One-Handed Touchscreen Input for Legacy Applications

Amy K. Karlson and Benjamin B. Bederson
Human Computer Interaction Lab, Computer Science Department
University of Maryland, College Park, 20742
{akk, bederson}@cs.umd.edu

ABSTRACT
Supporting one-handed thumb operation of touchscreen-based mobile devices presents a challenging tradeoff between visual expressivity and ease of interaction. ThumbSpace and Shift—two new application-independent, software-based interaction techniques—address this tradeoff in significantly different ways; ThumbSpace addresses distant objects while Shift addresses small object occlusion. We present two extensive, comparative user studies. The first compares ThumbSpace and Shift to peripheral hardware (directional pad and scrollwheel) and direct touchscreen input for selecting objects while standing and walking. The data favored the Shift design overall, but suggested ThumbSpace is promising for distant objects. Our second study examines the benefits and learnability of combining Shift and ThumbSpace on a device with a larger screen (3.5"). We found their combined use offered users better overall speed and accuracy in hitting small targets (3.6 mm) than using either method alone.

Author Keywords
ThumbSpace, Shift, one-handed mobile interaction design.

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
Mobile phones exist today in a wide spectrum of designs. The most widespread styles in circulation feature the classic combination of numeric keypad and non-touchscreen display, but the fervor surrounding new devices such as Apple's iPhone and LG's Prada phone indicate that larger touchscreen devices are gaining ground. Yet as devices evolve, users will remain constrained by the limits of their own visual, physical, and mental resources [14]. In particular, mobile users often have only one hand available to operate a device.

Users can benefit from interfaces that free one hand for the physical and intellectual demands of mobile tasks [16], and surveys [13] confirm that users would generally prefer to use touchscreens with one hand when possible. But today's hardware and software designs typically offer little support.

Indeed, designing interfaces for thumb-based interaction presents a challenging trade-off between visual expressivity and ease of interaction. On one end of the spectrum we have existing (legacy) mobile UI toolkits; these tend to use small, stylus-oriented widgets for touchscreen interaction, resulting in information-rich interfaces composed of targets that are too small [17] or too far to be hit reliably with the thumb [13]. On the other end of the expressivity-interaction spectrum, one could explicitly ensure all targets are thumb sized and within thumb reach [12]. This "lowest common denominator" approach wastes these small devices' precious screen real estate, and slows down information access when two hands *are* available. These factors have led us to consider an alternative design strategy.

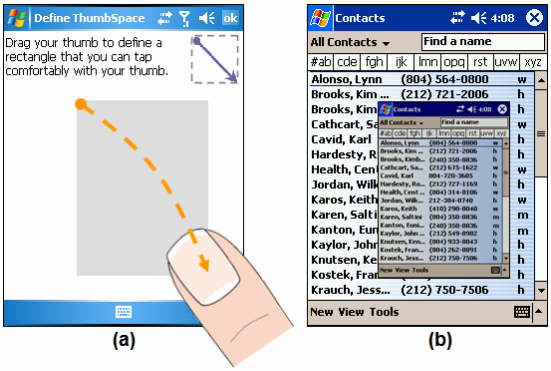


Figure 1. (a) Defining a ThumbSpace. (b) ThumbSpace in use.

We developed ThumbSpace (Figure 1) to address both the reach and accuracy problems that users experience when operating arbitrary touchscreen interfaces with thumbs [13]. Its design is inspired by the substantial body of research focused on the challenge that distance plays in large display interaction. ThumbSpace applies existing techniques for accessing objects out of *arm's* reach on large displays, adapting them to the problem of accessing objects out of *thumb's* reach on handheld devices.

Conceptually, ThumbSpace serves as an absolute touchpad superimposed on the display, to which all screen objects are mapped (Figure 1b). Reach limitations are addressed by allowing users to personalize ThumbSpace's size and placement (Figure 1a), thereby accommodating user differences in hand preference, geometry, motion range,

grip, and use scenario. ThumbSpace supports access to all display objects within an accessible portion of the screen, and applies dynamic visual feedback and selection tuning to alleviate high error rates that are typical in hitting small targets with big thumbs. Finally, ThumbSpace is user-activated and so is unobtrusive when not needed.

ThumbSpace is based on users' natural inclination to touch the interface with their fingers when a stylus is not available. But problems of high visual demand, targeting accuracy, and finger occlusion suggest peripheral hardware solutions might be more appropriate. Blackberry devices, for example, have shown how effective thumb wheels and trackballs can be for controlling a non-touchscreen device.

The first contribution of this paper is an extensive user study comparing the efficacy of touchscreen techniques—including ThumbSpace and Shift [21]—to peripheral hardware input methods. Our results show that touchscreen techniques can offer significant speed improvements over peripheral hardware input methods for dense 2D interfaces. Our results further show that Shift handles occlusion of nearby objects, and suggest that ThumbSpace may provide added benefit for distant objects. Since occlusion and reach can be considered orthogonal problems our study raises the question: *can ThumbSpace and Shift be composed?*

Our second contribution is a follow-up study that addresses this question by investigating the relative benefits of using ThumbSpace *together* with Shift. To more fully quantify ThumbSpace's solution to reach issues, this second study uses a larger screen than in the first (2.8" vs. 3.5"). Our results show that ThumbSpace and Shift compose effectively—their combined use significantly outperforms their individual use—while simultaneously addressing both occlusion and reach. Further, users find the combination intuitive and preferable. With some practice, users naturally choose to use the method that is quantitatively more effective for the given region: ThumbSpace for distant objects, and Shift for near objects.

RELATED WORK

One handed interaction

Research in one-handed mobile interaction has largely focused on specific user tasks, such as media play [18], text entry [22], and application navigation [12]. Karlson et al. [13] looked more generally at human factors requirements for one-handed use of mobile devices, including situational and task preferences for hand use as well as biomechanical limitations of thumbs. They found widespread interest in single-handed operation of personal devices, but that current designs, especially for touchscreen devices, do not accommodate one-handed scenarios well. Their results also show that the areas of a device users are comfortable interacting with vary by user and device size. ThumbSpace is specifically designed to address these findings.

Many current touchscreen interfaces consist of widgets similar in size and function to those featured on a desktop PC. While acceptable for interaction with a 1 mm stylus tip, research suggests touchscreen targets smaller than 9.6 mm [17] can result in unacceptably high error rates when accessed with the thumb due to finger occlusion. Recently, Vogel and Baudisch [21] developed the Shift technique as an improvement over Sears and Shneiderman's offset cursor [20]. The offset cursor couples a selection cursor positioned off the tip of a user's finger with a stabilization algorithm to achieve character-level selection accuracy. The downside to the offset cursor is that users have to aim *below* the intended target. Shift instead allows users to aim directly at the target. After a variable delay, it displays a portal view (callout) of the screen area covered by the user's finger. The callout includes a crosshair cursor to show the position of finger contact, which users can adjust before lifting their finger to perform selection. The delay function that determines when to show the callout is proportional to the size of the target under the finger—short for small targets, and longer for large targets. The result is that Shift is only shown when users are likely to need it, and does not interfere with selection otherwise.

While Shift holds great potential for one-handed selection of targets within reach of the thumb, further investigation is necessary to understand whether pixel-level selection is appropriate under mobile conditions, and whether Shift works equally well for objects along the perimeter of the screen, which occur frequently in today's designs. ThumbSpace, on the other hand, is designed to support targets at edges just as well as those away from the edges. More importantly, Shift was designed for two-handed index finger operation of mobile devices, and so does not address the limitations of thumb reach that ThumbSpace does.

Reaching Distant Objects

ThumbSpace draws its inspiration from table-top and wall-sized displays, which both confront problems with out-of-reach interface objects. A general problem in large display interaction is that the increase in real estate also increases the average distance between on-screen objects. Unfortunately, Fitts' Law dictates that increasing travel distance without a commensurate increase in target size will increase access time. Solutions have thus typically focused on 1) decreasing movement distance to targets and/or 2) increasing target sizes.

Improving target acquisition for mouse-based interaction has often involved clever manipulation of the control-display (CD) ratio. Slowing mouse movement over interaction targets (Semantic Pointing [6]), jumping the cursor to the nearest target (Object Pointing [9]), and predicting the user's intended target (Delphian desktop [2]) are three such examples. The drawback of these techniques is that their effectiveness decreases as the number of nearby objects increases. Other approaches in smart cursor control

make targets easier to hit by increasing the cursor size, such as area cursor [11] and Bubble Cursor [8]. Unfortunately, these techniques are not directly applicable to touchscreen interaction; touching the screen directly means a 1:1 correspondence between motor and display movement, so there is no CD ratio or cursor to manipulate.

Direct screen interaction with fingers or pens is common in tablet, mobile, and wall computing. Techniques to improve object access speed in these arenas have focused on minimizing the movement distance to targets. However, most research that focuses on icon placement or selection [3, 5, 10] is inappropriate for PDA interfaces because drag and drop and object placement are used much less frequently than interactions such as tapping buttons, links, and check boxes. Another approach has been to provide a nearby miniaturized version of the display, or Radar View [15], that can be manipulated directly. However, both the Radar View and the pen-based extension to the Bubble Cursor, the Bubble Radar [1], again focus on object placement tasks, rather than general application interaction.

THUMBSPACE DESIGN

ThumbSpace aims to develop an interaction strategy whereby rich touchscreen interfaces can be controlled with a thumb without sacrificing the expressiveness of information presentation or the efficiency of navigation when two hands are available. The design presented in this section builds off of previous work [13]; we conclude the section by detailing our changes to the original design.

ThumbSpace addresses individuals' variation in thumb agility [13], hand size, strength, and usage by supporting each user's most comfortable and stable grip. Each user defines her own *ThumbSpace*—a region of the touchscreen surface that she considers easy to reach and interact within—in an initial configuration step by dragging her thumb along a diagonal to define the upper right and lower left corners of a rectangular region (Figure 1a). The ThumbSpace region, shown in gray in Figure 1a, is confined to the same aspect ratio as the screen area by honoring the greater of the width or height of rectangle defined by the user's diagonal; constraining ThumbSpace in this way ensures a linear mapping to the original display. All thumb interaction then occurs within this personalized ThumbSpace, which remains fixed across all applications. However, the user can always redefine the ThumbSpace to best match the area she can reach comfortably.

To support access to all interaction targets within the confines of the user-defined region, the ThumbSpace behaves as a Radar View by honoring an input mapping between sub-regions of the ThumbSpace and objects in the original display. Consider the Radar View applied to the Windows Mobile Calendar application in Figure 2b; selecting a date in the miniature Radar View calendar would be equivalent to selecting the associated date in the original display. Since the ThumbSpace may not be required for all interactions, users only launch it when necessary. Because small Radar View objects may be difficult to hit reliably with the thumb, dynamic visual feedback is provided to help users refine their selections.

ThumbSpace Interaction

Object selection with ThumbSpace is performed in four phases: *trigger*, *aim, adjust,* and *lift*. In the *trigger* phase, the user presses the center of the directional pad ("enter") to launch the miniature representation of the display within her personal ThumbSpace (Figure 2b). In practice, any of the re-mappable hardware buttons may be used, but our prototype uses the "enter" button for its positional convenience. The user then *aims* her thumb at the object in the miniature representation associated with the one she wants to select in the main display. Here, the ThumbSpace can be likened to an absolute touchpad; if the user selects correctly, ThumbSpace provides direct access to all screen objects, most importantly those that would otherwise be difficult (e.g., too small) or impossible (e.g., out of reach) to hit directly with her thumb. Once the user's thumb touches a ThumbSpace proxy, the associated display object is visually identified with an object cursor, depicted by a thick

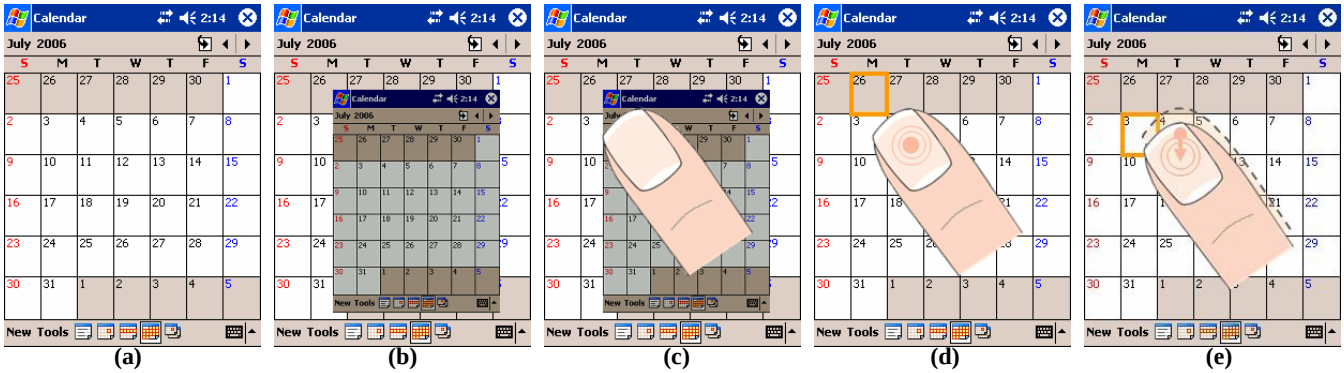


Figure 2. ThumbSpace interaction. (a) A calendar. (b) Pressing "enter" on the directional pad launches a Radar View within the user's predefined ThumbSpace. (c) To select July 3rd, the user aims for the 3rd in the Radar View. (d) Touching the screen highlights the object associated with the Radar View proxy hit (June 26th), and the ThumbSpace disappears. (e) To adjust the cursor the user rolls her thumb downward to move it from June 26th to July 3rd, and lifts her thumb to perform the selection.

orange border, and ThumbSpace disappears (Figure 2d).

The user then *adjusts* their selection. During this phase, ThumbSpace acts like a relative touchpad for controlling the object cursor. If the user rolls or drags her thumb more than a fixed number of pixels up, down, left, or right, the object cursor animates to the closest display object in the direction of movement (Figure 2e). Adjusting in ThumbSpace is similar to Object Pointing [9] in desktop environments, which ignores the white space between interface objects, and instead jumps the mouse pointer to the nearest object in the direction of movement once the pointer leaves an object's border. Our *adjust* strategy differs slightly from Object Pointing because the *adjust* threshold is independent of the display object sizes.

Finally, the user confirms the selection by *lifting* her thumb. This manner of object selection is inspired by Potter's lift-off strategy for touchscreen object selection [19], which allows users to visually confirm and adjust a selection before committing to the action.

The ThumbSpace interface design described here differs from the one reported in [13] in three ways. First, the initial design did not show a Radar View, only a whitewashed region. Second, the use of ThumbSpace was not optional in the original design, but was displayed for use at all times. Last, the aspect ratio of the original ThumbSpace was not constrained to that of the screen area as we now enforce. Our design changes are directly inspired by the results of the formal evaluation of the initial design reported in [13].

STUDY 1: DIRECT VS. INDIRECT INTERACTION

Our goal in the first investigation of the updated ThumbSpace design was to understand the relative advantages of peripheral hardware over touchscreen-based input methods, as they relate to task speed, accuracy and user satisfaction for one handed device use.

Independent Variables

Input Methods

We compared ThumbSpace to the common alternatives used with today's devices. For peripheral hardware we chose the scroll wheel (*ScrollWheel*)—as it is the original distinguishing feature of the non-touchscreen Blackberry devices—and the directional navigation pad (*DPad*), because nearly every cell phone has one. It was clear that for touchscreen interaction, we would compare ThumbSpace to direct touch (*DirectTouch*), since that is how users operate touchscreens one-handed today. We assumed ThumbSpace would offer users more accurate targeting at the expense of speed, so we also chose a technique that specifically addresses targeting accuracy for fingers on touchscreen. Recently, Vogel and Baudisch [21] showed users made fewer errors using their Shift technique over direct touch for hitting targets ≤ 2.9 mm, but not ≥ 4.3 mm. Given the possibility that Shift would help users in hitting the small targets used in our study (3.6 mm^2), we included it as a more competitive variant of DirectTouch.

Mobility

Many previous studies have established the negative impact movement has on mental demand and task performance (e.g., [16]). If our assumption that the hardware input methods are more stable and require less mental, visual, and physical demand than the touchscreen methods, then increased activity would be expected to degrade performance more when using touchscreen methods than hardware methods. To understand this relationship, we studied users performing tasks while both standing and walking. During the walking condition, users chose a comfortable walking pace along a 19' x 7.5' tape figure-8.

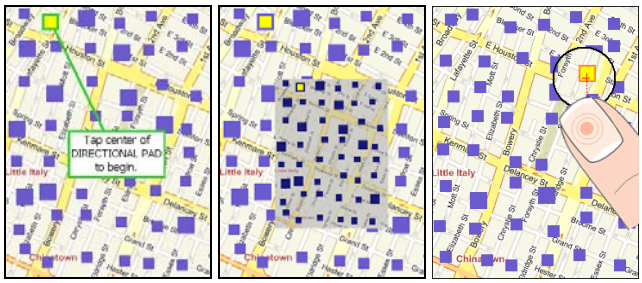
Target Sizes

Our initial study of ThumbSpace [13] confirmed that user performance was independent of target size, so here we chose only a single target size of 20x20 pixels (3.6 mm^2), representative of standard Windows Mobile widgets (e.g., checkboxes: 15 px, buttons: 21 px, text boxes: 19 px).

Tasks

Tasks were based on selection activities that would typically be performed with two hands using a stylus. Since our goal was to understand appropriate one-handed interaction techniques for rich interfaces, we opted for a 2D input space. Potential targets were placed within a 6x8 grid of 40x40 px^2 cells. For analysis purposes, we partitioned the targets into a 3x4 grid of regions, with 4 targets per region (see Figure 5). Regions were labeled "easy to reach" (light gray) or "hard to reach" (dark gray) based on the majority opinion of study participants.

Because the targets were smaller than their assigned cells (20 vs. 40 px), the target for each trial was placed in the center of the designated cell. All other potential targets were randomly assigned one of two sizes (20 px and 13 px, with probabilities 0.2 and 0.8 respectively), and positioned at random locations within their cells. The randomized locations were used to create the illusion of a non-uniform layout space, while the variable-sized objects increased the percentage of the background displayed. The targets were placed on a map background because we expected context would be useful during use of the Shift technique.



(a) **(b)** **(c)**

Figure 3. Study 1 screen shots. (a) Task instruction. (b) ThumbSpace, and (c) Shift representations.

Tasks were presented as a dialog that indicated the trial target and required action to begin the task (Figure 3a). The dialog was placed at the center of the ThumbSpace, unless it overlapped the target, and then was placed either above or below the target. For ScrollWheel, users pressed the scroll wheel to begin; for DPad and ThumbSpace, users pressed the center of the directional pad to begin. For DirectTouch and Shift, users tapped the dialog to begin. To distinguish the goal target from the others, it was filled with yellow.

A rectangular orange cursor was used as the input focus. For ScrollWheel and DPad, the cursor started at the upper left target, since this is a typical home position for cursors in today's interfaces. For DirectTouch, Shift and ThumbSpace, the cursor only appeared once the user had touched the screen to perform a selection. When a target was selected, the input cursor would animate toward the midpoint of the target and vanish to provide visual feedback, and audio sound indicated trial success or failure.

Implementation and Apparatus

The ThumbSpace prototype was developed as an input handler to custom applications written in C# using the PocketPiccolo.NET graphics toolkit [4]. The software ran on a Cingular 8525 PocketPC phone (11.2 x 5.8 x 2.2 cm) with a 2.8" display. We chose this device because it was the only one available that had all the hardware components we wanted to study, thus avoiding a potential confound.

Resistive touchscreens recognize only a single average point from a finger touch, so it is not only hard to predict, but also unstable, due to the varying deformation the finger tip on the screen surface. This problem is well known to make pixel-level targeting difficult, so various approaches have been used to stabilize finger input [20, 21]. We used the recursive dynamic filter of Shift [21], with cutoff frequencies of 5 and 20 Hz interpolated between 54 and 144 mm/s. Given the small size of our targets, Shift was configured to display the callout as soon as the finger touched the screen, but we did not correct for users' perceived contact points as in [21]. For DirectTouch we used a land-on strategy with input stabilization.

Method

The study was a 5 (*Input*: DPad, ScrollWheel, DirectTouch, Shift, ThumbSpace) x 2 (*Mobility*: standing, walking) x 12 (*Region*) x 4 (*Position*) repeated measures within-subjects design. Presentation of *Input* and *Mobility* were counterbalanced across participants, and the 48 *Region* x *Position* trials were randomized within blocks. Dependent variables collected included task time, error rate, satisfaction ratings, and interface preference rankings.

Participants and Procedure

Twelve right handed volunteers (8 male, 4 female) ranging in age from 21 to 31 (μ = 26) were recruited via fliers posted in the Department of Computer Science. Participants received $15 for 1.5 hours of their time.

Before each block of trials, the study administrator explained and demonstrated the input method that would be used. Participants were instructed to select each target as quickly as possible without sacrificing accuracy. Participants then assumed a standing position or began walking the figure-8. After performing half a block of practice trials, users performed the 48 timed tasks. After each block, participants filled out a short subjective questionnaire about the *Input* x *Mobility* condition completed, and then proceeded to the next block.

Participants finished with a "usability" phase, which provided the opportunity to use all input methods with a realistic interface. For this phase users remained standing as the study software presented 10 tasks for each of the 5 Input conditions (DPad, ScrollWheel, DirectTouch, Shift, and ThumbSpace) in that order (roughly familiar to unfamiliar) for a Windows Mobile Calendar interface (Figure 2). Following the usability phase, users ranked the input methods from 1 ("favorite") to 5 ("least favorite") by target type (easy- and hard-to-reach), mobility condition (standing and walking), and expected overall preference once sufficient practice and expertise had been achieved. This last question included an option for using ThumbSpace when desired, and Shift otherwise.

Study 1 Results

Task Times

Task time was measured from the onset of the trial (when the scroll wheel or center of the directional pad was released, or the user's finger was lifted from the task dialog) to the completion of the trial (when the scroll wheel or center of the directional pad was pressed, or the user's finger was lifted from the screen). Task times for each region were determined by averaging the region's four position trials. Trials with selection errors and outliers more than three standard deviations from the mean within each input type were excluded from the aggregation. Huynh-Feldt corrections are used where sphericity did not hold.

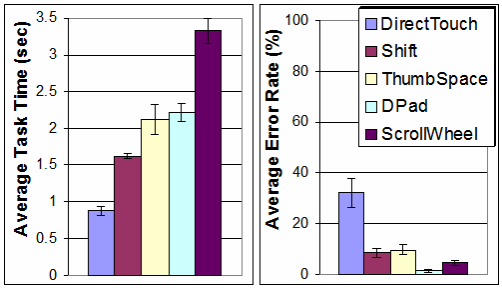


Figure 4. Study 1: average task times and error rate.

We carried out a 5 (*Input*) x 2 (*Mobility*) x 12 (*Region*) repeated measures Analysis of Variance (RM-ANOVA) on the average task time data. Main effects of *Input* $F(4, 24)=67.7$, $p<.001$, and *Region* $F(11,66)=68.5.4$, $p<.001$, were observed. In addition, an interaction of *Input* x *Region* $F(44,264)=50.0$, $p<.001$, was also observed. On average, DirectTouch was significantly faster (865 ms) and ScrollWheel was significantly slower (3311 ms) than all other interaction methods. Shift, ThumbSpace, and DPad did not differ significantly from one another (Figure 4).

Average task times increased with *Region* number because the DPad and ScrollWheel times generally increased with region while task times for the remaining three input methods were basically constant across regions (Figure 5). The increase in time by region for DPad and ScrollWheel reflect the serial nature of those input methods; for example, accessing a target with ScrollWheel required traveling through all lower numbered regions. Figure 5 also illustrates that Shift generally offered a speed advantage over ThumbSpace, but that the two methods were most closely matched in the "hard to reach" regions (1-3).

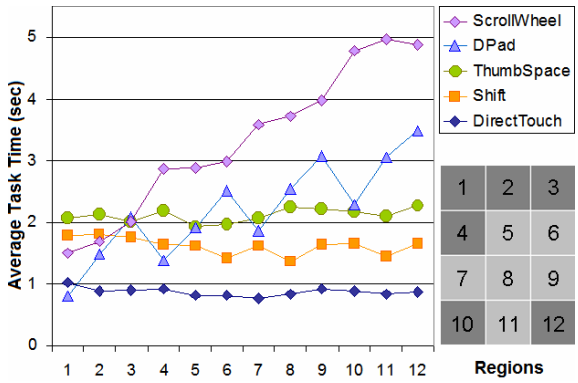


Figure 5. Average task time by region (dark gray = "hard to reach", light gray = "easy to reach) for each input type.

Error Rate

We performed a 5 (*Input*) x 2 (*Mobility*) x 12 (*Region*) RM-ANOVA on the average percent error data. Main effects of input $F(1.4,14.9)=21.2$, $p<.001$, mobility $F(1,11)=5.2$, $p=.04$, and an interaction of input x region $F(44,484)=2.6$, $p<.001$, were found.

On average, users were slightly more accurate while standing than walking (10% v. 12% error). While it is surprising that mobility did not play a bigger role in either speed or error performance, it may be that a) the device was easy to control in one hand; b) the task was not mentally challenging; and c) walking was self-paced and did not demand much visual attention.

With 32% error, DirectTouch was significantly less accurate than any of the other input methods (Figure 4b). DPad (1% error), on the other hand, was significantly more accurate than ScrollWheel (5% error), Shift (8% error) and ThumbSpace (10% error), which as a group were statistically indistinguishable from one another. User comments indicated that ScrollWheel's error rate was likely due to accidental movement when the scroll wheel was pressed to perform selection. Different scroll wheels may vary in their susceptibility to this type of error.

Satisfaction

We performed a 5 (*Input*) x 2 (*Mobility*) x 13 (*Rating*) RM-ANOVA on participant satisfaction ratings, selected from a 7 point scale (1=low, 7=high satisfaction). A main effect of rating $F(12,120)=7.9$, $p<.001$ and an interaction between input and rating $F(48,480)=4.3$, $p<.001$ were found.

Examining the input x rating data, DPad, Shift, and ThumbSpace scores averaged fairly high (majority ≥ 5), and were generally comparable across rating measures. Participants considered DPad more accurate than Shift and ThumbSpace (6.5 vs. 5.4 and 5.1, respectively) and and preferred ThumbSpace less for "easy to reach" targets than Shift and DPad (4.3 vs 5.5 and 5.1, respectively). DirectTouch ratings were similar to DPad, Shift, and ThumbSpace, except it was considered less accurate, more frustrating, less satisfying, and less useful for "hard to reach" than the other three. Finally, ScrollWheel stood out for being less fun, less satisfying, slower, and more physically demanding than the majority of other inputs.

Preference

Based on experience during the study, we asked participants to provide an absolute ranking of the 5 input methods, plus a sixth option of using a Shift+ThumbSpace combination, from 1 ("Best") to 6 ("Worst") for the majority of device interaction. Shift+ThumbSpace was most popular on average (2.1), followed by Shift (2.6), ThumbSpace and DPad (both 3.3), DirectTouch (3.8), and ScrollWheel (5.4). Shift received the most top rankings (4 users), followed closely by DPad and Shift+ThumbSpace (3 users). Notably, 75% of participants ranked Shift+ThumbSpace as their first or second choice, compared to only 42% for Shift, 33% for DPad, and 25% for ThumbSpace.

Discussion

That two touchscreen input methods ranked at the top of the preference ratings despite neither having an advantage in speed or accuracy over DPad bolsters our intuition that users prefer pointing at targets directly for selecting on-screen objects rather than using indirect hardware methods.

Indeed, without software techniques like Shift and ThumbSpace, direct thumb interaction would be unacceptably error-prone for small targets. Using the thumb directly (e.g., DirectTouch) to hit small targets (e.g., ≤ 9mm) is unacceptably error-prone, at least when considering use with standard resistive touchscreen technology. The study data suggest that Shift and

ThumbSpace are two software solutions that can greatly improve user accuracy in hitting small (3.6 mm) targets with the thumb.

Shift and ThumbSpace were statistically indistinguishable from one another in terms of overall speed, accuracy, and satisfaction ratings. However, trends in the data suggest that Shift may offer an advantage over ThumbSpace, since speed, accuracy, and satisfaction ratings were equivalent or higher for Shift than for ThumbSpace in "easy to reach" regions. This result is not surprising since "easy to reach" regions are precisely those for which we assumed ThumbSpace would not be required. Instead, the primary concerns in "easy-to-reach" areas are the imprecision of selection due to the thumb's large contact area with the touchscreens, and the related problem of target occlusion—issues that Shift was specifically designed to address. Since ThumbSpace offered little noticeable advantage in the remaining "hard to reach" regions, it is understandable that users chose Shift over ThumbSpace as the method they would prefer for device interaction in the general case.

STUDY 2: THUMBSPACE VS. SHIFT

The results of our first study leave open several questions regarding the relative benefits of ThumbSpace and Shift. We conducted a second study with the goals of: (1) understanding usage patterns when participants are given their choice of input methods; and (2) evaluating Shift and ThumbSpace on a "large" touchscreen device.

Trends from Study 1 in both user opinion and performance by region suggest that a direct exploration of using Shift *with* ThumbSpace is appropriate. First, on average, participants ranked Shift *with* ThumbSpace (Shift+ThumbSpace) as the most preferred usage option, even though they had not encountered this specific combination during the study. Since target size remained constant, a reasonable interpretation of this finding is that users perceived the two techniques to have different advantages based on the location of the desired target. This is supported by performance trends, which revealed that Shift and ThumbSpace had the most similar target access times in the "hard to reach" regions along the top of the device, the same regions for which ThumbSpace averaged higher accuracy scores. One explanation for this finding is that the top regions represent those that users found more difficult to reach.

Even if ThumbSpace improves over Shift for "hard to reach" regions on large devices, it is possible the advantages would not hold up in practice. Giving users a choice between ThumbSpace and Shift adds a mental calculation to every action, increasing the attention demands of the task, as well as the total selection time [7]. We study the following questions: Does the cost of making a decision result in users making no decision at all, instead using the technique that is less demanding on average for all targets? Are users willing to trigger ThumbSpace? That is, do the comfort and stability offered by a personalized ThumbSpace sufficiently compensate for a longer selection time?

ThumbSpace is designed for instances where portions of the touchscreen are physically difficult to reach, but this may not have been the case for the device used in Study 1. Although it satisfied our requirements for hardware peripherals, the Cingular 8525's 2.8" (7.1 cm) screen is 20% shorter than many common devices that have 3.5" (8.9 cm) displays, including several models of the HP iPAQ (whose screens are 1.0 cm wider and 1.5 cm taller) and the iPhone (whose screen is 0.8 cm wider and 1.8 cm taller). The performance data for the top regions may represent only the borderline cases between those regions which ThumbSpace and Shift are comparable, and it is possible ThumbSpace would provide further advantage over Shift in these regions when used with larger devices. We thus reran the speed and accuracy comparison of ThumbSpace and Shift on a larger, more commonly-sized screen.

Independent Variables

Four input methods (*Input*) were studied: *Shift*, *ThumbSpace*, Shift with ThumbSpace (referred to as *Combined*), and the baseline of using the thumb to hit targets directly without software enhancement (*DirectTouch*). For consistency with the previous study, only one target size (3.6 mm) was considered, and targets were again assigned to one of 12 device regions (*Region*). Finally, to take account for any learning effects within each *Input*, tasks were repeated across two blocks (*Block*).

Implementation and Apparatus

The study was performed on an HP iPAQ 4155 (11.4 x 7.1 x 1.3 cm). The codes for the Shift and ThumbSpace techniques were the same as those used in Study 1. Because the pixel sizes differ between the 8525 (0.18 mm/px) and the iPAQ (0.24 mm/px), pixel values were updated for the Shift camera offset, the Shift camera diameter, and the target sizes to maintain the absolute measurements (in mm) from the Study 1. Otherwise, the only software changes between Study 1 and Study 2 were to the study control software to reflect changes in the study design.

Tasks

For comparability across studies, the selection tasks were the same as those from Study 1, but with a few differences. Because of concerns in Study 1 that the yellow target was too similar to the yellow roads of the map, the trial target was shown in red. In addition, users started the trial timer by pressing the center of the directional navigation pad for all input methods. For Shift, users then aimed for the desired target. For ThumbSpace, users pressed the center button a second time to launch ThumbSpace. For the *Combined* input condition, users chose for each selection whether to aim directly at the target using Shift or whether

to press the center of the directional pad a second time to launch ThumbSpace.

We chose to use the same start procedure for both input methods to: 1) standardize users' grips across trials and input methods; and to 2) abstract away the cost of launching ThumbSpace (which will vary with different trigger mechanisms) by equalizing the movement burden across the input techniques. Results of the study should therefore be interpreted as having used the most optimistic trigger penalty of "a button press".

Method

The study was a 4 (*Input:* DirectTouch, Shift, ThumbSpace, Combined) x 12 (*Region*) x 4 (*Position*) x 2 (*Block*) repeated measures within-subjects factorial design. Presentation of three inputs (DirectTouch, Shift, ThumbSpace) were counterbalanced across participants, but the Combined input was always presented last to ensure participants had encountered both Shift and ThumbSpace prior to their combined use. For each *Input*, users performed 24 practice trials, 2 per regions, followed by two blocks of randomized *Region* x *Position* test trials. Prior to the Combined trials, users performed a "usability" phase in which they were given 2 minutes of self-directed exploration using Combined input for both a Calendar (Figure 2) and Start Menu interface.

Dependent variables collected included task time, error rate, satisfaction ratings, and interface preference rankings. For the Combined input condition, the user input choice for each trial was also recorded.

Participants and Procedure

Twelve right handed participants (5 male, 7 female), ages 18-29 (μ = 21) were recruited from the general campus population. Only one participant studied a technical field. Because of the potential for hand size to bias users toward one input technique or another (e.g., users with small hands might benefit most from ThumbSpace), we aimed for an even distribution of hand size in our participant population. Participants' hands were classified into three broad categories S (*n*=3), M (*n*=5), and L (*n*=4), based on relative comparison (±1 cm) to the administrator's own hand (M), which fit a M/L women's glove.

The procedure for Study 2 was modeled directly after that of Study 1. For the Combined input condition, participants were urged to use whichever technique (Shift or ThumbSpace) they felt offered them the highest speed and accuracy for each trial. Total session time lasted between 45 and 75 minutes for which participants were paid $15.

Study 2 Results

Task Times

Task time was measured from the onset of the trial (when the center of the directional pad was released) to the completion of the trial (when a user lifted her finger from the screen). Task time data were aggregated as in Study 1.

A 4 (*Input*) x 12 (*Region*) x 2 (*Block*) RM-ANOVA was carried out on the average task time data. Since there was no main effect of *Block*, it was removed from further analysis. Main effects of *Input* $F(3, 33)=38.0$, $p<.001$, and *Region* $F(6, 65.9)=4.7$, $p<.001$, were observed. In addition, an interaction between *Input* x *Region* $F(33, 363)=1.8$, $p=.006$, was also found.

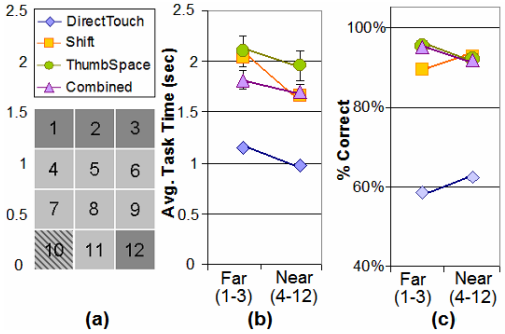


Figure 6. (a) Key and device regions, where dark gray = "hard to reach" and light gray = "easy to reach"; (b) average task times for regions 'Far" (1-3) and "Near" (4-12); (c) average percent correct by region type.

DirectTouch was significantly faster (1,017 ms) on average than the other input methods. Although ThumbSpace was the slowest input method on average (1,993 ms), it was not found to differ significantly from Shift (1,753 ms) or Combined (1,724 ms). Average task times were highest in top regions (1-4) and lowest in central regions (5,7-9,11). To make better sense of the trends in the Input x Region data, we felt it would be useful to aggregate the regions into "Far" and "Near" classes based on user opinion data. The 3x4 grid of Figure 6a depicts the 12 screen regions we analyzed. The color of each region conveys the majority user opinion about whether the region was "hard to reach" (dark gray) or "easy to reach" (light gray); region 10 did not have majority support for either classification. It is reasonable to conclude that the "hard to reach" distinction of the top regions (1-3) was because those regions were "Far" from the trial start point. Similarly, "easy to reach" regions were considered such because they were relatively "Near". Based on distance from the start point, we conclude that regions 10 and 12 are also both "Near", giving the categories Far (1-3) and Near (4-12).

We ran a 4 (*Input*) x 2 (*Distance:* Near, Far) RM-ANOVA on the aggregated region data. In addition to the main effect of *Input* reported above, the analysis revealed a main effect of *Distance* $F(1, 11)=12.5$, $p=.005$ and an interaction of *Input* x *Distance* $F(3,33)=3.4$, $p=.03$. Post-hoc comparison of task times by *Distance* confirmed that users took significantly longer performing tasks in Far regions (1,774 ms) than they did in Near regions (1,570 ms). Considering the *Input* x *Distance* interaction data, Figure 6b suggests

that Shift was more sensitive to differences in task distance than the other three input methods. Planned comparisons of the *Input* x *Distance* data using Shift as the reference input revealed that Shift/Far was significantly slower than Combined/Far (2,037 ms vs. 1,816 ms, p=.018) but that Shift/Near was significantly faster than ThumbSpace/Near (1,659 ms vs. 1,958 ms, p=.013).

Shift was likely faster than ThumbSpace in Near regions due to the occlusion problems that can occur when using ThumbSpace for targets that overlap the user-defined ThumbSpace. However, because Shift/Near did not differ significantly from Combined/Near, users apparently made appropriate decisions when choosing between Shift and ThumbSpace for Near targets. It is somewhat surprising that Combined/Far was significantly faster than Shift/Far, which was faster than ThumbSpace/Far, since participants were restricted to using either Shift or ThumbSpace in the Combined condition. One explanation is that users became more efficient with the techniques over the course of the study, and that this learning effect benefited Combined disproportionately since it was always presented last. This could be true even though our block design did not reveal learning effects in time or error rate *within* input types.

Error Rate

A 4 (*Input*) x 2 (*Region*) RM-ANOVA was performed for average input error data. A main effect of *Input* $F(1.4, 15.2)=40.5$, $p<.001$ and an interaction between *Input* x *Region* $F(33,363)=2.3$, $p<.001$ were observed. Despite the speed advantage offered by DirectTouch, users were significantly less accurate using DirectTouch (39% error) than the other three input methods, which were all equally accurate (7.4 - 8.4% error).

We again ran a 4 (*Input*) x 2 (*Distance:* Near, Far) RM-ANOVA on the aggregated region data to better understand the *Input* x *Region* error trends for "Near" and "Far" regions. As reported above, a main effect of *Input* was found, as well as an interaction effect of *Input* x *Distance* $F(3,33)=4.5$, $p=.009$.

As seen in Figure 6c, participants tended to be more accurate using Shift and DirectTouch in Near regions than Far regions, but the opposite trend was found for ThumbSpace and Combined. Planned comparisons of the *Input* x *Distance* data using Shift as the reference input revealed that Shift/Far was significantly less accurate than ThumbSpace/Far (p=.013) and Combined/Far (p=.027). For Near targets, Shift, ThumbSpace and Combined were indistinguishable from one another. Thus for both distances, Combined supported equivalent or better performance results as Shift and ThumbSpace individually.

Preference

Based on experience during the study, we asked participants to provide an absolute ranking of the four input methods from 1=Worst to 4=Best. On average, users preferred Combined ($\mu=3.7$), followed by ThumbSpace ($\mu=3$), Shift ($\mu=2$) and Direct ($\mu=1.3$). Seventy-five percent of the participants chose Combined as their preferred method, while the remaining 25% chose ThumbSpace.

Input Choice

To understand user strategies for using Shift and ThumbSpace in the Combined condition, we looked at the frequencies with which participants chose Shift vs. ThumbSpace by *Region* for Combined input. Figure 7 shows that ThumbSpace was chosen more often than Shift for targets in the top half of the device (1-6), and that Shift was used increasingly toward the bottom of the device.

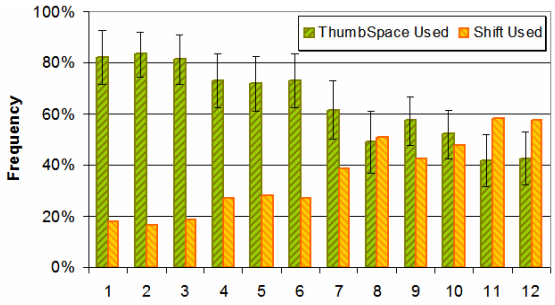


Figure 7. ThumbSpace vs. Shift usage for Combined input.

The fact that participants nearly always used ThumbSpace for Combined input in Far regions (1-3) explains why the error rates of ThumbSpace and Combined matched one another closely in Far regions (Figure 6c). Again, given that Combined enjoyed faster access times over ThumbSpace in Far regions, even though users chose ThumbSpace the majority of the time, suggests a learning effect.

The usage patterns of Shift and ThumbSpace for the Combined condition are intriguing. Users modified their input strategies based on the device region (Figure 7), and the choices users made were "good" with respect to both time and errors (Figure 6). Moreover, the benefit of allowing users to make a real-time choice for *Input* type according to the *Region* of access outweighed any time cost associated with the decision.

Satisfaction

A 4 (*Input*) x 13 (*Rating*) RM-ANOVA was performed on participant satisfaction ratings, selected from a 7 point scale, (1=low, 7=high satisfaction). Main effects of *Input* $F(3, 33)=11.9$, $p<.001$ and *Rating* $F(12,132)=3.9$, $p<.0001$ as well as an interaction between *Input* and *Rating* $F(36,396)=3.9$, $p<.001$ were found.

DirectTouch received significantly lower satisfaction scores on average than ThumbSpace and Combined (4.6 vs. 5.9 and 6.2), while Shift (5.5) was not considered statistically more or less satisfying than the others. It is unsurprising that average scores would differ across satisfaction measures, but in fact most were rated relatively high on

average (≥ 5.1), especially learnability of the techniques (6.5), which rated significantly higher on average than nearly all the other measures. Overall, Shift, ThumbSpace and Combined were rated very similarly to one another across satisfaction measures, except for Comfort and Stability, where Shift rated similarly to DirectTouch, and both were rated much lower than the other two.

CONCLUSION

We performed two extensive user studies to measure the comparative efficacies of peripheral hardware vs. software-based interaction techniques for one-handed touchscreen devices. No one approach proved to be a dominant strategy. In fact, it may not be reasonable to expect a single strategy to fit all legacy applications; our study focused on a dense 2D layout, where direct-touching methods thrived, but for sparse or 1D layouts, peripheral devices may be better suited. Indeed Blackberry devices are a great example of how thoughtful, complementary software designs can render scroll wheels effective and enjoyable. But for legacy applications, one generally does not have the luxury of redesigning the underlying software, thus interface-independent solutions such as ThumbSpace or Shift may prove advantageous.

Participants in our studies preferred the combination of ThumbSpace and Shift. Users quickly and effectively modified their input strategy between these two methods to match the input task: using Shift for near targets and ThumbSpace for far targets. These findings motivate and lend credence to a general approach of composing multiple input methods, each of which is tuned to a different type of task. Perhaps, for instance, peripheral hardware and software-based methods could be used for sparse and dense layouts, respectively. This raises many interesting questions. Of particular relevance to our studies: What roles do the cost of switching methods and variability in realistic mobile scenarios have on user willingness to compose interaction methods for legacy applications?

REFERENCES

1. Aliakseyeu, D., Subramanian, S., Gutwin, C. and Nacenta, M. Bubble radar: efficient pen-based interaction. In *Proc. AVI 2006*, ACM Press (2006), 19-26.
2. Asano, T., Sharlin, E., Kitamura, Y., et al. Predictive interaction using the delphian desktop. In *Proc. UIST 2005*, ACM Press (2005), 133-141.
3. Baudisch, P., Cutrell, E., Robbins, D. C., et al. Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems. In *Proc. INTERACT 2003* (2003), 57-64.
4. Bederson, B. B., Meyer, J. and Good, L. Jazz: An Extensible Zoomable User Interface Graphics Toolkit in Java. In *Proc. UIST 2000)*, ACM Press (2000), 171-180.
5. Bezerianos, A. and Balakrishnan, R. The vacuum: facilitating the manipulation of distant objects. In *Proc. CHI 2005*, ACM Press (2005), 361-370.
6. Blanch, R., Guiard, Y. and Beaudouin-Lafon, M. Semantic pointing: improving target acquisition with control-display ratio adaptation. In *Proc. CHI 2004*, ACM Press (2004), 519-526.
7. Card, S. K., Moran, T. P. and Newell, A. The keystroke-level model for user performance time with interactive systems. *Comm. of the ACM 23*(1980), 396-410.
8. Grossman, T. and Balakrishnan, R. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proc. CHI 2005*, ACM Press (2005), 281-290.
9. Guiard, Y., Blanch, R. and Beaudouin-Lafon, M. Object pointing: a complement to bitmap pointing in GUIs. In *Proc. GI 2004*, Canadian Human-Computer Communications Society (2004), 9-16.
10. Hascoët, M. Throwing models for large displays. In *Proc. HCI 2003*, British HCI Group (2003), 73-77.
11. Kabbash, P. and Buxton, W. The "Prince" technique: Fitts' Law and selection using area cursors. In *Proc. CHI 1995)*, ACM Press (1995), 273-279.
12. Karlson, A. K., Bederson, B. B. and SanGiovanni, J. AppLens and LaunchTile: two designs for one-handed thumb use on small devices. In *Proc. CHI 2005*, ACM Press (2005), 201-210.
13. Karlson, A. K. and Bederson, B. B. ThumbSpace: generalized one-handed input for touchscreen-based mobile devices. *Proc. INTERACT 2007*, Springer (2007), 324-338.
14. Kristoffersen, S. and Ljungberg, F. "Making place" to make IT work. In *Proc. SIGGROUP 1999*, ACM Press (1999), 276-285.
15. Nacenta, M. A., Aliakseyeu, D., Subramanian, S. and Gutwin, C. A comparison of techniques for multi-display reaching. In *Proc. CHI 2005*, ACM Press (2005), 371-380.
16. Oulasvirta, A., Tamminen, S., Roto, V. and Kuorelahti, J. Interaction in 4-sec. bursts: the fragmented nature of attentional resources in mobile HCI. In *Proc. CHI 2005*, ACM Press (2005), 919-928.
17. Parhi, P., Karlson, A. K. and Bederson, B. B. Target Size Study for One-Handed Thumb Use on Small Touchscreen Devices. In *Proc. MobileHCI 2006*, ACM Press (2006), 203-210.
18. Pirhonen, P., Brewster, S. A. and Holguin, C. Gestural and audio metaphors as a means of control in mobile devices. In *Proc. CHI 2002*, ACM Press (2002), 291-298.
19. Potter, R. L., Weldon, L. J. and Shneiderman, B. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In *Proc. CHI 1988*, ACM Press (1988), 27-32.
20. Sears, A. and Shneiderman, B. High-precision touchscreens: design strategies and comparisons with a mouse. *Int. J. Man-Mach. Stud. 34*, 4 (1991), 593-613.
21. Vogel, D. and Baudisch, P. Shift: a technique for operating pen-based interfaces using touch. In *Proc. CHI 2007*, ACM Press (2007), 657-666.
22. Wigdor, D. and Balakrishnan, R. TiltText: using tilt for text input to mobile phones. In *Proc. UIST 2003*, ACM Press (2003), 81-90.