspZoom*, users can press the back of the device to temporarily switch from panning to a zooming mode [23]. Stewart et al. used pressure from the front, back or sides of a mobile device and found that applying pressure from the device's sides works best [35]. Shi et al. tested different mapping functions, such as fisheyes [34]. However, none of these systems explore the use of pressure while simultaneously moving the finger.

Pressure-sensitive input has been predominantly used for text entry [22], Clarkson et al. added pressure sensors under a regular phone's keypad, e.g., to replace multi-tapping with different pressure levels [5]. Brewster et al. extend this to keyboards on touch screens to distinguish between lower- and uppercase [3]. Wilson et al. evaluated pressure-based techniques in mobile settings (i.e., while people were walking) and found that there is hardly any difference to the performance while being seated [37]. *Fat Thumb* allows for similar input styles, except that it measures contact size instead of pressure (unlike *Glimpse* [6]), while still exploiting the user's mental model of simulated pressure.

Contact Shapes as Additional Information for Input

Capacitive and vision-based touch screens allow sensing of the contact's shape and size respectively [7,20]. Instead of only giving the contact point (calculated from the shape's center of mass), contact shapes allow for disambiguation of different hand parts touching the surface. In *Sphere*, menus can only be triggered with a finger, while placing the palm on a menu item does not affect it [2]. Moscovich uses contact size to allow for a subsequent selection of all targets that were covered by a finger [24]. *SimPress* uses small contact sizes to simulate a *hover* state and larger ones for selecting a target [2]. *ShapeTouch* discriminates coarse contact shapes of the finger *vs.* hand to change modes [4]; *Fat Thumb* also uses contact shape to change modes, but differs as it only relies on fine-grained variations in thumb's contact shape. Rogers et al.'s *AnglePose* further uses a conventional capacitive touch screen to track the finger's angle in 3D [31]. This work further focuses on using the thumb's pitch (much like *Fat Thumb*) for a variety of applications.

The contact shape and size can be used for input correction or to increase selection accuracy. In the aforementioned *MicroRolls*, the contact size gives information about the finger's angle [33]. Holz et al. present a refined model that considers the shape's change over time to precisely estimate the finger's movement [15]. In addition, Wang et al. use the contact's shape to estimate the finger's rotation [36]. With the limitations of a capacitive touch screen on a consumer mobile device, *Fat Thumb* approximates these techniques.

DESIGNING FAT THUMB: CONTACT SIZE AS INPUT

Our overall goal is to allow for more expressiveness while only using the thumb as the input finger. As current UIs primarily track x,y positions of touch points, applications make use of buttons to switch modes which are usually located alongside the device's edges. However, this is costly: they use scarce screen space and they require more targeting and taps. Furthermore, Karlson et al. found that the device's center presents a "sweet spot", where the borders of the display are rather hard to reach and are not preferred by users [16]. Yet, permanently placing controls for mode switching at the screen's center is not an option, as they clutter the screen. For these reasons, we set out to add another input dimension – the thumb's contact size – to complement the x,y position *while* the thumb is in motion. That is, we wanted to exploit this third dimension for mode switching as one moves the thumb two-dimensionally on the display, giving different meanings to motion.

Pressure vs. thumb-tip angle. People can use two methods to alter contact size: pressure, and thumb-tip angle. First, people can adjust a finger's contact size by applying more or less pressure to the surface. The problem is that contact size only changes slightly on rigid surfaces (unlike on softer surfaces, such as used in *Liquid Displacement Sensing* [10]) leading to rather coarse and almost binary input values (see Figure 4). Vision-based systems can do better: because they operate with infrared light, they can use the contact point's brightness as an estimate of pressure: the brighter the blob, the more pressure applied. Likewise, a matrix of infrared emitters and detectors embedded directly behind the display provides similar information about the contact shape and size [14]. Yet most of today's mobile devices use a capacitive sensor matrix: this only allows sensing of the contact size (i.e., whether a pixel on the display is covered or not). This can be used to determine the outline of each contact point (e.g., [20]), but not the actual pressure of that touch.

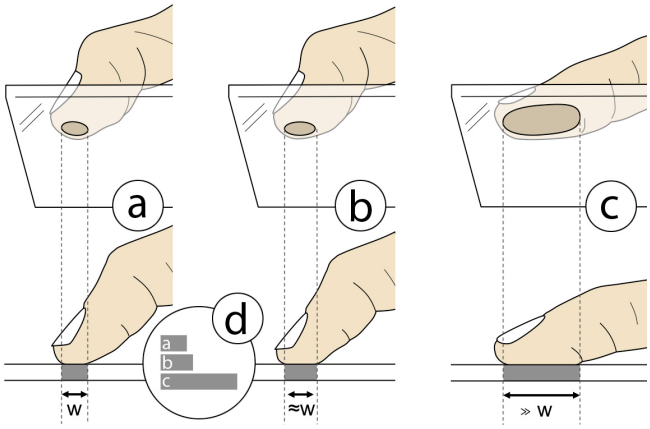

Figure 4. Compared to the original state (a), the contact size only slightly changes when using pressure (b), but changes significantly when pitching the thumb (c) on rigid surfaces. The callout shows the differences between all states (d).

Even if high accuracy pressure sensing becomes available on mobile devices, it has another fundamental drawback: applying more pressure increases the thumb tip's friction on the surface. As the thumb has to overcome resistance introduced by this friction, the force applied to move it might cause unwanted, coarse thumb movements, which affects movement accuracy. Thus, with this approach a user *either* performs regular two-dimensional input (with little or no pressure) *or* applies pressure. Although this would allow for switching modes in place, we decided against it, as it rules out mode switching while the thumb is in motion.

Instead of changing the contact size through pressure, people can also adjust it using the thumb-tip angle. Holz et al. identified the importance of all three angles (*roll*, *pitch*, and *yaw*) of the fingertip for precise touch input [15]. For our case, the thumb's orientation (*yaw*) likely does not affect the contact size itself. However, both the thumb's *pitch* and *roll* will change the contact size. Although implemented on a pixel-resistive touch screen, Roudaut et al.'s *MicroRolls* can make use of this effect. For instance, rolling the thumb to one side decreases its contact size. Likewise, the higher the finger's pitch, the lower its contact size gets (see Figure 4a, and c). For each of these strategies, friction only slightly increases and still allows for finger movement. Thus, this approach allows a combination of the two parameters (i.e., the thumb's x,y-coordinates plus the contact size).

APPLYING FAT THUMB TO PAN & ZOOM OPERATIONS

With contact size as additional dimension, we can integrate two (or more) modes into the same style of operation. As mentioned earlier, we chose zooming and panning a map – a common task on mobile devices – as our driving example. The usual method of *one finger move* (pan) and *two-finger pinch* (zoom) is awkward when the same hand also holds the device (see Figure 1). We mitigate this with the *Fat Thumb* approach, where we will show how both panning and zooming can be controlled through the thumb's motion with different contact sizes for each of the modes. The basic idea is that a person pans during a normal touch (small contact area) but rotates the thumb somewhat to zoom (large contact area).

Details and nuances of *Fat Thumb* pan and zoom are explained below. Because the range of contact sizes slightly varies among users (due to different thumb sizes), we use the following notation: *0* represents the smallest possible contact size, *1* the largest possible one, and fractions indicate sizes between these extremes. Both the smallest and largest possible contact size is obtained through a one-time calibration on a per-user basis.

Mapping Contact Sizes to Different Modes

The thumb's motion is naturally mapped to pan operations in an absolute and direct fashion. That is, whenever the thumb moves, the location it 'holds' underneath it moves with it accordingly. In contrast, when the mode is switched to *zoom*, the thumb's relative movement adjusts the level of zoom. That is, in this mode, moving the thumb in a certain direction (e.g., vertically up or down) zooms the map in or out with a zoom factor based on moved distance: the map does not move but only increases or decreases in size.

In both cases, the thumb's *contact size* determines the mode while the thumb's *movement* simultaneously performs the action associated to that mode. We assumed that it is slightly easier to move the thumb with a small contact size (i.e., only the tip touches the surface) as it (1) is the natural way we use mobile touch screens and (2) introduces slightly less thumb-tip friction. Also, we assumed that people use *panning* more frequently than *zooming*. For these reasons, we attribute smaller contact sizes (i.e., $size \leq 0.5$) to pan operations and larger ones (i.e., $size > 0.5$) to zoom interactions. Further investigation is needed to determine whether 0.5 is the best possible threshold for switching modes.

Panning. The thumb's movement while *panning* remains unchanged, which is compatible to how panning is performed in today's applications (see Figure 5a). However, we assume that because users are solely employing their

thumb instead of their index finger (in two-handed scenarios), they will use multiple short strokes close to the center rather than longer ones from corner to corner. This is due to the thumb's limited operational range for movements.

Zooming. As zooming with a single touch point is not done in an absolute fashion, there needs to be a meaningful mapping between the relative thumb movements and resulting zoom factors. As discussed before, one of the best fits for thumb motion is rotating it around its CMC joint (see Figure 5b). The relative nature of *zooming* (with some constraints to the origin of interaction as discussed in the next section) allows for such movements. For the interaction, it means that the angular change around the CMC joint as rotation center between two contact locations determines the zoom factor (see Figure 5b).

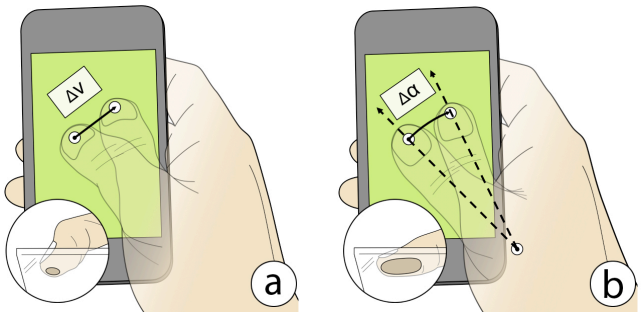

Figure 5. Mapping of thumb movements to panning (a) and zooming (b) a map.

Users can switch between these modes without lifting the finger off the display by altering the thumb tip's angle. That is, once the contact size changes from lower than 0.5 to greater then 0.5, the mode instantly changes. Thus, as described later, *Fat Thumb* allows seamlessly switching between the two modes to a certain extent.

Improvements: Zoom Center and Speed

Selecting the zoom center. The widely used, absolute two-finger *pinch* defines the zoom center (i.e., the location that remains fixed during a zoom operation) as the center point between the two touch points. With *Fat Thumb's* relative single touch point it remains unclear where the zoom origin should be located. One option is to use the center of the display as fixed point. Yet, this may be sub-optimal, as it may require subsequent panning to the actual desired region of interest. We can improve on this by taking the point where the mode changed to zoom (i.e., the first point with a contact size larger than 0.5) as the zoom origin for subsequent relative motions to zoom in or out. Although this only allows for coarsely selecting area of interest (i.e., it is still not pixel-exact), it likely reduces the amount of subsequent corrective pan (and potentially zoom) actions.

Zoom Speed. If the targeted region is much smaller (and larger respectively) than what is currently shown on the display, users may want to alter the magnification or demagnification factor of the zoom operation. With *linear* mapping, the thumb has to be moved by a large distance (potentially with several clutching operations) to accomplish this. Naturally, increasing the zoom factor (i.e., more change per angular change) allows for large changes with minor movements, but may lead to unwanted *overshooting effects*. As alternative, we decided to make use of the contact size again: until now, zoom occurs if the contact size is greater than 0.5 without considering the actual value. Instead, the actual value between 0.5 and 1 can be used to amplify the zoom speed. That is, more precise but slower zooming occurs when the contact size is close to 0.5. Faster and coarse zooming happens for contact sizes close to 1. Between these values, the amplification factor changes linearly.

IMPLEMENTATION

Fat Thumb (see Figure 6) runs on a standard Apple iPhone 4S (iOS version 5.1.1). The device's display has a resolution of 640 × 960 pixels and a diagonal of 3.5" (i.e., 329 dpi). Its 800 MHz dual-core A5 processor allows for smooth operations (60 touch points per second) even with larger, tiled map imagery. The iPhone's API only provides the major radius (and *not* the shape and orientation) in pixels of a contact point's ellipse, which is obtained as follows:

```
NSValue *val = [touch valueForKey:@"_pathMajorRadius"];
float size = [val floatValue]; // size in pixels
```

Conceptually, our application can be ported to other platforms as long as the touch screen of corresponding devices gives at least the contact size or (even better) its shape.

Adjusting the Sensed Touch Location

The iPhone's API reports touch events with their location corresponding to the contact area's center of gravity. For *Fat Thumb*, this has two major drawbacks: (1) especially when in zoom mode (i.e., covering a larger area with the thumb), the user's perceived touch location differs from the sensed location. And (2), altering the contact size by tilting the thumb leads to changes of the contact point's center. Previous vision-based systems allow for detecting the finger's contact size and orientation, allowing for a 'corrected' contact coordinate. Although the iPhone used for our prototype (unlike other devices) does not report the finger's orientation, the restricted position of the thumb allows for a reasonable approximation of the touch location as perceived by a user. Our prototype assumes that the contact point's ellipse is collinear with the thumb's orientation. Since the thumb roughly originates from the same position (unless the device is gripped differently), we can use the sensed location to estimate the thumb's orientation. To do so, we further need the (mostly fixed) location of the hand's CMC joint. With these two parameters, we can calculate the orientation vector and its angle.

With the orientation vector and sensed location, our prototype then calculates the point close to the finger's tip. In informal tests with several users, we found this rough approximation works reasonably well when the thumb is not

heavily flexed (i.e., the tip is not close to the palm). Nevertheless, this correction fails (if the orientation is not natively reported by the sensing hardware) once users adjust the grip in a way that the CMC joint is repositioned significantly.

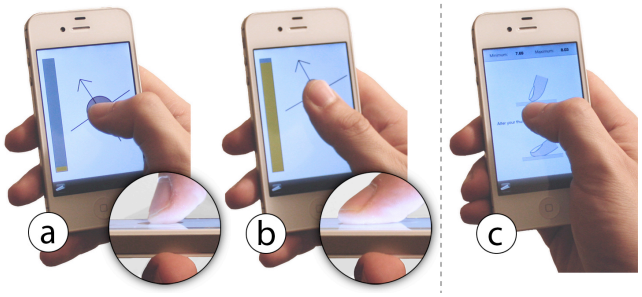

Figure 6. The *Fat Thumb* visualizer (a,b) and calibration screen (c). Callouts show the thumb's pitch angle.

One-time Setup and Calibration

At first launch, users perform a one-time calibration to measure the smallest and largest contact size (see Figure 6c). To do so, they place their thumb on the display and alter the thumb tip's angle while touching the display. This procedure requires only a few contact points, and takes less than a minute. To reduce errors, the implementation uses an interval that increases/decreases the lower/upper bound of the measured values by 2.5% (i.e., we used 95% of the interval to cut off noise). Users can verify their calibration with the Visualizer (see Figure 6a,b), and recalibrate if needed. Once satisfied, the values are stored in the device and they can start using *Fat Thumb*. The current implementation further requires users to specify in which hand they hold the device. While this should suffice in most cases, we acknowledge that detecting the hand holding the device (e.g., *HandSense* [38]) is a more elegant approach.

Limitations

Our implementation currently has two limitations: (1) the thumb's contact point (even with the aforementioned adjustment) will move a short distance during lift-off. That is, the map may (depending on the last sensed contact size) zoom or pan a bit. We currently circumvent this by ignoring the last three touch events (i.e., about 50 ms of the interaction for a touch rate of 60 Hz). And, (2) the thumb always has slightly varying contact sizes while in motion on the display (the differences increase when moved towards the screen's corners). However, *Fat Thumb* was specifically designed to work in the display's center; thus, avoiding the aforementioned case. If used in other scenarios, the *Fat Thumb* technique has to be adjusted.

EVALUATING SINGLE-HANDED PAN AND ZOOM

We conducted a user study to validate *Fat Thumb*. We were especially interested in how *Fat Thumb* compares to existing single-handed, single-touch approaches for controlling pan and zoom. The task was simple and familiar to users: given a predefined area of a map, participants had to use pan and zoom to fit and center it on the entire display.

Interface Conditions

Participants performed this task using four different interface conditions, each of which allows for panning and zooming content. In all conditions, panning was implemented as it is normally performed on touch screen devices.

- **Slider**: we added a slider on the right side of the screen (for left-handed use, the slider was displayed on the left side) that enabled for zooming in or out. The *Slider* did not offer absolute selection of zoom factors, as we were interested in thumb movements on the device. Although not commonplace on mobile multi-touch devices, it is the most known controls for zooming.
- **CycloStar**: this is a gesture-based technique [21]. Users had to perform a circular gesture to zoom in (clockwise) or out (counter-clockwise) around the center point of the drawn circle. Although not intended for thumb-use, we were interested in the limits of gesture-based interactions in single-handed scenarios. We chose *CycloStar* over Olwal et al.'s *Rubbing* [25] as pilot tests indicated that *Rubbing* less accurate and more fatiguing.
- **Tilt-to-Zoom**: this technique makes use of the built-in accelerometers [12]. To map tilting operations to changes in zoom factor, we used the same transfer function and parameters as described by Hinckley et al. [11].
- **Fat Thumb**: we implemented this condition as described before (including its improvements).

We did not include the iPhone's built-in pan/zoom methods for the following two reasons: first, its elaborate pinch gesture usually requires two hands; while possible with one hand by using the thumb and index finger, it requires an awkward hand posture. And second, the iPhone's single-handed zoom feature (i.e., double-tap to increase the zoom factor by a fixed amount) implements zooming in a discrete fashion instead of a continuous one. It further requires a tap with two fingers to zoom out, introducing problems similar to those found in the pinch gesture.

Task

Participants were presented with a series of individual area alignment tasks. We used three different area sizes (each of them mirrored the aspect ratio of the display to allow for perfect alignment) located diagonally from the center with constant distance. We avoided search times by making targets at least partially visible on screen, i.e., where they at least intersect with the display's boundaries. We also did not use different distances as panning is performed equally in all techniques, thus adding constant time to each task. We asked participants to bring the area into the display's center as quickly and accurately as possible.

Before starting a trial, participants saw the semi-transparent red target area (to minimize the time required for visual search during the task), as well as a *start button*. When participants pressed the *start button*, the button disappeared and timing began. They had to center and fit the red target area into the display's boundaries, which required a series

of pan and zoom operations with the given technique. The error was given in offset, which describes the difference in both the display's total area and the target area (i.e., an offset of 0% indicates that they fit perfectly). The offset was displayed at the bottom of the screen and updated frequently so that participants were aware of the current offset at all times. When a participant was satisfied with the alignment (and the offset value shown respectively), they performed a double-tap to end the trial. Timing was stopped after a successful alignment. Subsequently, the map was reset and the start button displayed and the process would repeat for the next trial. Overall, we recorded task time, target offset, and the number of strokes used during a trial. We also recorded all finger movements (i.e., every touch point with its contact size) as well as the target offset over time.

Study Design

We used a repeated measures within-subjects factorial design. Independent variables were *Technique* (*Slider*, *CycloStar*, *Tilt-to-Zoom*, and *Fat Thumb*), target area *Direction* (*NE*, *NW*, *SW*, and *SE*), and target area *Size* (*Small*, *Medium*, and *Large*). Targets had a fixed distance from the mobile display's center in each direction.

Technique was counter-balanced across our participants. We created pairs of each the four *Directions* and the three *Sizes*. These were presented in random order within each block. For each *Technique*, we had one practice and four timed blocks in the experiment. After completing one *Technique*, participants were asked to fill out a standard device assessment questionnaire. After the study, we asked them to rank the four *Techniques*. Each participant completed the study in 120 minutes or less, a time that included training of all techniques at the beginning of the study.

Apparatus

We conducted the experiment on an Apple iPhone 4S, and its standard 1.94" × 2.91" (640 × 960 pixel) retina display with 329 dpi. All techniques functioned as previously described. The map had 24 image tiles (4 rows, 6 columns); each tile was 2048 × 2048 pixels to increase iPhone performance (larger images significantly slow down this device). Since the map did not have the same aspect ratio (to force pan operations), it was scaled down so that it had the same height as the display, but was still wider. The resulting scale factor was about 0.12. The target sizes were (fully zoomed out): *Small* (75 × 112.5 pixels; 0.23" × 0.34"), *Medium* (150 × 225 pixels; 0.46" × 0.68"), and *Large* (225 × 337.5 pixels; 0.68" × 1.03"). Their center was horizontally and vertically located 293 (0.89") and 257.8 (0.78") pixels away from the display's center, regardless of their size.

To mimic a realistic scenario of panning and zooming a map while 'on-the-go', participants were not allowed to use their second hand at any time during the experiment, nor were they allowed to rest their forearm on any surface during a trial. However, they were seated and could take breaks in between trials if fatigued.

Participants

We recruited 16 participants (10 female) ranging in age from 19 to 32 years (avg.: 23.8 years). 8 of them had corrected vision, and three were left-handed. 14 own a mobile phone, and 15 were familiar with using touch-based mobile devices (median = 4, 5-point Likert scale where 5 equals highly experienced) and had used a mapping application before. They received $15 as compensation for their time.

Hypotheses

Based on our understanding of single-handed thumb use on mobile devices we hypothesized the following: (H1) for *Small* targets, *Fat Thumb* outperforms all other techniques in terms of offset. (H2) *Fat Thumb* outperforms the other techniques in terms of task time with comparable offsets for *Small* areas. (H3) *Fat Thumb* and *Tilt-to-Zoom* require the least amount of strokes for *Small* and *Medium* targets due to their clutch-free nature. (H4) *Fat Thumb* is the most preferred technique (particularly with respect to fatigue) as it is designed for the thumb's limited operational range.

RESULTS

We compared task completion time, target area offset, and the number of strokes used per trial with a separate repeated measures within-subjects analyses of variance (ANOVA). For pair-wise post hoc tests, we used Bonferroni-corrected confidence intervals to compare against $\alpha = 0.05$. In cases where the assumption of sphericity was violated, we corrected the degrees of freedom using Greenhouse-Geisser. All unstated p-values are $p > 0.05$.

We first performed a 4 × 4 (*Technique* × *Block*) within-subjects ANOVA and found significant main effects for *Block* and *Technique* regarding task time (all $p < 0.003$), but not offset and number of strokes. Post hoc multiple means comparison tests showed that *Block 1* differed significantly from *Blocks 2–4* (all $p < 0.017$). Thus, we excluded *Block 1* from subsequent analyses and aggregated the remaining *Blocks*. To verify that we can also aggregate across *Location*, we performed a 4 × 4 (*Technique* × *Location*) within subject ANOVA and found no main effects or interactions.

Offsets

With the aggregated offsets we performed a 4 × 3 (*Technique* × *Size*) within subjects analysis of variance. We found significant main effects for *Technique* ($F_{1.913,28.691} = 31.723$, $p < 0.001$). There was, however, no significant main effect for *Size*, or an interaction between *Technique* and *Size*. With post hoc multiple means comparisons we found that *Fat Thumb* was more accurate than *Slider* for *Small* and *Medium Sizes* (all $p < 0.022$). Also, *Fat Thumb* was more accurate – for all *Sizes* – than *Tilt-to-Zoom* (all $p < 0.042$), and *CycloStar* (all $p < 0.001$). This supports H1.

Figure 7 shows the offsets per *Technique* and *Size*. Most interestingly, the offset for each *Technique* is higher when *Size* increases. We attribute this to users rapidly overshooting the target as most techniques are designed for rapid

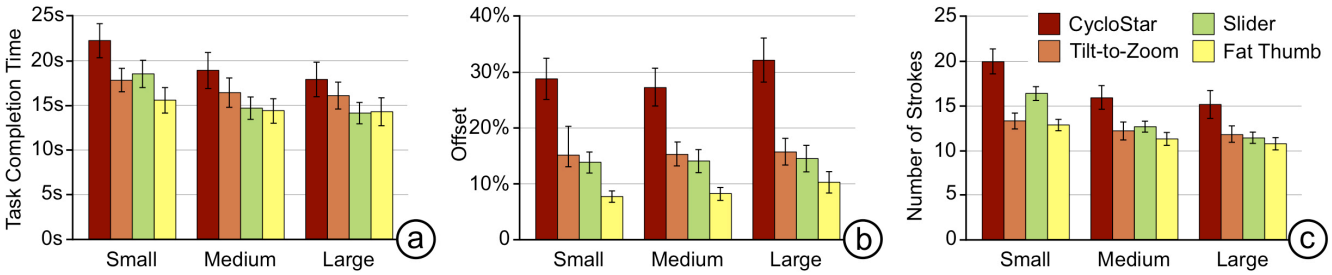

Figure 7. Task completion time (a), offset (b), and number of strokes (c) clustered by *Size* (error bars: 95% confidence interval).

zooming. The figure also reveals that *CycloStar* had the highest offset, which we believe has two reasons: (1) as expected, the technique is hard to perform using the thumb. And, (2) the technique caused user frustration, ending trials once an acceptable alignment was achieved. Overall, *Fat Thumb* had the least offset (M = 8.8%, SD = 1.5%), followed by *Slider* (M = 14.2%, SD = 2.5%), *Tilt-to-Zoom* (M = 15.4%, SD = 2.6%), and *CycloStar* (M = 29.4%, SD = 4.2%). However, one may argue that offsets of up to 15% may be acceptable depending on the task, which then only renders *CycloStar* as insufficient technique.

Task Completion Time

We measured task time from the moment the start button was pressed until participants double-tapped the target. We performed a 4 × 3 (*Technique* × *Size*) within subjects ANOVA and found significant main effects for *Technique* ($F_{3,45}$ = 4.133, $p < 0.011$), *Size* ($F_{2,30}$ = 37.744, $p < 0.001$), and a *Technique* × *Size* interaction ($F_{6,90}$ = 2.755, $p < 0.017$).

Figure 7 illustrates how task completion times increased with decreasing target area size for *CycloStar*. For the other techniques, the task time barely changes for *Large* and *Medium* sized targets, but changes for *Small* ones. This behavior may be explained with overshooting effects for targets that need high zoom factor adjustment. To discover the nature of the *Technique* × *Size* interaction, we ran separate ANOVA tests for each *Size* level. For *Large* areas, we found that there is no significant difference across techniques. For *Medium* areas, we found that *Slider* and *Fat Thumb* differ significantly from *CycloStar* (all $p < 0.04$). For *Small* areas, *Fat Thumb* differs significantly from *CycloStar* ($p < 0.02$). However, *Fat Thumb* does not show any significant difference regarding task completion time compared to *Slider* or *Tilt-to-Zoom* (thus, we reject H2). Nevertheless, there is a trend that it does not increase task time if targets get smaller while maintaining low offsets.

Overall, *Fat Thumb* was the fastest (M = 14.78s, SD = 1.94s), followed by *Slider* (M = 15.79s, SD = 1.74s), and *Tilt-to-Zoom* (M = 16.77s, SD = 1.81s). *CycloStar* was the slowest among all techniques (M = 19.68s, SD = 2.17s).

Number of Strokes

For each technique, we analyzed the number of strokes (i.e., one stroke begins with the finger being placed down and ends when it is released) that participants needed to complete a trial. We performed a 4 × 3 (*Technique* × *Size*) within subjects ANOVA and found significant main effects for *Technique* ($F_{2.015,30.228}$ = 14.002, $p < 0.001$) and *Size* ($F_{2,30}$ = 114.848, $p < 0.001$). We further found a *Technique* × *Size* interaction ($F_{6,90}$ = 14.571, $p < 0.001$).

Figure 7 suggests that the number of strokes is related to task completion time, thus leading to similar results. We separately analyzed each *Size* level and found significant main effects for each level ($p < 0.001$). For *Small* target areas, *Fat Thumb* and *Tilt-to-Zoom* significantly differ from the other techniques (all $p < 0.028$), with *Fat Thumb* presumably having fewer strokes. This supports H3. For *Medium* sized targets, *Fat Thumb* also differs significantly from *CycloStar* and *Slider* (all $p < 0.005$). *Tilt-to-Zoom*, however, only differs significantly from *CycloStar* ($p < 0.002$). Unsurprisingly, for *Large* targets, Figure 7 indicates that all but *CycloStar* have comparable number of strokes. However, only *Fat Thumb* differs significantly ($p < 0.023$).

Overall, *Fat Thumb* required the least strokes (M = 11.67, SD = 0.81), followed by *Tilt-to-Zoom* (M = 12.46, SD = 1.23), and *Slider* (M = 13.51, SD = 0.75). The most strokes were needed for *CycloStar* (M = 17.01, SD = 1.54).

Subjective Preferences

After each *Technique*, participants filled a questionnaire to assess the particular input technique. *Fat Thumb* scored consistently well across all categories on a five-point Likert scale. Most importantly, both *Fat Thumb* and *Tilt-to-Zoom* only caused little fatigue for fingers (median = 2), while *Slider* (median = 4), and *CycloStar* caused high fatigue for fingers (median = 4.5). Also, *Fat Thumb* ranked better for wrist fatigue (median = 2) than *Slider* and *Tilt-to-Zoom* (both median = 3) and *CycloStar* (median = 4). Furthermore, *Fat Thumb* was ranked the easiest to use (median = 4), followed by *Slider* (median = 3.5), *Tilt-to-Zoom* (median = 3), and *CycloStar* (median = 1). Consistently with these results, *Fat Thumb* had the highest general comfort (median = 4), followed by *Slider* and *Tilt-to-Zoom* (median = 3), and both *CycloStar* (median = 1.5).

After they completed the study, participants ranked the four *Techniques* by preference: We found a significant difference in ranks ($\chi^2(3) = 29.325$, $p < 0.001$). Post hoc analysis with Wilcoxon Signed-Rank Tests further revealed significant differences for all pairs (all $p < 0.038$) except *Slider*

and *Tilt-to-Zoom*. Overall, they ranked *Fat Thumb* highest (11), followed by *Slider* (4), *Tilt-to-Zoom* (1), and *CycloStar* (0), which proves our hypothesis H4.

DISCUSSION

Our study results support all but one of our hypotheses. *Fat Thumb* proved to be more accurate for *Small* targets than the other techniques (H1). In Figure 7, one can observe that *Fat Thumb's* task time and offset also does not increase much when target areas get smaller. Our results also showed that both *Fat Thumb* and *Tilt-to-Zoom* had the least number of strokes as expected (H3). As with offset and task time, for smaller target sizes, the number of strokes only slightly increases. However, further testing with a larger range of target sizes is needed to confirm this trend.

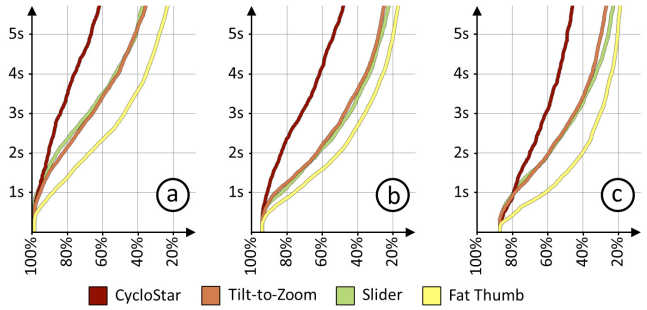

Figure 8. Progress of offset over time (first 5 seconds) for *Small* (a), *Medium* (b), and *Large* areas (c).

Surprisingly, however, the task time of *Fat thumb* did not show a significant difference to all other techniques for *Small* targets as hypothesized (H2): We assumed that one of the reasons was that participants tried to be highly precise – leading to high task times for all techniques. We plotted the progression of aggregated offsets over time to identify how much time of the overall task participants spent on fine-grained adjustments. As shown in Figure 8, *Fat Thumb* was the technique that approached the final offset faster than the other techniques. For example, *Fat Thumb* reaches 20% offset of *Medium* targets about two seconds faster than *Tilt-to-Zoom* and *Slider* We believe that this offset already is sufficient in our scenario, but acknowledge that other use-cases could potentially be harmed by this offset rate.

The gesture-based *CycloStar* approach proved poor for one-handed use. We believe that this is due to the thumb's limited operational range and not due to the original idea. *CycloStar* was designed for use with the index finger while the other hand holds the device, and in those cases it may fare well. The subjective ratings of our participants support this as *CycloStar* had the highest fatigue for wrist and finger. To understand these ratings, we analyzed the strokes performed on the device. In particular, we identified the areas participants used during the study and compared them against findings regarding 'good' and 'bad' areas on a PDA-sized touch device as identified in the literature [16]. As shown in Figure 9, *Fat Thumb* and *Tilt-to-Zoom* unsurprisingly used the center of the device. However, *CycloStar* almost used the entire touch screen. We assume that these reasons (i.e., large area of use, and fast movements) were the main sources for (1) low comfort and low ease-of-use, as well as (2) high fatigue. Thus, we suggest that *CycloStar* is not well suited for single-handed mobile touch-based interaction.

For task time and number of strokes, both *Fat Thumb* and *Tilt-to-Zoom* performed equally. However, *Tilt-to-Zoom* had lower subjective ratings regarding wrist and finger fatigue as well as the general comfort (H4). In addition, participants stated that tilting the display decreased the visibility during the task, which explains the higher offset of *Tilt-to-Zoom*. *Slider* had comparable results for *Large* targets, but permanently requires costly screen space. Also, its subjective ratings were consistently lower than *Fat Thumb*'s ratings. We assume that the necessity of reaching the side of the device introduced by the *Slider* lead to this effect.

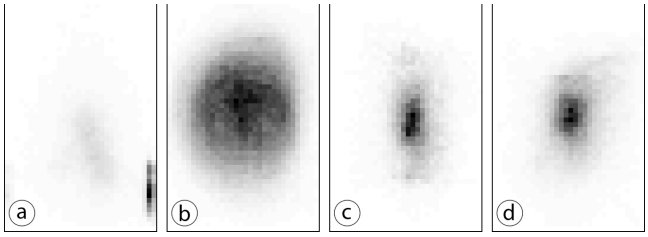

Figure 9. Heatmaps of touch points: *Slider* (a), *CycloStar* (b), *Tilt-to-Zoom* (c), and *Fat Thumb* (d).

One potential drawback of *Fat Thumb* is slightly higher occlusion introduced by larger contact sizes. While this may turn into a problem in selecting small targets, it does not interfere with our pan/zoom scenario. As well, *Fat Thumb* and its mapping could be implemented on a back-of-the-device touch pad (see [1]), eliminating occlusion.

CONCLUSIONS AND FUTURE WORK

We presented *Fat Thumb*, an alternative technique that adds a dimension to touch input on a mobile device, thus giving different meanings to touch motion. We do this by making use of the thumb's contact size to allow for seamless mode switching. We demonstrated the technique by integrating panning and zooming into a single operation where the contact size determines the mode while moving the thumb performs it. Our user study of panning/zooming revealed that *Fat Thumb* is at least as fast as other existing techniques, non-fatiguing, and the preferred technique, all while having the lowest offset rates for smaller target areas of all techniques. Together with *Tilt-to-Zoom*, *Fat Thumb* required the least number of strokes, in part because participants could switch between modes without lifting their finger.

While our study applied *Fat Thumb* to panning and zooming operations, we believe that *Fat Thumb* can be used for a variety of tasks. To explore this, we will investigate how many different contact size levels can actually be perceived and used by people, as studied with pressure levels [29]. Based on this, we will identify other single-handed mobile interactions made possible through *Fat Thumb*.

ACKNOWLEDGMENTS

This research is partially funded by the iCORE/NSERC/ SMART Chair in Interactive Technologies, Alberta Innovates Technology Futures, NSERS, and SMART Technologies Inc. The authors thank Matthew Dunlap, Marian Dörk, and Uta Hinrichs for their valuable feedback.

REFERENCES

1. Baudisch, P. and Chu, G. Back-of-device interaction allows creating very small touch devices. *Proc. CHI*, ACM (2009).
2. Benko, H., Wilson, A.D., and Baudisch, P. Precise selection techniques for multi-touch screens. *Proc. CHI*, ACM (2006).
3. Brewster, S.A. and Hughes, M. Pressure-based text entry for mobile devices. *Proc. MobileHCI*, ACM (2009).
4. Cao, X., Wilson, A., Balakrishnan, R., Hinckley, K., and Hudson, S. ShapeTouch: Leveraging Contact Shape on Interactive Surfaces. *Proc. TABLETOP*, IEEE (2008).
5. Clarkson, E.C., Patel, S., Pierce, J., and Abowd, G.D. Exploring Continuous Pressure Input for Mobile Phones. *GVU Techreport; GIT-GVU-06-20*, Georgia Inst. of Tech. (2006).
6. Forlines, C., Shen, C., and Buxton, B. Glimpse: a novel input model for multi-level devices. *Proc. CHI EA*, ACM (2005).
7. Han, J.Y. Low-cost multi-touch sensing through frustrated total internal reflection. *Proc. UIST*, ACM (2005), 115–118.
8. Harrison, C., Schwarz, J., and Hudson, S.E. TapSense: enhancing finger interaction on touch surfaces. *Proc. UIST*, ACM (2011), 627–636.
9. Henze, N., Rukzio, E., and Boll, S. 100,000,000 Taps: Analysis and Improvement of Touch Performance in the Large. *Proc. MobileHCI*, ACM (2011).
10. Hilliges, O., Kim, D., and Izadi, S. Creating malleable interactive surfaces using liquid displacement sensing. *Proc. TABLETOP*, IEEE (2008), 157–160.
11. Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E. Sensing techniques for mobile interaction. *Proc. UIST*, ACM (2000).
12. Hinckley, K. and Song, H. Sensor synaesthesia: touch in motion, and motion in touch. *Proc. CHI*, ACM (2011).
13. Hirotaka, N. Reassessing current cell phone designs. *Proc. CHI*, ACM (2003), 938–939.
14. Hodges, S., Izadi, S., Butler, A., Rrustemi, A., and Buxton, B. ThinSight: versatile multi-touch sensing for thin form-factor displays. *Proc. UIST*, ACM (2007), 259–268.
15. Holz, C. and Baudisch, P. Understanding touch. *Proc. CHI*, ACM (2011), 2501–2510.
16. Karlson, A.K., Bederson, B.B., and Contreras-Vidal, J.L. Studies in One-Handed Mobile Design: Habit, Desire and Agility. *Tech report HCIL-2006-02*, Comp. Sci. Dept., University of Maryland, MD (2006).
17. Karlson, A.K., Bederson, B.B., and SanGiovanni, J. AppLens and launchTile: two designs for one-handed thumb use on small devices. *Proc. CHI*, ACM (2005), 201–210.
18. Karlson, A.K. and Bederson, B.B. ThumbSpace: Generalized One-Handed Input for Touchscreen-Based Mobile Devices. *Proc. INTERACT*, Springer, 324–338.
19. Kuo, L.-C., Cooney, W.P., Chen, Q.-S., Kaufman, K.R., Su, F.-C., and An, K.-N. A kinematic method to calculate the workspace of the trapeziometacarpal joint. *Mechanical Engineers Journal of Engineering in Medicine 218*, 2 (2004).
20. Lee, S., Buxton, W., and Smith, K.C. A multi-touch three dimensional touch-sensitive tablet. *Proc. CHI*, ACM (1985).
21. Malacria, S., Lecolinet, E., and Guiard, Y. Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces. *Proc. CHI*, ACM (2010), 2615–2624.
22. McCallum, D.C., Mak, E., Irani, P., and Subramanian, S. PressureText: pressure input for mobile phone text entry. *Proc. CHI*, ACM (2009), 4519–4524.
23. Miyaki, T. and Rekimoto, J. GraspZoom: zooming and scrolling control model for single-handed mobile interaction. *Proc. MobileHCI*, ACM (2009).
24. Moscovich, T. and Hughes, J.F. Navigating documents with the virtual scroll ring. *Proc. UIST*, ACM (2004), 57–60.
25. Olwal, A., Feiner, S., and Heyman, S. Rubbing and tapping for precise and rapid selection on touch-screen displays. *Proc. CHI*, ACM (2008), 295–304.
26. Parhi, P., Karlson, A.K., and Bederson, B.B. Target size study for one-handed thumb use on small touchscreen devices. *Proc. MobileHCI*, ACM (2006), 203–210.
27. Pascoe, J., Ryan, N., and Morse, D. Using while moving: HCI issues in fieldwork environments. *ACM ToCHI 7*, (2000), 417–437.
28. Perry, K.B. and Hourcade, J.P. Evaluating one handed thumb tapping on mobile touchscreen devices. *Proc. GI*, Canadian Information Processing Society (2008), 57–64.
29. Ramos, G., Boulos, M., and Balakrishnan, R. Pressure widgets. *Proc. CHI*, ACM (2004), 487–494.
30. Robbins, D.C., Cutrell, E., Sarin, R., and Horvitz, E. ZoneZoom: map navigation for smartphones with recursive view segmentation. *Proc. AVI*, ACM (2004), 231 – 234.
31. Rogers, S., Williamson, J., Stewart, C., and Murray-Smith, R. AnglePose: robust, precise capacitive touch tracking via 3d orientation estimation. *Proc. CHI*, ACM (2011), 2575–2584.
32. Roudaut, A., Huot, S., and Lecolinet, E. TapTap and MagStick: improving one-handed target acquisition on small touch-screens. *Proc. AVI*, ACM (2008), 146–153.
33. Roudaut, A., Lecolinet, E., and Guiard, Y. MicroRolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. *Proc. CHI*, ACM (2009),.
34. Shi, K., Irani, P., Gustafson, S., and Subramanian, S. PressureFish: a method to improve control of discrete pressure-based input. *Proc. CHI*, ACM (2008), 1295–1298.
35. Stewart, C., Rohs, M., Kratz, S., and Essl, G. Characteristics of pressure-based input for mobile devices. *Proc. CHI*, ACM (2010), 801–810.
36. Wang, F., Cao, X., Ren, X., and Irani, P. Detecting and leveraging finger orientation for interaction with direct-touch surfaces. *Proc. UIST*, ACM (2009), 23–32.
37. Wilson, G., Stewart, C., and Brewster, S.A. Pressure-based menu selection for mobile devices. *Proc. MobileHCI*, ACM (2010), 181–190.
38. Wimmer, R. and Boring, S. HandSense: discriminating different ways of grasping and holding a tangible user interface. *Proc. TEI*, ACM (2009), 359–362.
39. Wu, M. and Balakrishnan, R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *Proc. UIST*, ACM (2003), 193–202.