# 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**
Department of Computer Science, University of Calgary
2500 University Drive 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 pan and zoom to a predefined region on a map. Participants performed fastest with the least strokes using Fat Thumb.

**Author Keywords**
Mobile devices, touch-screens, single-handed use, interaction techniques, pressure-based input.

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

**General Terms**
Design, Experimentation, Human Factors.

## INTRODUCTION
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* with one finger, *zooming* with two), 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.

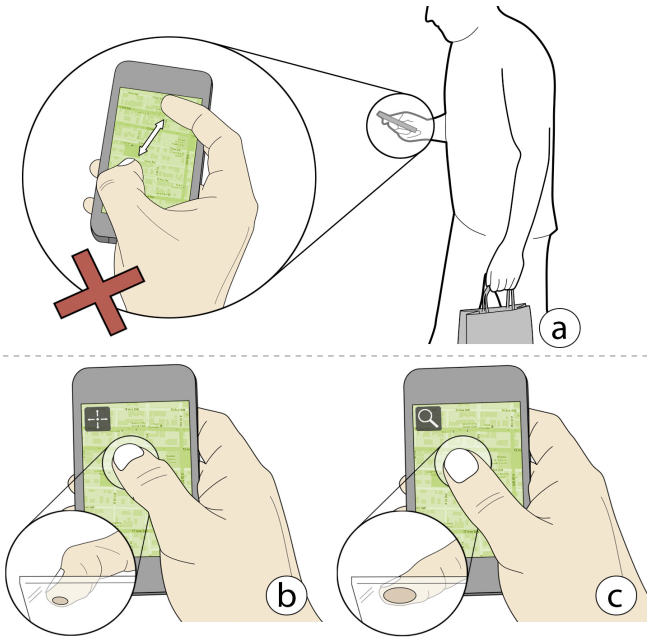


**Figure 1. Performing multi-touch actions on a device with just one hand requires awkward hand postures (a); the Fat Thumb technique allows fluent single-handed mobile interaction; for example, panning (b) and zooming (c) a map.**

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 strategies to achieve the same result. However, this usually involves an input mode switch between actions: selecting modes via buttons and menus, or single-touch gestures recognized as mode-switches (e.g., [18,21]).

Cite as:
Boring, S., Ledo, D., Chen, X., Marquardt, N., Tang, A., Greenberg, S. (2011), The Fat Thumb: Using the Thumb's Contact Size for Single-Handed Mobile Interaction.
*Research Report 2011-1015-27*, Department of Computer Science, University of Calgary, Calgary, AB, Canada T2N 1N4, September.

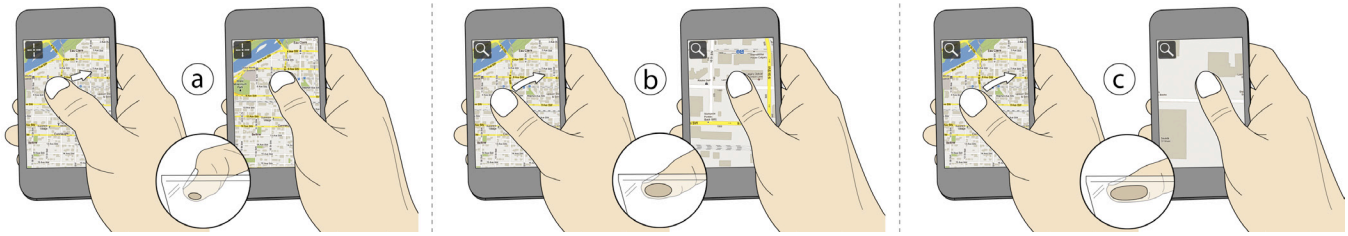


**Figure 2. Walkthrough of pan and zoom example: movement with small contact size pans the map (a), increasing contact size switches to zoom (b), and further increasing the contact causes faster zoom operation (c).**

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 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, *Rubbing* [21], and *CycloStar* [18]). Our findings demonstrate that users were fastest using the Fat Thumb technique, and overwhelmingly preferred it to existing techniques for one-handed pan-and-zoom.

## 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 main example throughout this paper. Figure 2 shows a walkthrough of the Fat Thumb technique in this scenario (also see the video figure, as it best reveals the dynamics of the Fat Thumb interaction): (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 wrist allows zooming in (right direction) and out (left direction). Finally, in (c) further increasing the contact decreases the control-display (CD) ratio resulting in a faster zoom operation.

*Fat Thumb* has three strengths over existing techniques for single-handed mobile device input:

1. It does not require the user to break out of a gesture to switch modes. Instead, it allows (in our example) for integrated panning and zooming by adjusting the contact size while moving the thumb. Thus, it reduces the number of clutching operations as well as thumb fatigue.
2. It optimizes for the thumb's limited operational range and motor skills. Compared to other methods, it avoids rapid thumb movement as required in *Rubbing* [21], or precise circular strokes as required in *CycloStar* [18].
3. Fat Thumb uses the contact size instead of an actual measure of applied pressure (which usually cannot be sensed on today's mobile touch screens). 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 Limitations 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 [10]. Because the hand's key muscles are employed to hold the device [14], 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 [16], however, that joint is also 'connected' to the device as it secures the grip [23].

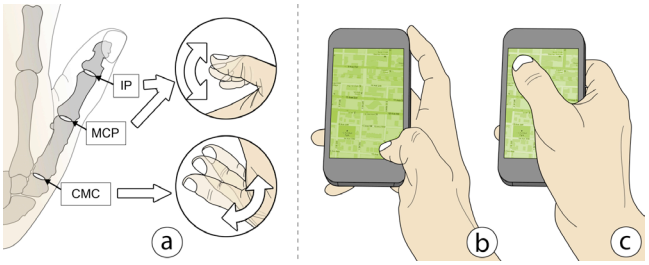


**Figure 3. Joints of the thumb (a) limit its movement and reachability on the mobile display (b,c).**

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 [22]. Karlson et al. and Henze et al. confirmed this for larger numbers of targets [8,14]. 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 [10]. As well, the lower left region is less preferred as it is at the thumb's angular limits [24].

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 [14]. 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 [33]. For example, Olwal et al. presented *Rubbing*, a technique for zooming in and out by performing fast strokes using a single finger [21]. Similar to the *Virtual Scroll Ring* [20], 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 [18]. 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.

A few researchers did explore designs that account for the thumb's specific input capabilities. *ThumbSpace* improves accuracy for small targets that are out of the thumb's reach [13]. *AppLens* and *LaunchTile* use scalable user interfaces to access multiple applications on the same screen [15], similar to *ZoneZoom* on mobile phones featuring a keypad [26]. With *MicroRolls*, Roudaut et al. demonstrate how users can perform 16 elementary gestures by rolling the finger rather than sliding it, e.g., rolling left has different meaning than rolling right [27]. All these methods involve an action on a single location. In contrast, Fat Thumb allows for continuous movement of the thumb on the display. For example, like MicroRolls, Fat Thumb allows for thumb-rolling, but unlike MicroRolls, it can be combined with motion. Allowing input also on the back of the device allows for dual-touch input (i.e., the thumb on the front, and one finger on the back) [1]. However, this may result in insecure grips or require the use of both hands [23].

#### 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). The general characteristics of pressure have previously been studied. Ramos et al. found that a level of six different pressure values is optimal and distinguishable by users [25]. Stewart et al. used pressure from the front, back or both sides of a mobile device and found that applying pressure from both sides of the device works best [29]. Shi et al. tested different mapping functions, such as fisheyes [28]. 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 – especially for mobile phones with multi-tap keypads. Similar to *PressureText* [19], Clarkson et al. added pressure sensors under a regular phone's keypad, for example, to replace multi-tapping with different pressure levels for text entry [6]. Brewster et al. extend this to keyboards on touch screens to distinguish between lower- and uppercase [4]. 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 [31]. Fat Thumb allows for similar input styles, except that it measures contact size instead of pressure, while still exploiting the user's mental model of simulated pressure.

#### Contact Shapes as Additional Information for Input

Both capacitive and vision-based multi-touch screens allow sensing of the contact's shape and size respectively [7,17]. Instead of only giving one contact point (usually 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 [20]. *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 [5]; Fat Thumb also uses contact shape to change modes, but differs as it only relies on fine-grained variations in thumb's contact shape.

The contact shape (and size respectively) can further be used for input correction or to increase selection accuracy. In the aforementioned *MicroRolls*, the contact size gives information about the finger's angle [27]. Holz et al. present a refined model that considers the shape's change over time to precisely estimate the finger's movement [12]. In addition, Wang et al. use the contact's shape to estimate the finger's rotation [30]. With the limitations of a capacitive touch screen on a consumer mobile device, Fat Thumb approximates these techniques.

### THE CONTACT SIZE AS INPUT DIMENSION

Our overall goal is to allow for more expressiveness while only using the thumb as the input finger. Current UIs of mobile devices have buttons placed on their edges. Yet, 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 [14]. However, 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 – thumb contact size – to complement the x,y position. That is, we wanted to exploit this third dimension for mode switching as one simultaneously moves the thumb two-dimensionally on the display.

### Adding a New Dimension: Pressure vs. Angle

People can use two methods to alter contact size: pressure, and thumb-tip angle. First, people can adjust (i.e., increase or decrease) 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 (as opposed to softer surfaces, such as use in *Liquid Displacement Sensing* [9]) 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. Alternatively, a matrix of infrared emitters and detectors embedded directly behind the display can provide similar information about the contact shape and size (*ThinSight* [11]). Yet most of today's mobile devices (e.g., Apple's iPhone) 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 touch contact point (e.g., [17]), but not the actual pressure of that touch.

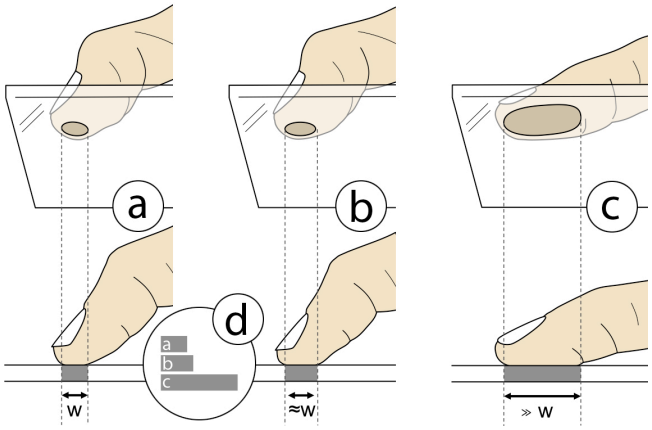


**Figure 4. Increasing the original contact size (a) by pressure (b) results in small changes; varying the thumb's pitch (c) leads to larger changes. (d) shows a comparison.**

Even if pressure becomes available, 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 friction, the force applied to move it may cause unwanted, coarse thumb movements, which also affects movement accuracy. Thus, with this approach a user *either* performs regular two-dimensional input (with little or no pressure) *or* applies pressure. This would allow for switching modes in place, but we decided against it, as it rules out combining the two.

A second approach uses the thumb-tip angle to adjust contact size. Holz et al. identified the importance of all three angles (*roll*, *pitch*, and *yaw*) of the fingertip for precise touch input [12]. For our case, the thumb's orientation (*yaw*) likely does not affect the contact size itself. However, both the thumbs's *pitch* and *roll* will change the contact size. 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. 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) to increase input expressivity.

### Allowing for 'Taps' with Different Contact Sizes

One common interaction on touch screens, however, is tapping at a certain location to perform a single selection (i.e., a *click*). In applications, this is usually sensed as sequences of *down* and *up* events. Contact size can also be used here to detect the 'intensity' of a finger tap, and use that as an additional input variable. While placing the finger down/lifting it up, the contact size rapidly increases/decreases rapidly. Furthermore, at first contact (i.e., the *down* event), the contact's size is very small. The *up* event produces a similar result (see Figure 5).

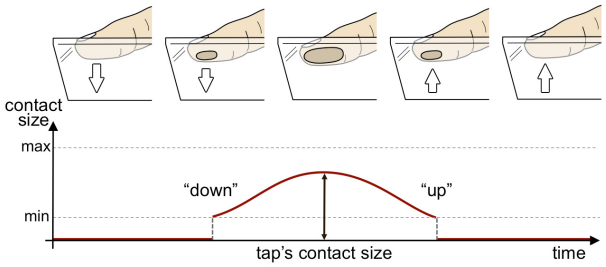


**Figure 5. Changing contact sizes when tapping over time. Normally, only the "down" and "up" events are intercepted; causing small contact sizes for each tap.**

Nevertheless, during these two events, the contact size reaches a peak, which represents the actual surface area of the tap. Thus, to allow for taps with different contact sizes, an application can consider not only the *down* and *up* events, but also observe the largest contact size present during the tap. Modern capacitive touch screens allow this as they sense slight motions even during a very short tap, returned as a *move* event. Such *move* events occur as the contact size changes, which also influences the sensed $x,y$-coordinates of the contact point (i.e., the contact shape's center). Through this, even short taps allow for different meanings using their contact size.

### Using Contact Size for Input Correction

In current capacitive touch screens, a touch is represented by a single point, which corresponds to the contact area's center of gravity. However, Benko et al. suggest that users perceive the touch's location closer to the tip of their finger [2]. Their vision-based approach allows for detecting a finger's contact size and orientation, making it easy to estimate their 'corrected' contact coordinate. Capacitive touch screens lack this ability. Instead, they only allow for the contact shape (i.e., its outline) without additional information about where the finger's phalanges are on top of the surface. Furthermore, touch-enabled consumer handhelds (e.g., iPhone 4) only return the major radius (as a scalar) of the contact's elliptical shape. Still, the restricted position of the thumb allows for a reasonable approximation of the touch location as perceived by a user.

*Approximating the thumb's orientation.* Although our capacitive touch screen only allows for retrieving the major radius of the contact's ellipse, we can assume it to be 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 detected location $p$ to estimate the thumb's orientation. The second parameter necessary is the location $l$ (in screen coordinates) of the hands CMC joint. With these two parameters, we can calculate the orientation vector $v$ and its angle $\alpha$ with $v = p - l$, and $\alpha = \arctan(v_y / v_x)$. This approximation works best for locations when the thumb is not heavily flexed (i.e., the tip is not close to the palm).

*Deriving the perceived input location.* As mentioned before, the user's perceived input location usually is closer to the thumb's tip [2]. Presumably, it is also rather independent of the contact size (see Figure 6a,b): the distance from the tip remains constant with the value $a$. Together with the thumb's detected input location $p$, its orientation $v$ and the contact's major radius $r$, we can calculate the corrected input point $p'$ as: $p' = p + (r - a) \cdot (v / \text{len}(v))$. As thumb sizes vary among users, so does the fixed factor $a$. For this reason, we set $a$ to the user's smallest possible contact size (i.e., the center of the thumb's tip).

In informal tests with several users, we found this rough approximation works reasonably well. However, this correction fails once users adjust the grip of their mobile device in a way that the CMC joint is repositioned significantly. We believe that future capacitive sensors in consumer devices will alleviate this problem by sensing and reporting the entire outline in a pixel-exact way.

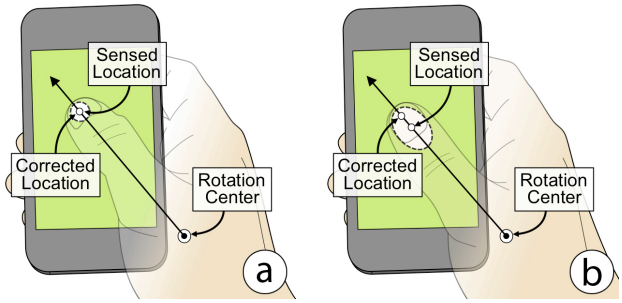


**Figure 6. Using contact size for input correction. Small contact sizes are not corrected (a); larger ones are corrected (b) based on the CMC joint and sensed location.**

## INTEGRATING ZOOM & PAN THROUGH CONTACT SIZE

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* to pan and *two finger pinch* to zoom the map is awkward when the same hand also holds the device. We mitigate this with the Fat Thumb approach, where we will show how both panning and zooming can be performed using only the one contact point of the thumb. The basic ideas is that a person pans during a normal touch (with a large contact area) but rotate their thumb somewhat to zoom (detected as a smaller contact area).

Details and nuances of Fat Finger pan and zoom are explained below. Because the range of contact sizes 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.

### Mapping Contact Sizes to Different Modes

The thumb's motion is naturally mapped to pan operations. That is, whenever the thumb moves, the location it 'holds' underneath it (and thus the entire map) moves with it. This represents an absolute and direct input style. 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 (say: 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.

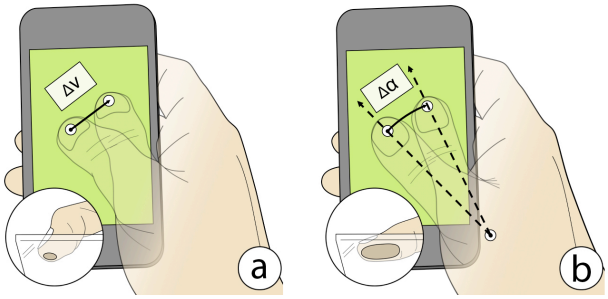


**Figure 7. Mapping of thumb movements to panning (a) and zooming (b) a map.**

In both cases, the thumb's *movement* performs the action associated to the mode. Thus, the contact size indicates the mode the user wants to perform. We assumed that it is still slightly easier to move the thumb with a small contact size (i.e., only the tip touches the surface). At the same time, we assume that people use the *panning* mode more frequently than *zooming*. For this reason, we attribute small contact sizes (i.e., $\leq 0.5$) to pan operations and large contact sizes (i.e., $> 0.5$) to zoom interactions.

The thumb's movement while *panning* remains unchanged, which is compatible to panning as now done in most applications. However, we assume that because users are employing a thumb rather than a finger, they will use multiple short strokes close to the center rather than longer ones. This is due to the thumb's limited operational range for movements (see Figure 7a). In the *zoom* mode, however, there needs to be a meaningful mapping between relative movements and zoom factors. As discussed before, one of the best fits for thumb motion is rotating it around its CMC joint (see Figure 7b). The relative nature of *zooming* (with 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 7b). For two points $p_1$ and $p_2$, the resulting change in zoom factor $\Delta f$ is given as follows: $\Delta f = \alpha_2 - \alpha_1$, $\alpha_i = \arctan(v_{iy} / v_{ix})$ where $v_i$ is the vector between $p_i$ and the rotation center.

This mapping may have two limitations: (1) the thumb's contact point (even with input correction) 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. As the movement is really small, we did not see any problems when informally testing with potential users. And, (2) the thumb always has slightly varying contact sizes while moving on the display (especially when moved towards the display'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.

### Improvements: the Zoom's Center and Speed

One challenge with zooming as described above is that it is not clear where the zoom origin should be located. Zooming the center of the display may be sub-optimal, as it may require subsequent panning to the actual desired location. We can improve on this by taking the user's initial contact point as the zoom origin for subsequent relative motions to zoom in or out. This only allows for coarsely selecting the area of interest (i.e., not pixel-exact), but likely reduces the amount of subsequent corrective pan and zoom actions.

In several cases, users may want to zoom with a high magnification or demagnification factor to reach a very small region or gain an overview of a larger area. In our design above, the thumb has to be moved by a large distance (potentially with several clutching operations) to accomplish this. Increasing the zoom factor (i.e., more change per angular change) is one solution, but may lead to unwanted *overshooting effects*. As alternative, we can 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

The Fat Thumb prototype (see Figure 8a-d) runs on a standard Apple iPhone 4 (iOS version 4.3.5) and is implemented entirely in iOS. The device's display has a resolution of 640 × 960 pixels and a diagonal of 3.5" resulting in 329 dots per inch. Its 1 GHz A4 processor allows for smooth operations (we measured an average of 56.9 touch points per second on our device) even with larger, tiled maps. As mentioned before, the iPhone API provides the major radius (*not* the shape) in pixels of multiple contact points. Conceptually, our application can be ported to other platforms provided that the touch screen of corresponding gives at least this contact size or (even better) its shape.

At first launch, users perform a one-time calibration to measure the smallest and largest contact size (see Figure 8e). To do so, they place their thumb on the display and alter the thumb'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% to reduce errors (i.e., we used 95% of the interval to cut off noise). Users can verify their calibration with a Visualizer (see Figure 8f), and recalibrate if needed. Once satisfied, the values are stored in the device and they can start using Fat Thumb.

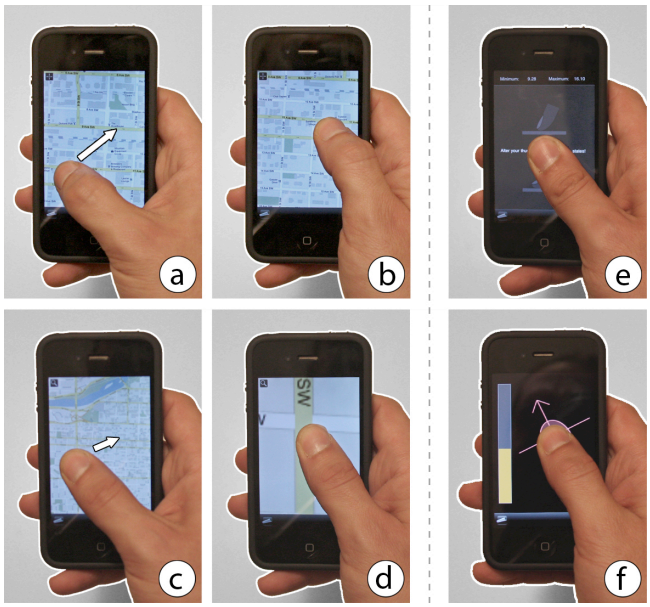


**Figure 8. Fat Thumb prototype: panning (a,b) and zooming (c,d) a map; calibration (e); visualizer (f).**

The current implementation further requires users to specify whether they are left or right handed (i.e., in which hand they predominantly hold their device). While this should suffice in most cases, we acknowledge that detecting the hand holding the device is a more elegant approach. We anticipate that this may be possible with future mobile device, e.g., as suggested by Wimmer et al.'s *HandSense* [32].

## 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 (gesture-based) approaches for controlling pan and zoom. The task was simple: given a predefined area of a map, participants had to use pan and zoom to fit and center it on the entire display. This is equivalent to panning and zooming a map into a particular region of interest.

### 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.
- **Rubbing**: this is based on Olwal et al.'s technique [21] with slight changes due to the thumb's restricted movement capabilities. For both zooming in and out, the rubbing direction is SW ↔ NE. The direction of the first stroke decides whether to zoom in (first stroke: SW → NE) or out (first stroke: NE → SW). For left-handed users, it was NW → SE (in), and SE → NW (out).
- **CycloStar**: this is also a gesture-based technique [18]. In this condition, users had to perform a circular gesture to zoom in (clockwise) or out (counter-clockwise) around the center point of the drawn circle.
- **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 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. Second, the zoom feature in the iPhone *Map* application implements discrete rather than continuous zooming. It also 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 asked participants to bring the area into the display's center as quickly and accurately as possible.

Before each trial began, the participant saw the semi-transparent red target area (to minimize the time required for visual search during the task), and a *start button* (Figure 9a). When participants pressed the *start button*, that button disappeared and timing began. They had to center and fit this red target area into the display's boundaries, which required a series of pan and zoom operations with the given technique (see Figure 9b). Once the target reached an offset of less than 50% (i.e., the area not within the display's boundaries as described by Boring et al. [3]), the area turned blue indicating that the area may be tapped (see Figure 9c). In more detail, the offset describes the difference in both the display's total area and the target area (i.e., an offset of 0% indicates that they fit aligned perfectly). When a participant was satisfied with the alignment (and the offset was less than 50%), they performed a double-tap to end the trial. Timing was stopped after a successful alignment. The map was then 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. In addition, we recorded all finger movements (i.e., every touch point with its contact size) as well as the target offset whenever an input event occurred.

### Study Design

We used a repeated measures within-subjects factorial design. Independent variables were *Technique* (*Slider*, *Rubbing*, *CycloStar*, and *Fat Thumb*), target area *Direction* (*NE*, *NW*, *SW*, and *SE*), and target area *Size* (*Small*, *Medium*, and *Large*). The target areas measured 0.23" × 0.34", 0.46" × 0.68", and 0.68" × 1.03" on the mobile display when fully zoomed out. They were horizontally and vertically located away at 0.89" and 0.78" from the display's center.

*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 one *Technique* was completed, they were asked to fill out a short standard device assessment questionnaire. After the study, we asked them to rank the four *Techniques*. Each participant completed the study in 90 minutes or less, a time that included training of all techniques at the beginning of the study.

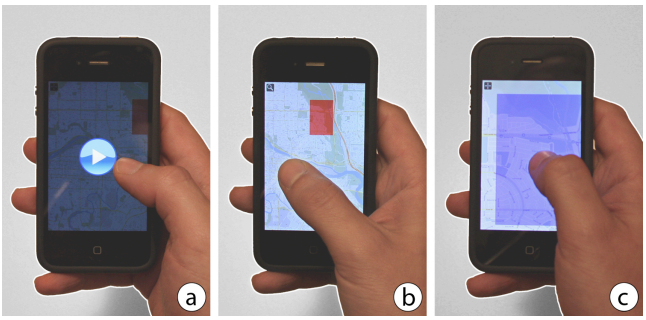


**Figure 9. User study trial sequence: start button (a), navigating to target area (b), and confirming selection (c).**

### Apparatus

We conducted the experiment on an Apple iPhone 4, 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 tiles (4 rows, 6 columns); each tile was 2048 × 2048 pixels to increase iPhone performance (larger images significantly slow down this device). The map did not have the same aspect ratio (to force pan operations) and was scaled down so that it had the same height as the display. The resulting scale factor was 0.12. The target sizes in pixels were: *Small* (75 × 112.5 pixels), *Medium* (150 × 225 pixels), and *Large* (225 × 337.5 pixels). Their center was horizontally and vertically located 293 and 257.8 pixels away from the display's center, regardless of their size.

To mimic a 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 24 participants (12 female) ranging in age from 18 to 31 years (average: 22.6 years). 13 of them had corrected vision (no one reported color-blindness). One was left-handed. 20 own a mobile phone. All participants were familiar with using touch-based mobile devices (median = 4 on a 5-point Likert scale where 5 equals highly experienced) and 23 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 anticipated the following hypotheses: (H1) *Fat Thumb* outperforms all other techniques in terms of task time while having comparable error rates for small areas. (H2) *Fat Thumb* performs faster than *Rubbing* and *CycloStar* without increasing error rates. (H3) *Fat Thumb* requires the least amount of strokes for small targets.

## 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 $p > 0.05$.

Before the main analysis, we performed a 4 × 4 (*Technique* × *Block*) within-subjects ANOVA and found no significant main effect or interactions for *Block* regarding task time, offset, and number of strokes. Thus, we aggregated all dependent variables across *Block* for each participant. To verify that we can further aggregate across *Location* as well, we performed a 4 × 4 (*Technique* × *Location*) within subject ANOVA and also found no main effects or interactions regarding all three dependent variables.

### Offsets

With the aggregated offsets we performed a 4 × 3 (*Technique* × *Size*) within subjects analysis of variance. We found significant main effects for *Technique* ($F_{3,69} = 3.104$, $p < 0.032$) and *Size* ($F_{2,46} = 3.296$, $p < 0.046$). Most relevant, however, is the significant *Technique* × *Size* interaction ($F_{6,138} = 2.306$, $p < 0.037$). With post hoc multiple means comparison tests we found that both *CycloStar* and *Rubbing* are more accurate than *Slider* (all $p < 0.014$) for *Large* target areas. For *Small* and *Medium* areas, no significant differences were found for all techniques. Most importantly, no significant differences were found for *Fat Thumb* with *Large* target areas, showing that it has comparable low offset rates for all tested target sizes.

The offset rates were relatively high for all *Techniques*, but may be explained in part by the double-taps (for confirmation) slightly shifting the area. Overall, *CycloStar* was had the least (M = 34.2%, SD = 0.7%), followed by *Fat Thumb* (M = 34.3%, SD = 0.7%), *Rubbing* (M = 34.5%, SD = 0.6%), and *Slider* (M = 35.8%, SD = 0.6%). Nevertheless, we argue that such offsets in the given scenario of selecting a region on a map are possibly acceptable.

### Task Completion Time

We measured task time from the moment the start button was pressed to the moment participants double-tapped the target area. We performed a 4 × 3 (*Technique* × *Size*) within subjects ANOVA and found significant main effects for *Technique* ($F_{1.66,38.17} = 26.433$, $p < 0.001$) and *Size* ($F_{1.6,37.1} = 23.921$, $p < 0.001$). We further found a *Technique* × *Size* interaction ($F_{2.64,60.67} = 5.121$, $p < 0.005$).

Figure 10 illustrates how task completion times increased with decreasing target area size for *Slider* and *Rubbing*. *CycloStar* shows an interesting behavior in that it has the lowest task time for *Medium* sized targets. This may be explained with overshooting effects for targets that need little or high zoom factor adjustment. Post hoc multiple mean comparison tests revealed that *Fat Thumb* was faster than all other techniques for *Small* target areas ($p < 0.002$). This supports H1. For *Medium* and *Large* target areas, however, *Slider* and *Fat Thumb* did not show significant differences, but both techniques were faster than *Rubbing* (all $p < 0.003$) and *CycloStar* (all $p < 0.001$). This supports H2. We also found, that *Fat Thumb* gets faster when *Size* increases (all $p < 0.02$).

Overall, *Fat Thumb* was the fastest (M = 5.29s, SD = 0.24s), followed by *Slider* (M = 6.20s, SD = 0.34s), and *Rubbing* (M = 11.57s, SD = 1.22s). *CycloStar* was the slowest among all techniques (M = 16.68s, SD = 1.57s).

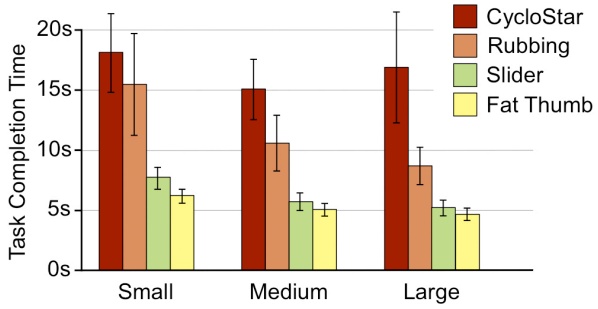


**Figure 10. Mean task completion time clustered by size. Error bars represent 95% confidence intervals.**

### Number of Strokes

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 with each technique. We performed a 4 × 3 (*Technique* × *Size*) within subjects ANOVA and found significant main effects for *Technique* ($F_{1.90,43.75} = 15.388$, $p < 0.001$) and *Size* ($F_{1.33,30.65} = 53.975$, $p < 0.001$). We further found a *Technique* × *Size* interaction ($F_{2.96,67.97} = 5.195$, $p < 0.003$).

Figure 11 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 on each level ($p < 0.001$). Post hoc multiple

mean comparison tests revealed that *Slider* and *Fat Thumb* needed fewer strokes than *CycloStar* ($p < 0.005$) and *Rubbing* ($p < 0.008$) for *Medium* and *Large* target area *Sizes*. For *Small* areas, only *Fat Thumb* showed a significant difference for the number of strokes compared to *CycloStar* and *Rubbing* ($p < 0.001$). Most interestingly is that participants also needed fewer strokes with *Fat Thumb* compared to *Slider* for all *Sizes* ($p < 0.003$). This supports H3.

Overall, *Fat Thumb* required the least strokes (M = 5.66, SD = 0.30), followed by *Slider* (M = 7.78, SD = 0.34), and *Rubbing* (M = 9.84, SD = 0.87). The most strokes were needed for *CycloStar* (M = 12.10, SD = 1.09).

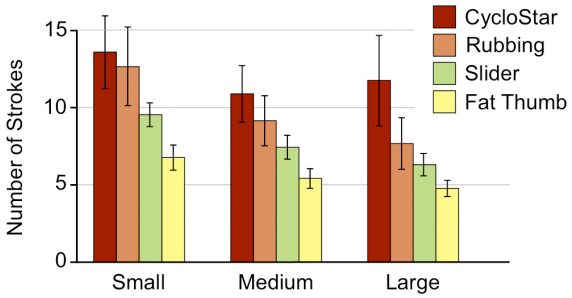


**Figure 11. Average number of strokes per trial clustered by size. Error bars represent 95% confidence intervals.**

### Subjective Preference

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, *Fat Thumb* caused no fatigue for fingers and wrist (both median = 1), followed by *Slider* (both median = 2). *Rubbing* and *CycloStar* caused high fatigue for fingers (median = 4) and slightly less for the wrist (median = 3). Furthermore, *Fat Thumb* and *Slider* were ranked the easiest to use (median = 5), followed by *Rubbing* (median = 2), and *CycloStar* (median = 1). Consistently with these results, *Fat Thumb* had the highest general comfort (median = 5), followed by *Slider* (median = 4), and both *Rubbing* and *CycloStar* (median = 2).

After they completed the study, participants ranked the four *Techniques* by reference. We found a significant difference in ranks ($\chi^2(3) = 50.55$, $p < 0.001$). Post hoc analysis with Wilcoxon Signed-Rank Tests further revealed significant differences for all pairs ($p < 0.004$) except *CycloStar* and *Rubbing*. Overall, ranked *Fat Thumb* highest (17), followed by *Slider* (5), *Rubbing* (2), and *CycloStar* (0).

## DISCUSSION

Our study results support each of the hypotheses, but adds detail. Particularly for small targets, *Fat Thumb* performs better than existing techniques. However, while the *Slider* worked as expected, both *Rubbing* and *CycloStar* proved surprisingly poor for one-handed use. We believe that this is due to the thumb's limited operational range and not due to their original idea. *Rubbing* and *CycloStar* were designed for use with the index finger while the other hand holds the device, and in those cases they may fare well.

To identify the nature of both high task times and low user 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 [14]. As shown in Figure 12, *Fat Thumb* and *Rubbing* used the center of the device. However, *Rubbing* had (1) a higher number of strokes and (2) required more rapid movements of the thumb (which resulted in low comfort ratings). *CycloStar*, on the other hand, 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 *Rubbing* and *Cyclostar* are not well suited for single-handed mobile touch-based interaction.

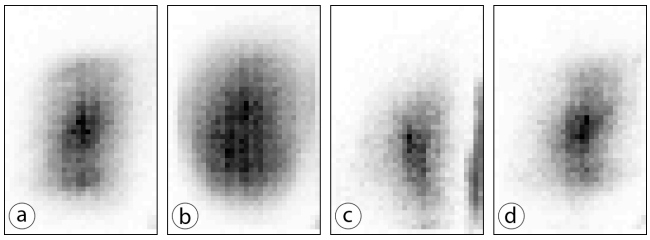


**Figure 12. Heatmap of touch points and strokes: *Rubbing* (a), *CycloStar* (b), *Slider* (c), and *Fat Thumb* (d).**

In all measures, the *Slider* almost performed equally compared to *Fat Thumb*. However, it permanently requires costly screen space. Furthermore, 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.

Another somewhat surprising finding was the rather high error rate for all *Techniques*. In most trials, however, the target was contained inside the display (i.e., it covered about 66% of the display). In a real scenario, this would lead to having more content shown with a slight loss of detail. As stated before, we do not see this as problematic in our scenario, but acknowledge that other use-cases could potentially be harmed by this offset rate.

One potential drawback of Fat Thumb is slightly higher occlusion introduced by larger contact sizes. While this could turn into a problem in selecting small targets, it does not interfere with our pan/zoom scenario, as the entire content responds to the performed interaction still allowing for visual tracking. As well, Fat Thumb could be implemented on a back-of-the-device touch pad (see [1]), thus eliminating the occlusion problem.

## CONCLUSIONS AND FUTURE WORK

We presented Fat Thumb, a technique that integrates panning and zooming into a single operation. It makes use of the thumb's contact size to allow for seamless mode switching. We further demonstrated how we use this additional parameter for input correction, i.e., for determining the perceived touch point. Our user study of panning/zooming

revealed that Fat Thumb is fast (especially when large zoom factors are required), non-fatiguing, and the preferred technique, all while maintaining the offset rates of other techniques. In addition, 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 [25]). 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 also 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. of CHI '09*, ACM (2009), 1923-1932.
2. Benko, H., Wilson, A.D., and Baudisch, P. Precise selection techniques for multi-touch screens. *Proc. of CHI '06*, ACM (2006), 1263-1272.
3. Boring, S., Baur, D., Butz, A., Gustafson, S., and Baudisch, P. Touch projector: mobile interaction through video. *Proc. of CHI '10*, ACM (2010), 2287-2296.
4. Brewster, S.A. and Hughes, M. Pressure-based text entry for mobile devices. *Proc. of MobileHCI '09*, ACM (2009).
5. Cao, X., Wilson, A., Balakrishnan, R., Hinckley, K., and Hudson, S. ShapeTouch: Leveraging Contact Shape on Interactive Surfaces. *Proc. of TABLETOP '08*, IEEE (2008).
6. Clarkson, E.C., Patel, S., Pierce, J., and Abowd, G.D. Exploring Continuous Pressure Input for Mobile Phones. *GVU Technical Report; GIT-GVU-06-20*, Georgia Institute of Technology (2006).
7. Han, J.Y. Low-cost multi-touch sensing through frustrated total internal reflection. *Proc. of UIST '05*, ACM (2005).
8. Henze, N., Rukzio, E., and Boll, S. 100,000,000 Taps: Analysis and Improvement of Touch Performance in the Large. *Proc. of MobileHCI '11*, ACM (2011).
9. Hilliges, O., Kim, D., and Izadi, S. Creating malleable interactive surfaces using liquid displacement sensing. *Proc. of TABLETOP '08*, IEEE (2008), 157-160.
10. Hirotaka, N. Reassessing current cell phone designs. *Proc. of CHI '03*, ACM (2003), 938-939.
11. Hodges, S., Izadi, S., Butler, A., Rrustemi, A., and Buxton, B. ThinSight: versatile multi-touch sensing for thin form-factor displays. *Proc. of UIST '07*, ACM (2007), 259-268.
12. Holz, C. and Baudisch, P. Understanding touch. *Proc. of CHI '11*, ACM (2011), 2501-2510.
13. Karlson, A.K. and Bederson, B.B. ThumbSpace: Generalized One-Handed Input for Touchscreen-Based Mobile Devices. *Proc. of INTERACT '07*, Springer, 324-338.
14. 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).
15. Karlson, A.K., Bederson, B.B., and SanGiovanni, J. AppLens and launchTile: two designs for one-handed thumb use on small devices. *Proc. of CHI '05*, ACM (2005), 201-210.
16. 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. *Proceedings of the Institution of Mechanical Engineers Journal of Engineering in Medicine 218*, 2 (2004), 143 -149.
17. Lee, S., Buxton, W., and Smith, K.C. A multi-touch three dimensional touch-sensitive tablet. *Proc. of CHI '85*, ACM (1985), 21-25.
18. Malacria, S., Lecolinet, E., and Guiard, Y. Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces. *Proc. of CHI '10*, ACM (2010), 2615-2624.
19. McCallum, D.C., Mak, E., Irani, P., and Subramanian, S. PressureText: pressure input for mobile phone text entry. *Proc. of CHI '09*, ACM (2009), 4519-4524.
20. Moscovich, T. and Hughes, J.F. Navigating documents with the virtual scroll ring. *Proc. of UIST '04*, ACM (2004).
21. Olwal, A., Feiner, S., and Heyman, S. Rubbing and tapping for precise and rapid selection on touch-screen displays. *Proc. of CHI '08*, ACM (2008), 295-304.
22. Parhi, P., Karlson, A.K., and Bederson, B.B. Target size study for one-handed thumb use on small touchscreen devices. *Proc. of MobileHCI '06*, ACM (2006), 203-210.
23. Pascoe, J., Ryan, N., and Morse, D. Using while moving: HCI issues in fieldwork environments. *ACM ToCHI 7*, (2000), 417-437.
24. Perry, K.B. and Hourcade, J.P. Evaluating one handed thumb tapping on mobile touchscreen devices. *Proceedings of GI '08*, Canadian Information Processing Society (2008), 57–64.
25. Ramos, G., Boulos, M., and Balakrishnan, R. Pressure widgets. *Proc. of CHI '04*, ACM (2004), 487-494.
26. Robbins, D.C., Cutrell, E., Sarin, R., and Horvitz, E. ZoneZoom: map navigation for smartphones with recursive view segmentation. *Proc. of AVI '04*, ACM (2004), 231 - 234.
27. Roudaut, A., Lecolinet, E., and Guiard, Y. MicroRolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. *Proc. of CHI '09*, ACM (2009), 927-936.
28. Shi, K., Irani, P., Gustafson, S., and Subramanian, S. PressureFish. *Proc. of CHI '08*, ACM (2008), 1295-1298.
29. Stewart, C., Rohs, M., Kratz, S., and Essl, G. Characteristics of pressure-based input for mobile devices. *Proc. of CHI '10*, ACM (2010), 801-810.
30. Wang, F., Cao, X., Ren, X., and Irani, P. Detecting and leveraging finger orientation for interaction with direct-touch surfaces. *Proc. of UIST '09*, ACM (2009), 23-32.
31. Wilson, G., Stewart, C., and Brewster, S.A. Pressure-based menu selection for mobile devices. *Proc. of MobileHCI '10*, ACM (2010), 181-190.
32. Wimmer, R. and Boring, S. HandSense. *Proc. of TEI '09*, ACM (2009), 359-362.
33. Wu, M. and Balakrishnan, R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *Proc. of UIST '03*, ACM (2003), 193-202.