ThumbSpace: Generalized One-Handed Input for Touchscreen-Based Mobile Devices

Amy K. Karlson and Benjamin B. Bederson

Human-Computer Interaction Lab, Computer Science Department
University of Maryland, College Park, MD 20742, USA
{akk, bederson}@cs.umd.edu

Abstract. In this paper, we present ThumbSpace, a software-based interaction technique that provides general one-handed thumb operation of touchscreen-based mobile devices. Our goal is to provide accurate selection of all interface objects, especially small and far targets, which are traditionally difficult to interact with using the thumb. We present the ThumbSpace design and a comparative evaluation against direct interaction for target selection. Our results show that ThumbSpace was well-received, improved accuracy for selecting targets that are out of thumb reach, and made users as effective at selecting small targets as large targets. The results further suggest user practice and design iterations hold potential to close the gap in access time between the two input methods, where ThumbSpace did not do as well as direct interaction.

Keywords: ThumbSpace, one handed mobile interaction.

1 Introduction

With the number of cell phones out-shipping more traditional computers by more than 4:1 in 2006 [17, 28], the emergence of the phone as a ubiquitous personal accessory is clear. Traditionally, cell phones have served as little more than wireless telephones, while mobile information management has been reserved for PDAs and laptops. But with rapid advances in processing power, storage capacity, and connectivity, these different device types are converging into the "smartphone": a feature-rich, Internet-enabled mini-computer. As the numbers and types of devices grow, so will opportunities to explore a wider range of interaction methods, exemplified by Apple's recent announcement of the multi-touch capacitive touchscreen iPhone [25]. Devices will remain small because mobility will remain a driving factor. Users in diverse mobile environments are bound to be visually and mentally distracted [16] and to have one or both of their hands frequently occupied. Given these trends, today's touchscreen software designs often do a poor job supporting mobility because the majority require two-handed stylus input.

Interfaces that accommodate single-handed interaction can offer a significant benefit by freeing one hand for the physical and intellectual demands of mobile tasks [20]. Surveys [14] confirm that users would generally prefer to use touchscreens with one hand when possible, but hardware and software designs of today's devices

C. Baranauskas et al. (Eds.): INTERACT 2007, LNCS 4662, Part I, pp. 324–338, 2007.

typically offer no such support. Styli are often required for touchscreen interaction because their software interfaces are composed of targets that are either too small [21] or ill-positioned to be hit reliably with the thumb. One solution is to build interfaces that explicitly accommodate thumb interaction by ensuring that all targets are thumb sized and within thumb reach [15]. Yet this "lowest common denominator" approach to interface design is unlikely to catch on. This is because small screens so severely constrain information presentation already that placing further limitations on visual expressivity can hurt the design in other ways. For example, increasing target sizes to accommodate thumbs means fewer targets can be shown per screen, and so will require more screens to present the same amount of information. This can slow down access to information, even when two hands *are* available. These observations, together with the reality that existing mobile UI toolkits include only small, stylus-oriented widget palettes, have led us to consider an alternative strategy.

We present ThumbSpace, a novel interaction technique to facilitate the thumb accessibility of rich touchscreen interfaces. ThumbSpace combats both the reach and accuracy problems that users experience when using thumbs on touchscreens. It works like an absolute-position touchpad superimposed on a portion of the standard touchscreen interface. Users can access all locations on the screen by interacting with only a sub-region of the display. Thumb reach limitations are addressed by allowing users to personalize the size and placement of ThumbSpace, thereby accommodating individual differences in hand preference, geometry, motion range, grip, and use scenario. As a result of decoupling input space and output space, ThumbSpace eliminates the dependence of target size on thumb size, and so alleviates the accuracy issues involved with directly hitting small targets with big thumbs. ThumbSpace does this without constraining the complexity of applications' interfaces; in fact, ThumbSpace is designed to be *application independent*.

Given the dynamic environments and varied tasks of mobile computing, user choice about the number of hands to use is expected to be fluid. Effective mobile interfaces will therefore maximize presentation power and interaction efficiency regardless of hand availability. With ThumbSpace, touchscreen devices can continue to take full advantage of the available display real estate for rich presentation and interaction with two hands while at the same time supporting one-handed scenarios.

In the remainder of this paper we describe the design and implementation of ThumbSpace and our evaluation of ThumbSpace versus direct thumb interaction for accessing targets of varying size, density, and location.

2 Related Work

2.1 One Handed Interaction

The physical and attentional demands of mobile device use were reported early on for fieldworkers [16, 22]. This resulted in specific design recommendations for minimal-attention and one-handed interfaces [22] which, though well suited to the directed tasks of fieldwork, do not generalize to the varied and complex personal information management tasks of average users. Since then, research in one-handed device interaction has largely focused on either the hardware or the tasks supported.

Technology-oriented efforts have investigated the potential benefits of accelerometer-augmented devices [12] and touchscreen-based gestures [7, 15] for one-handed interaction. Other research has focused on enabling one-handed support for specific tasks, including media control [23] and text entry [30]. But overall, there has been relatively little focus on thumb interaction.

Karlson et al. [14] looked more generally at human factors requirements for one-handed interaction with personal mobile devices, including situational and task preferences for hand use as well as biomechanical limitations of thumbs. They found that in addition to the common practice of one-handed phone use, there is wide interest in single-handed use of (generally larger) touchscreen-based PDAs, but that current designs do not accommodate one-handed scenarios well. Their results further suggested that users were more comfortable when interaction could be limited to a sub-region of the device surface, preferably toward the center of the device. ThumbSpace's design addresses these findings directly.

Others have studied how physical characteristics of the thumb interact with touchscreen technology. Parhi et al. [21] investigated the effect of target size on input accuracy when operating a PDA one handed with the thumb. Previously, others had determined appropriate target sizes for pen-based interaction on mobile touchscreen devices [18], and index fingers for interaction on desktop-sized displays [8, 26], but pens and index fingers are smaller than thumbs and their use scenarios differ from the mechanic constraints of holding the screen with the same hand used for interaction.

Current touchscreen interfaces consist of widgets similar in size and function to those featured on a desktop PC. While acceptable for interaction with a 1mm stylus tip, such targets are much smaller than the average thumb pad in at least one dimension, making reliable access difficult or impossible. Recently, Vogel and Baudisch [29] presented the Shift technique as an improvement over Sears and Shneiderman's offset cursor [27], both of which address issues of occlusion during finger selection of small touchscreen targets. While Shift holds great potential for one-handed selection of targets within reach of the thumb, further investigation would be necessary to understand whether pixel-level selection is effective under mobile conditions, and whether Shift works equally well for objects along the perimeter of the screen, which occur frequently in today's designs. 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 seeks to address.

2.2 Reaching Distant Objects

ThumbSpace draws its inspiration from the observation that large table-top and wall-sized displays both confront issues with out-of reach interface objects. A general problem for 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. We classify these further into indirect and direct interaction methods.

Indirect Interaction. 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 [10]), 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 [13] and Bubble Cursor [9].

Direct Interaction. Direct screen interaction with fingers or pens is common in tablet, mobile, and wall computing, but the absence of a remotely controlled cursor means there is 1:1 correspondence between motor and display movement, and thus no CD ratio to manipulate. Since increasing target widths has generally been achieved via CD manipulation, techniques for direct input have focused on reducing the movement distance to targets. Drag-and-pop [3] reduces an object's drag distance by drawing full-sized proxies of targets closer to the moved item. The Vacuum widget [5] allows users to select the display sector of interest before moving object proxies to within reach. Drag-and-throw and push-and-throw [11] reduce movement distance by virtually extending the reach of the finger or pen.

A final class of direct interaction techniques offers users miniaturized, nearby versions of the entire display area for manipulating distant objects. With Radar View [19], the miniature representation appears as users begin drag operations. The Bubble Radar [1] extends the Bubble Cursor to pen-based computing, using a Radar View representation and dynamic expansion of the pen's activation area. ThumbSpace uses a similar approach to the Radar View, while addressing occlusion and small displays.

3 ThumbSpace

Our goal with ThumbSpace has been to develop an interaction strategy whereby rich touchscreen interfaces can be effectively controlled with a thumb, without sacrificing the expressiveness of information presentation or the efficiency of navigation when two hands are available.

3.1 ThumbSpace Design

Karlson et al.'s investigation of thumb movement when using mobile devices with one hand [14] found that users were more comfortable interacting with some surface regions than others. Although the centers of devices were generally easy to access, and corners and edges were generally hard to access, gradations in opinion between these extremes indicated that participants were not in perfect agreement. Indeed, individual differences in hand size, thumb length, agility, and strength can all affect thumb range of motion. Furthermore, differences in use scenario will affect the freedom users have to change grip while maintaining control of the device, and so will also impact how well users can access different regions of the device surface.

Given such variations in thumb ability, hand preference, and mobile usage requirements, the first principle of ThumbSpace is to support the user's most comfortable and stable grip. Each user therefore defines his or her own ThumbSpace - a region of the touchscreen surface that is easy to reach and interact with. To

configure ThumbSpace, the user is instructed to drag her thumb along a diagonal to define the upper left and lower right corners of a rectangular region (Fig. 1a). All thumb interaction then occurs within this personalized ThumbSpace, which remains fixed (but reconfigurable) across all applications.

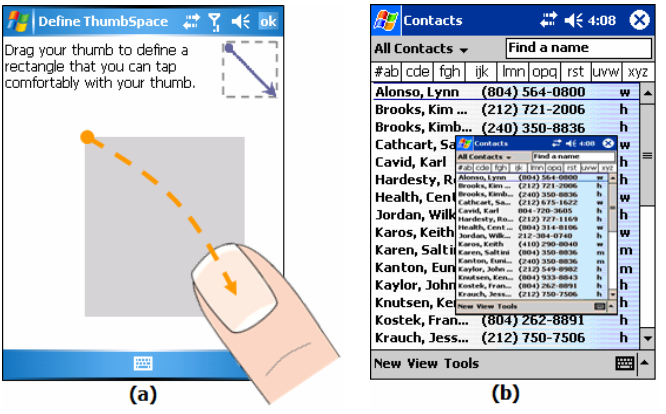

Fig. 1. (a) Defining the ThumbSpace. (b) The ThumbSpace as a traditional Radar View.

To support access to all interaction targets within the confines of the ThumbSpace, the user-defined region behaves as a type of a Radar View for the display. Traditionally, a Radar View is a miniaturized representation of a large display that serves as a within-reach proxy for out-of-reach objects; interactions upon objects within the Radar View are propagated to the associated objects in the original display, hereafter referred to as the "DisplaySpace". This approach has proven successful for accessing distant windows and icons on large displays [19], but hasn't been applied to small screen interaction, which presents novel challenges. Consider, for example, that a straightforward implementation of a Radar View for the Windows Mobile Contacts application is shown in Fig. 1b. This approach has several problems: 1) the Radar View representation occludes a large number of DisplaySpace objects; 2) the Radar View proxies are unreadable; 3) the detailed Radar View representation contributes to unacceptable visual clutter; and 4) the Radar View proxies are far too small to access reliably with the thumb.

To address problems (1-3), we avoid using a miniature representation. Instead, we offer only a whitewashed region to suggest where the user should focus her attention. Fig. 2a shows this representation of ThumbSpace, which overlays the application at all times. Even without the miniature displayed, ThumbSpace retains the spirit of the Radar View by honoring an input mapping between the ThumbSpace and the DisplaySpace. ThumbSpace is partitioned so that each object in the DisplaySpace is associated with a sub-region (*proxy*) in the ThumbSpace; tapping a proxy in ThumbSpace selects the assigned object in the DisplaySpace. If ThumbSpace were to represent a linearly scaled DisplaySpace (as in Fig. 1b), the partition of ThumbSpace into DisplaySpace proxies would be that shown in Fig. 2b. Yet the ThumbSpace partition is not required to be a scaled representation of the DisplaySpace; in section 3.4 we discuss how different partitioning strategies may improve user performance.

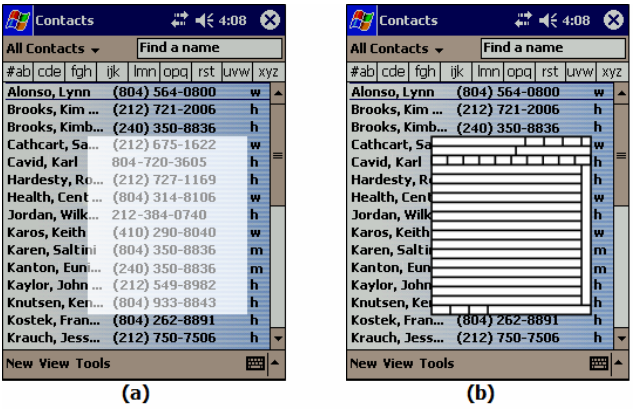

Fig. 2. (a) The actual ThumbSpace representation. (b) One possible partitioning of the ThumbSpace into proxies for DisplaySpace objects.

3.2 Using ThumbSpace

The final challenge ((4) above) in using a miniature representation as an interaction medium is that the proxies can be too small to hit reliably with a big thumb, even when users can *see* the representation. ThumbSpace introduces further uncertainty because it provides no visual cues for how its sub-regions map to DisplaySpace objects. ThumbSpace manages these uncertainties during interaction by providing dynamic visual feedback.

Object selection in ThumbSpace is performed in three phases: *aim*, *adjust,* and *lift*. The *aim* phase requires users to have formed a mental model of the input mapping between the ThumbSpace and DisplaySpace. Again, the simplest assumption is that the ThumbSpace represents a linear scaling of the DisplaySpace. Based on her mental model, the user makes an initial guess about the sub-region of ThumbSpace that corresponds to the intended target, and *aims* at the sub-region with her thumb (Fig. 3a). In the *aim* phase, ThumbSpace can be likened to an absolute touchpad - if the user guesses correctly, ThumbSpace provides touchdown access to objects that would otherwise be difficult (e.g., too small) or impossible (e.g., out of reach) to hit directly with her thumb.

As soon as the user's thumb touches a ThumbSpace proxy, the associated DisplaySpace object is highlighted using an object cursor, depicted as a thick orange border. The user then enters the *adjust* phase for fine-tuning the selection. During the *adjust* phase, ThumbSpace acts like a relative touchpad for controlling the object cursor. If the user rolls or drags her thumb more than 10 pixels up, down, left, or right, the object cursor animates to the closest DisplaySpace object in the direction of movement (Fig. 3b). In the *adjust* phase, ThumbSpace interaction is similar to Object Pointing [10], which ignores the whitespace between interface objects in desktop-based mouse interaction by jumping the cursor to the closest object in the direction of movement when the mouse cursor leaves an object's bounds. The *adjust* strategy

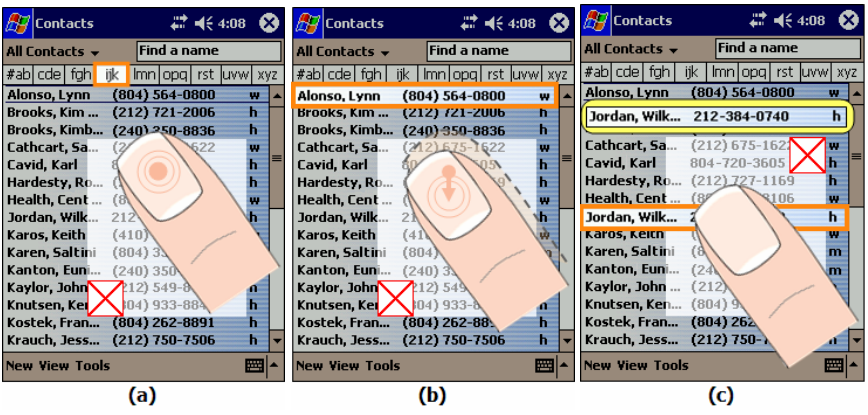

Fig. 3. Selecting objects with ThumbSpace. Assuming the user wants to select the first name in the list, she first (a) *aims* at the probable ThumbSpace proxy for 'Alonso'; (b) the initial ThumbSpace point of contact maps to 'ijk' so the user *adjusts* the intended target by dragging her thumb downward. The user confirms the selection by lifting her thumb, or cancels the selection by dragging her thumb to the X before lifting; (c) ThumbSpace occlusion correction.

differs slightly from Object Pointing because *adjust* does not take into account proxy widths and heights, only the delta of thumb movement.

Finally, the user confirms the selection by *lifting* her thumb. This manner of object selection is inspired by the lift-off strategy for touchscreen object selection developed by Potter [24], which allows users to visually confirm and adjust a selection before committing to the action. To allow users to cancel a selection, the user can drag her thumb over the red X, which appears during interaction in either the upper right or lower left corner of ThumbSpace, whichever is furthest from the first point of contact.

A ThumbSpace proxy may overlap its corresponding DisplaySpace object, or even appear below it, which may cause the thumb to occlude the DisplaySpace object. At these times, the selected item is highlighted in the original display, and an unobstructed representation appears at a fixed location above the ThumbSpace to improve the visibility of the current selection (Fig. 3c). Again, this solution can be considered an adaptation of the take-off touchscreen selection strategy [24], which avoided finger occlusion by placing the selection cursor above the user's finger. Note that, by displaying the obstructed object above ThumbSpace, and by allowing the user to specify her ThumbSpace (Fig. 1a), we support both left- and right-handed users.

A video demonstration of ThumbSpace definition and interaction can be viewed at http://www.youtube.com/user/thumbspace.

3.3 Interactions Supported

Traditional PDAs support three primary interactions: *tap* for object selection, *tap-drag-release* to drag objects and cancel selections, and *tap-and-hold* to trigger context-sensitive menus. The ThumbSpace *aim-adjuist-lift* interaction technique repurposes the traditional *tap-drag-release* for object selection. While object selection

is required far more frequently than the other two in standard interfaces, failure to support them would certainly limit the utility of ThumbSpace for general-purpose use.

For widgets that support more specialized interaction than tapping, we propose a 2 step process: 1) the *aim-adjust-lift* sequence gives focus to the widget or opens an associated context-sensitive menu (leaving the red cancel box visible; and 2) any drag operation then performed in the ThumbSpace moves or scrolls the widget in sync with the thumb until the user lifts her thumb to perform the final selection, or selects the red cancel box to abort the operation. For example, selecting a combo box via *aim-adjust-lift* gives focus to the widget and opens the list of available items; dragging the thumb in ThumbSpace moves the highlight among the combo box list; lifting the thumb either selects an item in the list or cancels the selection if performed over the red cancel box. In this paper we explore the viability of object selection only, and will investigate the other interaction types in future work.

3.4 Deriving ThumbSpace Proxies

Just as manipulations of the CD ratio for mouse input has been shown to improve object selection in desktop systems under certain conditions [6, 10], our strategy for mapping ThumbSpace proxies to DisplaySpace objects also has the potential to influence system usability. An interface with a sparse object layout would suffer from a strict linear mapping between DisplaySpace and ThumbSpace because a large percentage of ThumbSpace pixels would be "dead" in that they would not be associated with any DisplaySpace object. This approach would waste scarce input space and result in unnecessarily small proxies that would be difficult to hit.

Ideally, the ThumbSpace to DisplaySpace mapping would instead optimize the size and placement of ThumbSpace proxies to achieve the following goals: (1) The ThumbSpace should be fully partitioned into proxies so that every pixel in ThumbSpace corresponds to some DisplaySpace object. Our goal is to maximize the sizes of proxies and thus improve user accuracy in the *aim* phase of object selection. (2) "Landmark" objects in DisplaySpace, such as those in the four corners and center of the display, should retain their relative positions within the ThumbSpace; ideally "landmark" proxies would be relatively larger than those for similarly sized objects, again to improve user accuracy in the *aim* phase of object selection. (3) ThumbSpace proxies should retain the same up-down and left-right positions to one another as do their corresponding DisplaySpace objects. This goal allows for efficient and intuitive targeting in the *adjust* phase of object selection.

For our initial exploration of the ThumbSpace technique, we define the mapping between ThumbSpace proxies and DisplaySpace objects by hand, and will develop generalized proxy partitions in future design iterations.

3.5 Implementation

As a real-world input system, ThumbSpace will need to cooperate with a PDA's operating system in order to capture and reinterpret thumb events for stylus-oriented interface designs. However, to first establish its viability, we have implemented ThumbSpace as a generalized input handler to custom interfaces written in C# (.NET

Compact Framework) for Windows Mobile Pocket PCs using the PocketPiccolo open source graphics toolkit [4].

4 User Study

To understand the potential of ThumbSpace in terms of usability, interaction effectiveness, and user satisfaction, we conducted a quantitative study to compare ThumbSpace to direct thumb interaction for object selection tasks.

4.1 Tasks

Because object selection via tapping is the predominant interaction type on PDAs today, we focused on basic target selection for our study. We hypothesized that users would be able to select objects faster and more accurately with one hand overall when using ThumbSpace than when using only their thumbs directly. In particular, we predicted that the relative benefits of ThumbSpace would vary by target size, distance, and overall target density as follows:

Target Size: In determining the target size conditions, we were mainly concerned with the difference in performance between selecting a target that is too small to be hit reliably with the thumb and one that is large enough to be hit. Based on previous investigations on appropriate target sizes for one-handed thumb interaction [21], we chose two sizes to study: *small* (20 pixels = 4.8 mm), and *large* (40 pixels = 9.6 mm). Since ThumbSpace decouples input and output space, we did not expect target size to impact selection accuracy, as is prevalent in direct thumb selection.

Target Distance: When operating a device with one hand, users generally find some surface areas of a device easier to access than others [14]. Using guidelines from [14], we classified targets into two distance categories: *near* (comfortable to reach) and *far* (uncomfortable to reach). Due to thumb reach constraints, we expected users would perform near tasks better than far tasks when directly tapping. Since ThumbSpace is within user reach by design, we did not expect target location to affect performance for ThumbSpace.

Target Density: Target density plays a role in ThumbSpace because it affects the size of the ThumbSpace proxies, and thus user accuracy in the *aim* phase of object selection. We chose to study 2 densities: *sparse* (proxies were 80 px^2, or 19.2 mm^2) and *dense* (proxies were 40 px^2, or 9.6 mm^2). With ThumbSpace, we expected users would be slower when hitting targets in dense conditions due to the need to perform more corrective moves in the *adjust* phase of object selection than under sparse conditions. In contrast, we did not expect density to impact direct thumb interaction.

The target configurations studied are shown in Fig. 4a. Tasks began with a message box indicating the target to select (Fig. 4b). Tapping the message box started the task timer, and if appropriate, the user's ThumbSpace would appear (Fig. 4c). Once a target was selected, a "success" or "error" sound provided accuracy feedback. The study software ran on an HP iPAQ h4155 Pocket PC measuring 4.5x2.8x0.5 inches with a 3.5 inch, 240 x 320 resolution screen, calibrated prior to the study.

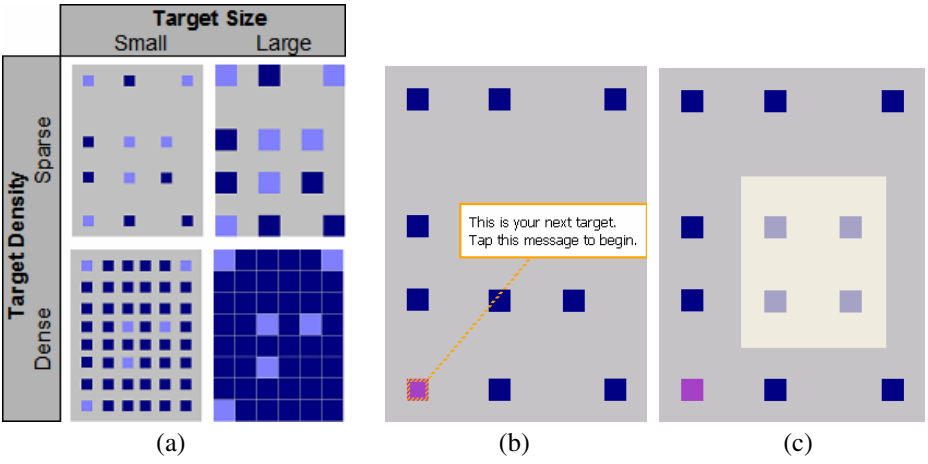

Fig. 4. (a) Target configurations for the user study. (b,c) Example study task.

4.2 Method and Measures

The study was a 2 (input type: ThumbSpace v. direct interaction) x 2 (size: small v. large) x 2 (density: sparse v. dense) x 2 (distance: near v. far) x 3 trial within subject design. For each input type, users performed the tasks in 5 blocks, for a total of 240 trials. Input type was counterbalanced across participants, and tasks were randomized within blocks. Dependent variables collected during the study included task time, error rate, user satisfaction ratings on a 7 point scale, and overall input preference.

4.3 Participants and Procedure

We recruited right-handed participants via fliers posted in our Department of Computer Science, and through flier distribution in an undergraduate HCI course for non-majors. Sixteen respondents (8 female, 8 male) participated in the study, ranging in age from 18 to 28 with an average age of 23 years of age. Participants received $10 for approximately 45 minutes of their time.

The study began with an introduction to each input type. Participants first read a description of direct thumb interaction, followed by practice with a single block of tasks. Participants then read a description of ThumbSpace, defined a personal ThumbSpace (Fig. 1a), and practiced with two blocks of tasks; the administrator demonstrated ThumbSpace for the first 5 of these tasks. After the practice phase, the study proper began: users completed 5 task blocks for one input type, followed by a usability questionnaire, then repeated the process with the second input type. After this official data collection phase, users completed a "usability" phase, which provided the opportunity to use the techniques with a real interface. With each input type, users performed 20 object selection tasks, presented as in Fig. 4b, for a Windows Mobile Start Menu and Calendar program.

Users completed the study by recording their preferred input type for targets by condition (small, large, near, and far), their preferred input type overall, and expected input preference after gaining sufficient expertise using ThumbSpace.

4.4 Results

Task Times. We performed a 2 (input type: ThumbSpace v. direct interaction) x 2 (size: small v. large) x 2 (density: sparse v. dense) x 2 (distance: near v. far) repeated measures Analysis of Variance (RM-ANOVA) on mean selection time data. Significant main effects of input type, size, and density were observed, as were significant interactions of input x size, input x density, and input x distance (Table 1).

Table 1. Significant results for Task Times and Percent Correct data

		input	size	density	input x size	input x density	input x dist.
Task Times	$F_{1,15}$	26.8	11.0	62.0	5.2	32.8	12.8
	p	<.001	.005	<.001	.04	<.001	.002
Percent Correct	$F_{1,15}$	-	18.0	5.1	69.4	5.3	9.8
	p	-	<.001	.04	<.001	.04	.007

On average, direct interaction was faster than ThumbSpace (811 ms v. 2068 ms), large targets were easier to select than small targets (1367 ms v. 1512 ms), and sparse targets were easier to select than dense targets (1312 ms v. 1566 ms). The two way interactions involving input type confirmed two of our hypotheses: (1) target size impacted user performance when using direct interaction (large = 691 ms v. small = 930 ms), but not when using ThumbSpace; and (2) target density impacted user performance more when using ThumbSpace than using direct interaction, presumably because users performed more corrective moves in the *adjust* phase of selection for dense targets than sparse targets. However, our prediction that distance would not impact access time with ThumbSpace was incorrect. In fact, users were significantly slower selecting near targets with ThumbSpace than far targets (Fig. 5a). We surmise that most users' ThumbSpaces directly overlapped the near targets, resulting in considerable thumb occlusion for during these tasks. Our study setup exacerbated the problem because targets were not visually distinct from one another, which diminished the efficacy of ThumbSpace's occlusion solution.

Percent Correct. We carried out a 2 (input type) x 2 (size) x 2 (density) x 2 (distances) RM-ANOVA on percent correct selection data. Significant main effects of target size and density were observed, as well as significant interactions of input x size, input x density, and input x distance (Table 1).

Overall, participants were more accurate selecting large targets than small targets (.95 v. .90), and more accurate selecting from among sparse targets than among dense targets (.93 v. .91). As with the time results, target size significantly influenced error rate for direct interaction (large = .87 v. small = .99) but not for ThumbSpace interaction. Target density only influenced user performance when using ThumbSpace (sparse = .94 v. dense = .90), but had no impact during direct interaction. Assuming that users perform fewer corrections in the *adjust* phase for

sparse targets than for dense targets, then it is reasonable that the success rate would be higher for sparse targets. We saw a similar pattern for input x distance with error rate as selection time: with direct interaction, users were more accurate hitting near targets, whereas with ThumbSpace users were more accurate selecting far targets. In fact, users were significantly more accurate hitting far targets with ThumbSpace (.94) than with direct interaction (.92), as shown at the bottom of Fig. 5a.

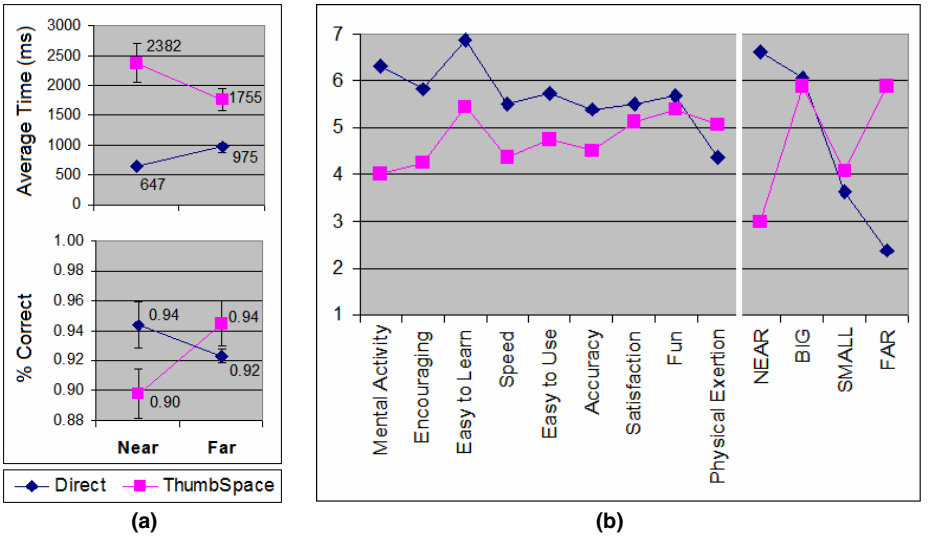

Fig. 5. (a) Average selection time and percent correct for near v. far targets by input method. Note the scale for percent correct does not start at 0. (b) User satisfaction (7 = most satisfied).

Satisfaction: User satisfaction data correlated with the performance data, with users on average preferring direct interaction (μ = 5.7) to ThumbSpace (μ = 4.8). Participants generally found ThumbSpace more mentally challenging than direct interaction, rated the two methods similarly in terms of task satisfaction and fun, but found ThumbSpace required less physical exertion than direct interaction. ThumbSpace received its lowest scores for accessing near targets, but was much preferred over direct interaction for selecting far targets. Only a quarter of the users stated a preference for using ThumbSpace for one-handed interaction given their comfort level at the end of the study, but 69% stated they would prefer ThumbSpace to direct interaction for general one handed device use if given sufficient practice.

5 Summary and Future Work

ThumbSpace is the result of our exploration of applying techniques from large-display interaction to small, mobile devices. We have shown that a direct application of existing techniques is not enough; issues such as occlusion, limited real estate, and users' varying biomechanics must be taken into account. ThumbSpace addresses these

issues with specific, intuitive mechanisms, including a user-defined input space (Fig. 1a) and a consistent *aim-adjust-lift* interaction technique (Fig. 3).

The results from our user study indicate that applying large-display interaction techniques to mobile devices is promising and that ThumbSpace is a strong step in this direction. In particular, we are encouraged that ThumbSpace made progress toward both of our design goals: (1) "decouple target size from thumb size": ThumbSpace makes users as effective at selecting small targets as large targets, evidenced by the fact that neither speed nor error data were affected by target size; (2) "improve selection for targets that are out of thumb reach": users accessed far targets more accurately using ThumbSpace than direct interaction. Finally, users thought ThumbSpace held promise; ThumbSpace was given relatively high ratings on a 7-point scale for task satisfaction (5.1), fun (5.4) and learnability (5.4), and the majority of users indicated that ThumbSpace would be preferable to direct interaction after sufficient practice.

However, ThumbSpace does not appear to be an immediate, realistic replacement for direct thumb interaction for one-handed device use. Our findings suggest that both user practice and strategic design iterations may have a strong impact on closing the performance gap between the two input types. Further, it seems that the mental demand of learning ThumbSpace could not be overcome within the time constraints of the study, and was at least partially to blame for lower user performance.

Lastly, some design problems must be addressed for ThumbSpace to be an effective one-handed interaction solution. ThumbSpace was designed to improve one-handed access to far targets, since near targets should already be easy to tap. However, our current ThumbSpace design makes near targets harder to select than far targets. One possible solution is to allow users to launch ThumbSpace on-demand for far targets, and provide only occlusion avoidance techniques for easy to reach objects. The problems of speeding the user learning curve and reducing the mental demand of ThumbSpace requires further exploration, but offering visual cues about the proxy to object mappings, and constraining users' ThumbSpace definitions to those with the same aspect ratio as the DisplaySpace, are promising future directions.

Acknowledgments. Many thanks to Dave Levin for his helpful edit suggestions.

References

1. Aliakseyeu, D., Subramanian, S., Gutwin, C., Nacenta, M.: Bubble radar: efficient pen-based interaction. In: Proceedings of AVI ’06, pp. 19–26. ACM Press, New York (2006)
2. Asano, T., Sharlin, E., Kitamura, Y., Takashima, K., Kishino, F.: Predictive interaction using the delphian desktop. In: Proceedings of UIST ’05, pp. 133–141. ACM Press, New York (2005)
3. Baudisch, P., et al.: Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems. In: Interact ’03, pp. 57–64. Springer, Heidelberg (2003)
4. Bederson, B.B., Meyer, J., Good, L.: Jazz: An Extensible Zoomable User Interface Graphics Toolkit in Java. In: Proceedings of UIST ’00, pp. 171–180. ACM Press, New York (2000)

5. Bezerianos, A., Balakrishnan, R.: The vacuum: facilitating the manipulation of distant objects. In: Proceedings of CHI '05, pp. 361–370. ACM Press, New York (2005)
6. Blanch, R., Guiard, Y., Beaudouin-Lafon, M.: Semantic pointing: improving target acquisition with control-display ratio adaptation. In: Proc. CHI'04, pp. 519–526. ACM Press, New York (2004)
7. Brewster, S.A., Lumsden, J., Bell, M., Hall, M., Tasker, S.: Multimodal 'Eyes-Free' interaction techniques for mobile devices. In: Proc. CHI '03, pp. 473–480. ACM Press, New York (2003)
8. Colle, H.A., Hiszem, K.J.: Standing at a kiosk: effects of key size and spacing on touch screen numeric keypad performance. Ergonomics 47(13), 1406–1423 (2004)
9. Grossman, T., Balakrishnan, R.: The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In: Proceedings CHI '05., pp. 281–290. ACM Press, New York (2005)
10. Guiard, Y., Blanch, R., Beaudouin-Lafon, M.: Object pointing: a complement to bitmap pointing in GUIs. In: Proc. GI '04. Canadian Human-Computer Comm. Soc, pp. 9–16 (2004)
11. Hascoët, M.: Throwing models for large displays. In: British HCI Conference. British HCI Group, pp. 73–77 (2003)
12. Hinckley, K., Pierce, J., Sinclair, M., Horvitz, E.: Sensing techniques for mobile interaction. In: Proceedings of UIST '00, pp. 91–100. ACM Press, New York (2000)
13. Kabbash, P., Buxton, W.: The "Prince" technique: Fitts' Law and selection using area cursors. In: Proceedings of CHI '95, pp. 273–279. ACM Press, New York (1995)
14. Karlson, A.K., Bederson, B.B., Contreras-Vidal, J.L.: Understanding One Handed Use of Mobile Devices. In: Lumsden, J. (ed.) Handbook of Research on User Interface Design and Evaluation for Mobile Technology, Idea Group Reference (2007)
15. Karlson, A.K., Bederson, B.B., SanGiovanni, J.: AppLens and LaunchTile: two designs for one-handed thumb use on small devices. In: Proc. CHI '05, pp. 201–210. ACM Press, New York (2005)
16. Kristoffersen, S., Ljungberg, F.: "Making place" to make IT work: empirical explorations of HCI for mobile CSCW. In: Proceedings of SIGGROUP '99, pp. 276–285. ACM Press, New York (1999)
17. Milanesi, C., et al.: Market Share: Mobile Devices by Technology, Worldwide, 4Q06 and 2006, Gartner, Inc. (2007)
18. Mizobuchi, S., Mori, K., Ren, X., Yasumura, M.: An empirical study of the minimum required size and the number of targets for pen on the small display. In: Proceedings of MobileHCI '02, pp. 184–194. Springer, Heidelberg (2002)
19. Nacenta, M.A., Aliakseyeu, D., Subramanian, S., Gutwin, C.: A comparison of techniques for multi-display reaching. In: Proceedings of CHI '05, pp. 371–380. ACM Press, New York (2005)
20. Oulasvirta, A., Tamminen, S., Roto, V., Kuorelahti, J.: Interaction in 4-second bursts: the fragmented nature of attentional resources in mobile HCI. In: Proceedings of CHI '05, pp. 919–928. ACM Press, New York (2005)
21. Parhi, P., Karlson, A.K., Bederson, B.B.: Target Size Study for One-Handed Thumb Use on Small Touchscreen Devices. In: Proceedings of MobileHCI '06, pp. 203–210. ACM Press, New York (2006)
22. Pascoe, J., Ryan, N., Mores, D.: Using while moving: HCI issues in fieldwork environment. Trans. on Computer-Human Interaction 7(3), 417–437 (2000)

23. Pirhonen, P., Brewster, S.A., Holguin, C.: Gestural and audio metaphors as a means of control in mobile devices. In: Proceedings of CHI '02, pp. 291–298. ACM Press, New York (2002)
24. Potter, R.L., Weldon, L.J., Shneiderman, B.: Improving the accuracy of touch screens: an experimental evaluation of three strategies. In: Proc. CHI '88, pp. 27–32. ACM Press, New York (1988)
25. Robinson, S.: Apple iPhone: Catalyst for Capacitive Touchscreen-Only Phones to Balloon to 115 Million Units within Two Years, Strategy Analytics (2007)
26. Sears, A., Revis, D., Swatski, J., Crittenden, R., Schneiderman, B.: Investigating touchscreen typing: the effect of keyboard size on typing speed. Behaviour & Information Technology 12(1), 17–22 (1993)
27. Sears, A., Shneiderman, B.: High-precision touchscreens: design strategies and comparisons with a mouse. International Journal of Man-Machine Studies 34(4), 593–613 (1991)
28. Shao, J.: Personal Computer Quarterly Statistics Worldwide By Region: Final Database, Gartner, Inc. (2007)
29. Vogel, D., Baudisch, P.: Shift: A technique for operating pen-based interfaces using touch. In: Proceedings of CHI '07, ACM Press, New York (2007)
30. Wigdor, D., Balakrishnan, R.: TiltText: using tilt for text input to mobile phones. In: Proceedings of UIST '03, pp. 81–90. ACM Press, New York (2003)