Escape: A Target Selection Technique Using Visually-cued Gestures

Koji Yatani[1], Kurt Partridge[2], Marshall Bern[2], and Mark W. Newman[3]

[1]Department of Computer Science
University of Toronto
www.dgp.toronto.edu
koji@dgp.toronto.edu

[2]Computing Sciences Laboratory
Palo Alto Research Center, Inc.
www.parc.com
kurt@parc.com, bern@parc.com

[3]School of Information
University of Michigan
www.si.umich.edu
mnewman@umich.edu

ABSTRACT

Many mobile devices have touch-sensitive screens that people interact with using fingers or thumbs. However, such interaction is difficult because targets become occluded, and because fingers and thumbs have low input resolution. Recent research has addressed occlusion through visual techniques. However, the poor resolution of finger and thumb selection still limits selection speed. In this paper, we address the selection speed problem through a new target selection technique called Escape. In Escape, targets are selected by gestures cued by icon position and appearance. A user study shows that for targets six to twelve pixels wide, Escape performs at a similar error rate and at least 30% faster than Shift, an alternative technique, on a similar task. We evaluate Escape's performance in different circumstances, including different icon sizes, icon overlap, use of color, and gesture direction. We also describe an algorithm that assigns icons to targets, thereby improving Escape's performance.

Author Keywords

Target selection, finger gesture, touch screen, mobile device

ACM Classification Keywords

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – Input devices and strategies, Interaction styles.

INTRODUCTION

Everyone wants a mobile device to be small—until they start to use it. Tiny screens are hard to see, and tiny user interfaces are hard to control.

Many mobile devices have a screen that a user can control by touch. Although these devices can also be controlled by a stylus, many people prefer to use their thumbs [10].

CHI 2008, April 5–10, 2008, Florence, Italy.

A recent research study of thumb use recommended that on-screen targets be no smaller than 9.2mm wide [13]. Below this size, performance begins to degrade when the user tries to select a target with a thumb since thumb-presses are simply too large and too variable to give an accurate selection point. Although users can accurately select smaller targets by another method, such as by using a stylus, they lose the ease of thumb-based interaction. Furthermore, it is often not practical to make a target large enough for thumb-based interaction because larger targets occupy more space, leaving less room on a small display for other targets and information.

Although users cannot accurately select targets smaller than 9.2mm with direct thumb touch, techniques such as Offset Cursor [15] and the more recent Shift [17] improve selection accuracy by helping users refine their initial selection position. Originally designed for fingertip operation, these techniques overcome the general problem of digit occlusion by offsetting the cursor from the selection point (Offset Cursor), or by displaying an inset of the selection region (Shift).

While these approaches are more accurate for smaller targets, they are also slower. When selecting a 12 pixel (2.6 mm) target with a fingertip, participants using Shift made only about 20% as many errors as normal pointing, but took 70% longer [17].

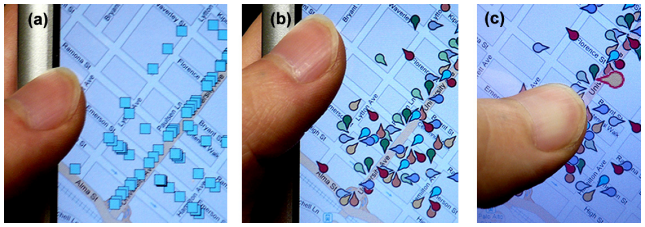

Figure 1. (a) It is difficult to select a target when it is surrounded by other selectable objects. (b) The icons in *Escape* indicate finger gestures that disambiguate the selection. (c) A thumb tap followed by a gesture (without the release of the thumb) enables a user to select the target quickly and correctly even when it is small or occluded by other objects.

In this paper, we present "Escape," an accurate and fast selection technique for small targets. Unlike conventional selection, in which the contact point alone determines the target, in Escape the contact point need only lie close to the target. If the point can be unambiguously associated with a single target, the user can then lift their finger or thumb and the selection is made. However, if multiple targets are near the contact point, the user instead gestures in a direction suggested by the icon, thus disambiguating the selection (see Figure 1). In our experiments, for targets between six and twelve pixels wide, target selection using Escape is on average at least 30% faster than using Shift, without a significant difference in error rate.

Escape is presented in the context of thumb-based one-handed target selection on a map application for a mobile touch screen. However, Escape could also be useful in other circumstances such as two-handed operation, general user interface widgets, and non-mobile devices.

ESCAPE INTERACTION

Figure 2 shows in more detail how the Escape selection technique works. The user presses his thumb close to (but not necessarily on) the target icon (more specifically, within the area of a "Parhi" box, explained later), and then makes a linear gesture in the direction that the target icon points. Icons can be packed close together, but are still easily distinguished as long as each icon is well-separated from the other icons that have the same gesture. We say that no two identical icons can share the same "Parhi box," in reference to the previously-mentioned finding by Parhi *et al*. [13] that, to keep error rates low, targets should be at least 9.2mm x 9.2mm square. Although the minimum-area shape of such a target is, in practice, not likely a box, we ignore this distinction here.

An advantage of this approach is that it relies less on the user's visual feedback loop. In traditional target selection, the user moves a cursor closer to the desired target, looks to see if the cursor lies within the target, and then repeats these steps until the cursor is properly positioned. This process can take several hundred milliseconds for small targets.

With Escape, the user need only use their visual ability to recognize the position and appearance of the icon. After this, they need only tap their thumb in the 9.2 mm box around the icon position and make the gesture. Their visual system is used only to guide their thumb to the first point of contact, not to direct a cursor after the initial contact. Also, there is no need for the user to reorient to any other visual changes, such as the position of the Offset Cursor or the dynamically appearing inset of Shift. Explained in terms of user interaction techniques, Escape replaces the visually demanding and time-consuming target-selection task that follows the initial thumb press with a much coarser selection task followed by a crossing task [1] of making a sufficiently-long gesture.

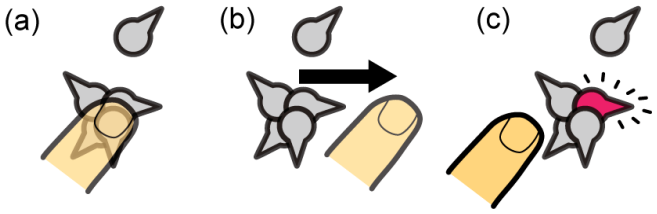

Figure 2. The Escape target selection technique. (a) The user presses her thumb near the desired target. (b) The user gestures in the direction indicated by the target. (c) The target is selected, despite several nearby distracters.

RELATED WORK

Target Selection on a Touch Screen

Much prior work has addressed how to improve selection on touch screens. Albinsson and Zhai [2] propose two techniques for very precise positioning. In *Cross-Keys,* the user adjusts the cursor position by tapping soft arrow keys displayed around the cursor. In *Precision-Handle,* the user controls a handle whose motions are scaled down to control the cursor more precisely. They show that although these techniques are faster than Offset Cursor for one-pixel targets, they are slower for eight-pixel (3.2 mm) targets.

Benko *et al*. [3] investigate a precise pointing technique in the domain of two-handed interaction. The primary finger performs an initial selection while the secondary finger improves the precision by controlling an in-situ zoom or the properties of the cursor. Their techniques outperform earlier techniques, particularly in selection accuracy for objects smaller than eight pixels (4.8 mm). Despite the advantages, using two hands is impractical in many contexts involving small mobile devices.

Another important issue for touch screens is occlusion. A target is usually occluded by the thumb or finger during selection. Earlier work that addresses occlusion is the aforementioned Offset Cursor technique (also called *Take-off*) by Potter *et al*. [15]. This study uses desktop touch screens, so the results are for finger selection rather than thumb selection, however the techniques are generally applicable. In Offset Cursor, the cursor is placed above the actual position of the finger, and the object under the cursor is selected when the finger is released. Offset Cursor is less error-prone than alternative approaches, but its selection time is significantly longer than a technique that simply selects the first item that the user's finger contacts. Although the reasons are not analyzed in detail, this appears to happen because Offset Cursor requires that the user spend time correcting her finger position before selecting the target object.

Sears and Shneiderman [16] explore a stabilization technique that makes Offset Cursor significantly both faster and more accurate for targets less than four pixels wide. However, differences in the experimental setup make direct

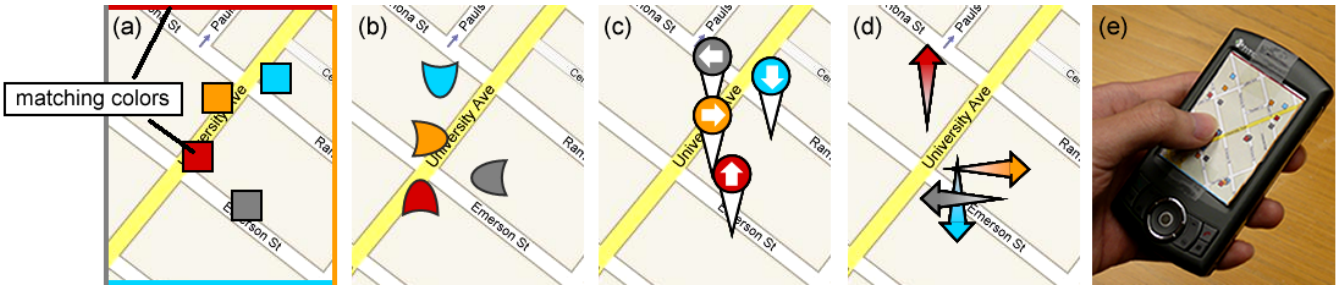

Figure 3. Sample icon designs from the first pilot study. Designs were evaluated by showing study participants paper prototypes taped to the screen of a functional mobile device.

comparison of these results to mobile device studies difficult.

Mobile Touch Screen Target Selection

Shift [17] addresses the disadvantages of Offset Cursor and adapts the technique to a mobile device. When using Offset Cursor, the user cannot know the precise position of the cursor until he presses the screen. Furthermore, always offsetting above the finger makes it impossible to select a target at the bottom of the screen. Shift copies the area occluded by the finger to an inset above, left, or right of the contact position. By not offsetting the cursor, and by keeping the selection point under the user's finger, the user can "aim for the actual target." A user study showed that Shift was more accurate then direct touch for targets 12 pixels wide and less, and faster than Offset Cursor for targets 48 pixels and wider.

However, despite these benefits, both Shift and Offset Cursor's selection times are significantly slower than those of direct pointing for targets 12 pixels wide, and also appear to be slower for targets six pixels wide, although high error rates make significance unclear.

Karlson *et al.*'s Thumbspace [9] presents a way to control a large mobile screen from a smaller input area using only a thumb. The input area shows a miniaturized version of the larger screen, but rather than naively magnifying the user's motions, only the initial press is mapped to the original screen position. Pre-release motions then use the object pointing technique [6] to jump between selectable targets. Although ThumbSpace offers more accurate selection of small objects and reachability of distant objects, selection time is slower than direct pointing.

BubbleCursor [5] also employs object-based selection, and improves upon the general idea by changing the cursor size dynamically. While BubbleCursor could be adapted for use in a mobile touch screen device, it does not address selection among overlapped objects, thus limiting the density of selectable targets that can be displayed.

One-Handed Mobile Touch Screen Gesture Operation

While gesture-based techniques have been heavily explored for both pen- and mouse-based interfaces [7, 14], they have not been explored as much for one-handed interaction using fingers or thumbs. Gesture-based interaction has been used for thumb-based navigation among applications on a handheld [11], and has been adopted commercially by the iPhone and HTC Touch. However, none of these systems have used strokes to assist target selection.

PILOT STUDIES TO INFORM DESIGN

As we considered how to implement Escape we recognized that design decisions about icon design, icon size, number of gestures, and type of gestures could significantly affect target density and usability measures such as selection time, error rate, and learnability. To determine good values for these parameters and improve Escape's overall design, we conducted three pilot tests.

First pilot study: Preliminary Icon Design

Early in the design process we conducted a quick low-fidelity pilot test to help us assess the intuitiveness and recognizability of four initial icon designs for Escape. The designs are shown in Figure 3.

One preliminary design for Escape (Figure 3a) used gestures in four directions, and color, not shape. This design has an advantage in an ultra-dense cluster of icons: even one pixel of color may be enough to suggest an icon's presence and how to select it. Only four gesture directions are used, and a one-pixel border around the screen edge is colored to teach users the proper color/direction association.

The half-moon icon (Figure 3b) combines color and shape, and does not require the border. The pushpin (Figure 3c) resembles existing map icons. The arrow (Figure 3d) shows direction more clearly and contains a gradation, which we thought would improve recognition. We showed both monochrome and color versions of this icon to participants.

Method

Two people participated in this pilot test. Each was presented with a handheld device to which was taped color printouts of each of the five designs (see Figure 3e). The printouts showed both isolated and overlapping icons. We explained Escape and asked each participant to individually "select" 5-10 icons of each design. We then asked their impressions of the strengths and weaknesses of each design.

Figure 4. The icon designs of the second pilot study. (a) A beak icon in which a beak and a color represent the direction of a gesture; (b) A pushpin icon; (c) A two-beak icon. The two beaks of each two-beak icon represent a multi-level gesture (e. g., going downward and then going leftward).

Results

Our pilot users preferred the colored arrows (Figure 3d), followed by the pushpin (Figure 3c), and the half-moon (Figure 3b). However, although the arrow design seemed clear and easy to learn, the clutter introduced by overlapping arrows was distracting. Color helped resolve the clutter, although it did not seem to help identify gesture direction. The pushpin icons were favored for their familiarity, but one participant suspected that they might require more visual attention to identify the gesture direction. The half moon icons were also easy to learn and easier to see in overlapping conditions, but their "bluntness" made them less recognizable when isolated.

We formed two conclusions from these observations. First, shape and color should be used redundantly, since shape best indicates the direction of a gesture and color helps distinguish icons. (We revisit the issue of color's value in the formal experiments.) Second, an icon should be both simple, to reduce clutter, and asymmetric, to distinguish itself.

Second Pilot Study: Icon Size, Density and Gesture Type

The goal of the second pilot study was to decide the final icon design. Based on the experience from our first pilot study, we devised a new "beak" design that combined the best features of the colored squares, half-moons, and arrows (Figure 4a). We also retained the push-pin icon (Figure 4b) for its intuitiveness.

In selecting the final icon design, we also considered icon size. Our goal was to find icons large enough to see and recognize, but small enough to allow a large number of targets on the screen at one time. To explore this, we constructed 20 frames with either 2, 4, 8, 16, or 24 icons per Parhi box. Icons were either 8, 12, 16, or 24 pixels wide.

However, a single Parhi box can only support as many targets as there are distinct gestures. Adding more straight-line gestures makes more targets available per unit area, but also increases gesture error rates, as shown by studies of pie menus [12]. To explore one alternative, we constructed a two-level gesture design (Figure 4c). To select such an icon, the user would first move in the direction of the top beak, and then in the direction of the bottom beak.

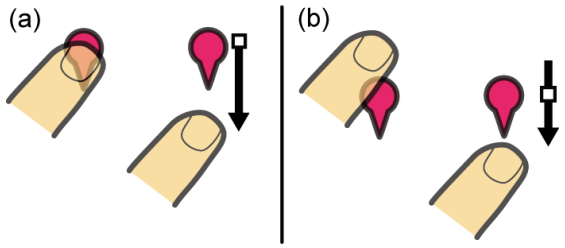

Figure 5. Determination of the gesture location. (a) The initial contact point determines the location of the gesture. (b) An alternative in which the gesture midpoint determines the gesture location.

Method

Eight new participants were presented with color printouts taped to a device as in the first pilot study. Each participant was asked which design they preferred. To determine whether the icons were recognizable, we asked participants to count the number of icons that they could easily see.

Results

The participants found that the single-level beak icon was more distinguishable than the pushpin icon. The two-level beak icons were difficult or impossible to recognize when the number of the icons in a Parhi box was more than eight. This led us to decide not to pursue the two-level design further, and to choose the basic beak icon.

In assessing density, participants found the smallest beak icons (8 pixels wide) in the densest box (24 icons) to be both countable and identifiable. Also countable were 12-pixel beak icons packed 16 to a box. Because such small icons supported the greatest target density and seemed feasible for Escape, we focused our efforts on smaller-sized icons in later studies.

Third Pilot Study: Gesture-to-Icon Distance Metric

Our third pilot study investigated two approaches to associating gestures with targets. Our first design matched the gesture with the icon whose beak direction matched the gesture direction and whose center lay nearest to the gesture's *start point*. The second design matched a gesture with an icon based on the gesture's *midpoint.* (Figure 5). This latter technique is similar to crossing-based interaction [1] and gives a user more freedom in her gesture starting point, because she can compensate by extending or truncating her gesture.

We ran a pilot test with two users, this time with the operational prototype described in the next section. Our results did not show a noticeable improvement in performance or error rate, and appeared not to be

Figure 6. The experimental task. (a) The start button and the crosshair to indicate the target position; (b) The target and distracters; (c) Visual feedback during the selection.

immediately intuitive, so we stuck with the original approach based on gesture start point.

IMPLEMENTATION

The Escape prototype was implemented as a C# Windows Mobile application. It used the 8-directional beak icons shown in Figure 4a.

For comparison, we reimplemented Shift [17] as an alternative selection technique. We used the same escalation time for each target size (0, 5, 39, and 240 milliseconds for 6, 12, 18, and 24 pixel targets, respectively). The correction vector was tuned and fixed before the experiment. For the dynamic low pass filter, we found that a cut-off frequency of 3 and 14 Hz interpolated between 18 and 48 mm/sec worked best for our device.

Validating our implementation of Shift was complicated by differences in experimental conditions. The details are covered in the discussion section of Experiment 1.

EXPERIMENT 1: COMPARISON WITH SHIFT

Procedure

Before starting each block of tasks, participants performed a practice set that used the same tasks as the test session. The participants could continue to practice until they were comfortable. Participants were allowed to take a break between blocks. The entire experiment took between 30 and 60 minutes, depending on the participant's performance.

The task, shown in Figure 6, was designed to estimate the time to select a target that the user had already identified from a crowded field of other targets. In each task, a crosshair and large pink start button appeared on the screen. The distance between the crosshair and the center of the start button was 98 pixels. The participant tapped on the start button, and the target appeared, surrounded by seven distracter targets. Two distracter targets were positioned to meet the *Exposure* variable (explained later), and the others were located randomly within the Parhi box as long as they did not overlap the target. Times were measured between the tap of the start button and the selection of a target.

In both conditions, targets turned yellow when selected. The Escape condition also provided a legend to match icon color with gesture direction.

Participants identified the correct target to select by its position in the exact center of the screen, where the crosshair had been. Additionally, for Shift, the target was red while the distracters were blue. For Escape, the target had a light blue outline. These clues minimized the amount of time participants needed to determine the right target to select (an artifact of our experimental setup), while accounting for the time spent in thumb movement as well as icon identification (realistic time costs that the experiment was designed to measure).

Independent Variables

The independent variables were *Technique* (Shift or Escape); *TargetSize* (the size of the target: 6, 12, 18, or 24 pixels), and *Exposure* (the fraction of the target that was visible: 0.25, 0.5, 0.75, or 1). When the target was partially occluded in the Escape condition, the beak was always exposed. We found that icon arrangement algorithms (described later in this paper) allow icons to be chosen in such a way to satisfy this assumption for most target arrangements. (We investigated the effects of beak occlusion in Experiment 2.) Finally, we studied thumbnail use (not thumbpad use) because of the low sensitivity of the touch screen.

Eight different *Directions* were used for Escape; for Shift, the condition was simply repeated. *Technique* was counterbalanced, and the order of *Direction* was randomized. Eight blocks were used, four per technique, with each combination of *TargetSize* and *Exposure* presented twice in each block. Thus, there were 2 (*Technique*) * 4 (*TargetSize*) * 4 (*Exposure*) * 8 (*Direction)* = 256 trials per participant.

Hypotheses

(H1) Escape would be faster than Shift, and less affected by target size.

(H2) Shift would have fewer errors on smaller targets and more occluded targets, since the icon's gesture would be difficult to determine using Escape.

(H3) *Exposure* would influence the performance of both techniques, but in different ways. Shift's performance would be affected by the smaller target size. Escape's performance would be affected by the increasing difficulty of recognizing the icon.

Apparatus

The experiment was conducted on a T-Mobile Wing, which has a 41 x 54 mm, 240 x 320 pixel display. Its effective resolution is 5.9 pixel/mm.

Participants

Twelve people (nine male and three female) from our institution participated. We recruited only right-handed participants to simplify the study. All participants had some experience with a touch screen mobile device. Each participant was given a $20 gift card.

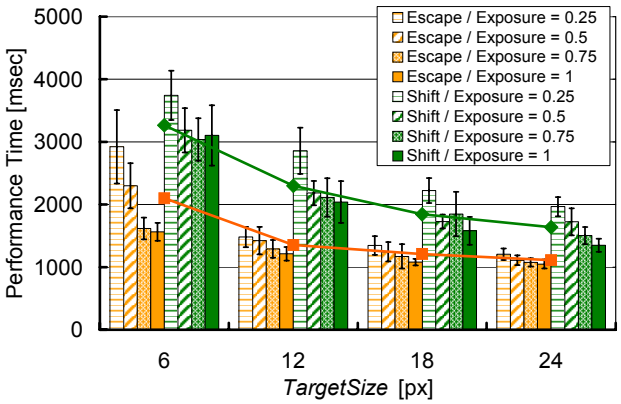

Figure 7. The mean performance time for *Technique* X *TargetSize* X *Exposure* using thumbnails in Experiment 1. Lines connect averages across all exposures for each technique. Escape is significantly faster than Shift, although performance degrades for heavily occluded, very small icons. In this and all later charts, error bars represent 95% confidence intervals.

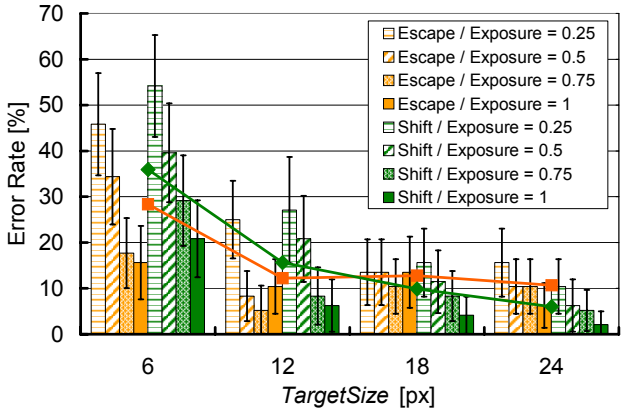

Figure 8. The mean error rate for *Technique* X *TargetSize* X *Exposure* in Experiment 1. No significant difference for technique was found in error rate.

EXPERIMENT 1 RESULTS

Selection Time

Figure 7 shows the mean performance time by *Technique, TargetSize,* and *Exposure*. We performed a within-subjects analysis of variance (ANOVA) for *Technique* X *TargetSize* X *Exposure*, and a main effect was found for each: *Technique* ($F_{1,3070}$=325.12, p<0.001), *TargetSize* ($F_{3,3068}$=166.38, p<0.001), and *Exposure* ($F_{3,3068}$=33.59, p<0.001). The significant interactions were *Technique* X *TargetSize* ($F_{7,3076}$=10.5, p<0.001), *TargetSize* X *Exposure* ($F_{15,3069}$=2.81, p<0.01), and *Technique* X *TargetSize* X *Exposure* ($F_{31,3053}$=2.44, p<0.01). Tukey's post-hoc pairwise comparison showed that Escape was significantly faster than Shift in all *TargetSizes*.

Error Rate

Figure 8 shows the mean error rate. An ANOVA test for *Technique* X *TargetSize* X *Exposure* showed a main effect for *TargetSize* ($F_{3,1532}$=65.62, p<0.001), and *Exposure* ($F_{3,1532}$=29.72, p<0.001), but not *Technique*. The significant interactions were *TargetSize* X *Exposure* ($F_{15,1520}$=4.39, p<0.001) and *Technique* X *TargetSize* X *Exposure* ($F_{31,1504}$=4.39, p<0.001).

EXPERIMENT 1 DISCUSSION

The results support hypothesis H1. Figure 7 shows that Shift's task time increases as the exposed target size decreases, as would be expected from Fitts' Law [4]. Escape's task time also increases, however at a different rate. This effect arises because even as target icons become harder to identify, the physical target size remains one Parhi box. The effect on performance is shallower than Shift's up to 50% occlusion of 6-pixel icons, where both task time and error rates jump because the icons are hard to see.

Our results do not support H2; we did not find a significant difference between Shift and Escape's error rate. In this regard, Shift performed better than we expected.

Our results support H3 partly. Shift's performance was affected by *Exposure* because the effective target size shrinks. In Escape, *Exposure* influences the performance more when *TargetSize* is smaller. We look into this effect more deeply in Experiment 2.

Although Escape's task time outperformed our reimplementation of Shift in this study, Escape's performance is only marginally better than the original published results [17] for targets 12 pixels or less, and is somewhat worse for targets 18 pixels or greater. However, the original results were for finger and fingernail use. To better establish the differences between the techniques, and to validate our implementation of Shift, we reran Experiment 1 with four participants for only the Shift condition, and instructed them to use the fingernails of their index finger. In our implementation, the actual target sizes were 25% smaller than those in [17]. The target distance in our device was 17.6 mm, compared to 28.8 mm in [17]. Therefore, we decided to compare our implementation against the implementation in [17] based on a Fitts' Law prediction for the time-to-first press for a target 28.8 mm away, given our original data for a target 17.6 mm away. Figure 9 shows the comparison of the two Shift techniques with the estimate of what our results would be for the further target. For this task, there is a close agreement between our implementation of Shift and the original results. This leads us to conclude that the slower performance of Shift in Experiment 1 relative to the published results primarily reflects the difference in using the thumbnail instead of the fingernail.

EXPERIMENT 2: OCCLUSION, COLOR, AND DIRECTION

We conducted a second study to explore variations on the basic Escape idea. One question was the extent to which color helped identify target direction under different

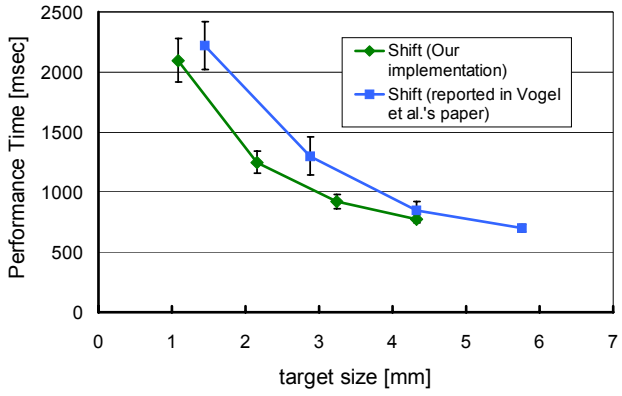

Figure 9. Performance comparison between our reimplementation of Shift and the results reported by Shift [15] for Experiment 1 using fingernails and adjusted for different target distances.

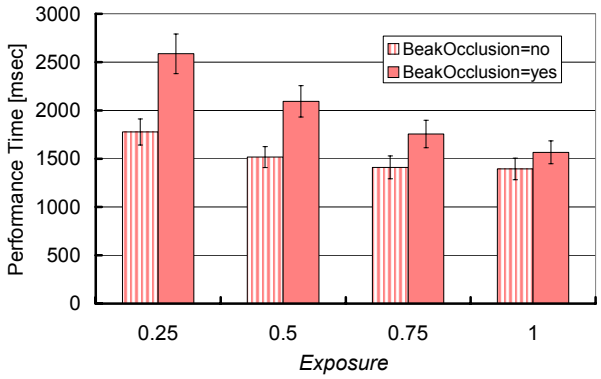

Figure 10. Performance as a function of *Exposure,* averaged over *TargetSize* in Experiment 2, showing the differences in *BeakOcclusion*. When the beak is exposed, performance only degrades if 25% or less of the icon is visible.

occlusion conditions. Although our pilot study results had suggested that direction-indicating colors improved performance, some applications might prefer to use color for other purposes, so we wanted to quantify the benefit. Another question was how occlusion of the beak affected performance differently from occlusion of the body. A third question was how error rates varied with gesture direction. Because of human hand physiology, it seemed that gestures were easier to make in some directions than in others.

Independent Variables

The independent variables in this experiment were *TargetSize* (6, 9, and 12 pixels), *Exposure* (0.25, 0.5, 0.75, and 1), *Direction* (8 directions); *Color* (whether icons are monotone (light gray) or colored by gesture direction), and *BeakOcclusion* (whether the occluding object comes from the beak direction or base direction). We narrowed the *TargetSize* range because Experiment 1 showed little difference for targets more than 12-pixels wide. The experiment used a total of 3 (*TargetSize*) * 4 (*Exposure*) * 8 (*Direction*) * 2 (*Color*) * 2 (*BeakOcclusion*) = 384 trials per participant.

Color and *BeakOcclusion* were kept constant within blocks and counterbalanced. The other variables were presented randomly within a session. The apparatus, tasks, stimuli and procedures were the same as in Experiment 1.

Hypotheses

(H4) *Color* would improve performance time and error rate.

(H5) *BeakOcclusion* would increase task time and error rate since it would be harder to recognize the gesture indicated by the target icon.

(H6) Error rates would vary with *Direction*.

Participants

Eight right handed people (six male and two female) participated in this experiment. As in Experiment 1, all participants had some experience with a touch screen mobile device. Each was compensated with a $20 gift card.

EXPERIMENT 2 RESULTS

Selection Time

Within-subjects ANOVA showed a main effect for all variables: *TargetSize* ($F_{2,3069}$=56.91, p<0.001), *Exposure* ($F_{3,3068}$=37.42, p<0.001), *BeakOcclusion* ($F_{1,3070}$=88.06, p<0.001), and *Color* ($F_{1,3070}$=17.32, p<0.001). Significant interactions were found for *Exposure* X *BeakOcclusion* ($F_{7,3064}$=7.44, p<0.001) and *TargetSize* X *BeakOcclusion* X *Color* ($F_{11,3060}$=3.66, p<0.05).

Tukey's post-hoc pairwise comparison showed significant differences in *BeakOcclusion* in all *Exposure*s except fully exposed. Furthermore, in the case of no beak occlusion, there was no significant difference in performance among *Exposure*s greater than 0.25 (see Figure 10). These results indicate the importance of making the beak visible.

Surprisingly, one-colored icon selection was as fast as or faster than eight-colored icon selection. Figure 11 shows the mean performance time by *TargetSize, BeakOcclusion,* and *Color*. No significant differences for color were found, except for 6-pixel targets with no *BeakOcclusion*, in which case one-colored icons were faster.

Error Rate

For *TargetSize* and *Exposure*, error rates showed a pattern similar to performance—less *Exposure* or a smaller *TargetSize* was more error-prone. Somewhat surprisingly, no significant differences were found in *Direction*, although there was a trend with gestures up and to the left causing more errors (Figure 12). An ANOVA test on error rate aggregated across *Direction* for *TargetSize* X *Exposure* X *Color* X *BeakOcclusion* found main effects in *TargetSize* ($F_{3,380}$=27.93, p<0.001), *Exposure* ($F_{3,380}$=25.04, p<0.001), and *BeakOcclusion* ($F_{1,382}$=30.74, p<0.001). The significant interactions were *TargetSize* X *Exposure* ($F_{3,380}$=7.17,

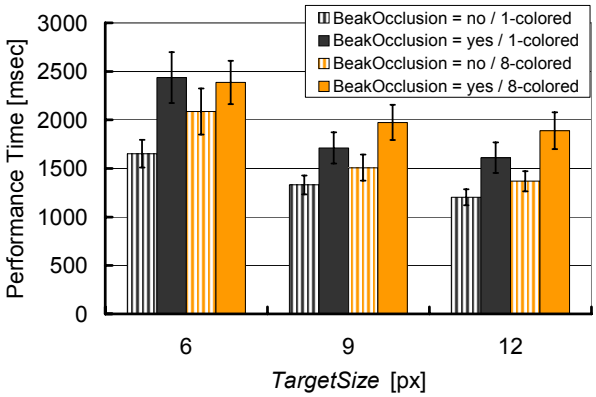

Figure 11. The mean performance time for *BeakOcclusion* X *TargetSize* X *Color* in Experiment 2, averaged over participants and target exposure. The graph of error rates looks similar, but has larger variance.

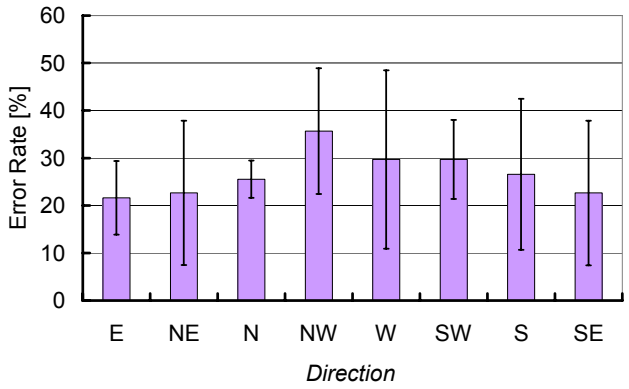

Figure 12. The error rate for *Direction,* averaged over all other independent variables in Experiment 2. Error rates are higher than Figure 8 because target sizes are smaller.

$p<0.001$), and *TargetSize* X *Color* X *BeakOcclusion* ($F_{3,380}=4.23$, $p<0.01$).

EXPERIMENT 2 DISCUSSION

The effects of color surprised us; our results did not support H4. We had expected color to help performance, not degrade it. Post-experimental interviews revealed that participants did find the colors distracting and that the colors were not discernable in small targets.

Our results support H5. Most participants said that they used beak shape rather than color to determine gesture direction; the results agreed with the participants' statements. This confirmed our belief that Escape should deliberately arrange icons to avoid beak occlusion.

Although no significant effect of *Direction* was found, some participants did dislike some directions (NW, W, and SW) because they involved stretching the thumb, whereas other participants disliked other directions (S and SE) because they involved contracting the thumb. This finding implies that Escape might offer a user-definable parameter to favor certain gesture directions over others.

ICON ARRANGEMENT

We now describe an algorithm to assign icons to target positions. The algorithm's primary task is to find an assignment that allows icons to be well-separated from other icons with the same gesture. Additionally, the system should minimize icon overlap, especially of the beak. This problem is similar to graph coloring [8], which is known to be NP-complete even for planar graphs. Thus, there is no known efficient optimal algorithm. Here we describe a heuristic algorithm that appears to work well in practice.

Beak occlusion occurs when targets are located near each other. To minimize the effects of closely spaced icons, Escape attaches the tip of the icon's beak to the target location. The icon body may then be put in any of eight possible locations around the target. This flexibility helps avoid many occlusions that would occur if the target location were instead attached to the icon center.

Our algorithm represents each target as a node in a graph. Each node is connected by a link to all other nodes in its *neighborhood,* defined as a 9.2 mm radius circle around the target. Each node also has eight subnodes representing the eight possible icons, and each subnode has a weight representing the likelihood that the corresponding choice of the icon will cause an occlusion or a violation of the spatial constraint. The algorithm calculates the initial weight of each subnode based only on occlusions. Subnodes close enough to other nodes are given higher weights because there is less freedom to place an icon there.

After the initial weight assignment, the algorithm first finds the node that has the most other nodes in its neighborhood, and then finds the subnode of that node with the least weight. Then the weights at the neighborhood nodes are updated by adding a large weight to their subnodes that represent the same kind of icon. The algorithm proceeds in this greedy manner, at each step choosing a least-weight subnode for a node with the largest number of the neighborhood nodes. The calculation stops when it has assigned icons for all items.

To test the algorithm's performance, we ran a simulation that varied the number of onscreen targets from 10 to 100. 1000 screens of icons were tested for each number of targets. The simulator chose target locations randomly, but avoided a 20 pixel margin around the edges of the screen. We considered a screen a success when the algorithm could assign all icons to targets without violating the spatial constraint. Figure 13 illustrates how our algorithm improves upon a random icon assignment.

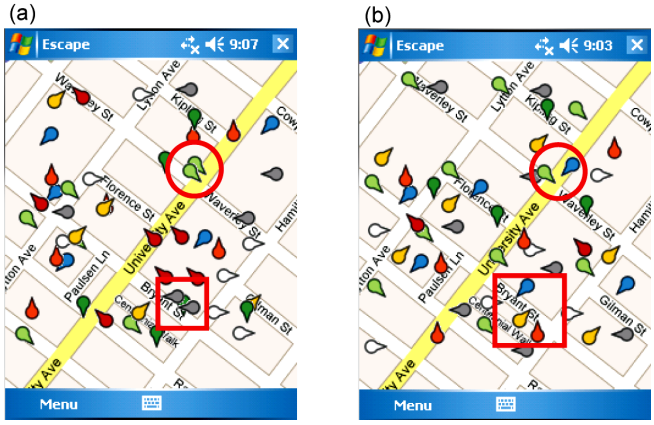

Figure 13. By carefully assigning icons, overlaps and unnecessary icon proximities can be avoided. (a) Random assignment; (b) Our overlap-avoidance algorithm. The circled region shows a case where the algorithm avoids placing identical icons together, and the squared region shows how the algorithm avoids icon overlap.

For a high rate of success, the algorithm can only handle five icons per neighborhood, which works out to a density of 2.3 icons per square centimeter. Note that this is for a high success rate over an average density over 1000 screens, some of which have concentrated regions with much higher local neighborhood densities. The algorithm calculates the arrangement of 100 items in around three seconds on a Windows Mobile emulator.

Note that icon assignments can be precomputed offline in some applications. Moreover, Escape can also be useful for manually-designed user interfaces, in which case the maximum density can be predictably achieved.

LIMITATIONS OF ESCAPE

The performance benefits of Escape do not come without drawbacks. Many applications use selections in background spaces to perform operations like map drags and generic pop-up menus. Because Escape expands the selection zone around a target, there is less open space in which to perform a target-free selection. In some cases, this can be overcome by using a more complex gesture (*e.g.*, by making a multi-segment gesture), but it is more work for the user.

Also, because Escape requires that icons indicate gesture, the maximum number of onscreen selectable targets is less than that of Offset Cursor and Shift, which can handle selection of individual pixel elements. This excludes applications like drawing programs, where pixel accuracy is critical.

Finally, gestures cannot go beyond a screen edge, so the set of icons allowed near the edge of the screen is more limited than the set allowed at the center. This reduces target density near the screen edge.

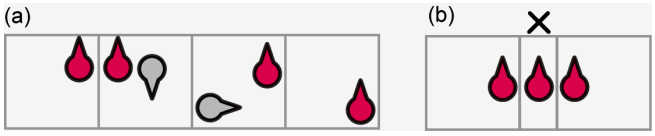

Figure 14. Two icons with similar gesture directions can be near each other if 9.2mm Parhi boxes can be drawn around each such that they contain no other similar icons. The four upward-pointing icons in (a) are well-separated; the three upward-pointing icons in (b) are not.

IMPROVEMENTS TO ESCAPE

Our user study also inspired additional variations that would be useful for a practical deployment. In addition to the design implications above, there are several other improvements that could be made to Escape.

Enhancement to Thumb Gestures

Icon appearance is not the only possible cue to suggest a gesture. In some cases, relative icon positions may be sufficient. For example, dialog boxes containing two adjacent buttons might use a rightward gesture to select the right button, and a leftward gesture to select the left.

In our experiments, many participants desired a mechanism to cancel an in-progress gesture. Escape could interpret returning to the gesture starting point as a cancellation operation.

Although the results from the second pilot study discouraged us from two-level gestures because of our icon design, there are other gesture mechanisms, such as multi-length gestures or zone or polygon gestures [18] that might be easier to use. While these gestures are easily performed and easily distinguished, it is not obvious what icon designs would suggest these gestures clearly in high-density situations.

Arrangement-Specific Selection Zones

Greater densities and more layout flexibility can be achieved if the selection region for an icon is not centered on it. Two immediately adjacent icons, indicating identical gestures, can still be easily distinguished if it is possible to draw a Parhi box around each as long as there are no additional icons with the same gesture inside those Parhi boxes (see Figure 14). This works as long as all nearby identical icons are visible, so the user knows on which side of an icon to begin a gesture.

A variant of this idea is to expand a target's initial selection zone beyond a Parhi box to its cell in the Voronoi diagram constructed from all targets. This approach has been shown to improve selection performance in traditional target selection [5]. However, it is important to limit the distance at which a target could be selected, both to avoid confusion when targets are far from the contact point, and to allow background regions to support non-target selecting commands.

Generalized Distance

The method used in this paper to match gestures and targets first limited the search space to icons within a Parhi Box, and then found the icon with the most closely matching gesture. An alternative is to frame the problem as finding the closest icon represented by an (x_i,y_i,θ_i) point in a three-dimensional space to the (x_g,y_g,θ_g) point given by the user's gesture. This approach would be more forgiving of positioning errors and might reduce overall error rate.

Combining Escape and Shift

Escape and Shift could be combined to make a target selection technique that would likely perform better than Escape for icons six pixels wide and smaller. Dense target clusters would bring up the Shift inset, after which the user could more easily see the icons in that space and perform the disambiguating gesture. The inset might not only magnify the area, but also better separate dense icon groups to make it easier to identify separate icons, and draw icons with finer resolution than is possible at the base resolution.

CONCLUSIONS

We have presented a thumb-based touch screen target-selection technique called Escape. In Escape, the user establishes an initial approximate position of interest, followed by a disambiguating gesture that is cued by the target to be selected. A controlled study showed that Escape is significantly faster than Shift while roughly matching its accuracy. Although direct touch selection will likely be faster than Escape for larger icons, poor accuracy rates make Escape a preferred solution for smaller icons.

ACKNOWLEDGEMENTS

We would like to thank Bo Begole for making helpful comments on this project, Ellen Isaacs and Diane Schiano for their help for the experimental design, and Alan Walendowski for helping us with the implementation of Shift. We also thank Daniel Vogel for providing the code for the dynamic low-pass filter and Khai N. Truong for giving us comments on this paper. We thank all the participants in our experiments for their help and cooperation.

REFERENCES

1. Accot, J. and Zhai, S. More than dotting the i's --- foundations for crossing-based interfaces. In *Proceedings of CHI '02*, ACM Press (2002), 73-80.
2. Albinsson, P. and Zhai, S. High precision touch screen interaction. In *Proceedings of CHI '03*, ACM Press (2003), 105-112.
3. Benko, H., Wilson, A. D., and Baudisch, P. Precise selection techniques for multi-touch screens. In *Proceedings of CHI '06*, ACM Press (2006), 1263-1272.
4. Fitts, P. M. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6), (1954), 381-391.
5. Grossman, T., and Balakrishnan, R. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of CHI '05*, ACM Press (2005), 281-290.
6. Guiard, Y., Blanch, R., and Beaudouin-Lafon, M. Object pointing: a complement to bitmap pointing in GUIs. In *Proceedings of GI '04*, ACM Press (2004), 9-16.
7. Hinckley, K., Baudisch, P., Ramos, G., and Guimbretiere, F. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In *Proceedings of CHI '05*, ACM Press (2005), 451-460.
8. Johnson, D. S. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9(3), (1974), 256-278.
9. Karlson, A. K., and Bederson, B. B. ThumbSpace: generalized one-handed input for touchscreen-based mobile devices, In *Proceedings of INTERACT '07*, Springer (2007), 324-338.
10. Karlson, A. K., Bederson, B. B., Contreras-Vidal, J. Understanding one handed use of mobile devices, *Handbook of Research on User Interface Design and Evaluation for Mobile Technology*, Idea Group, 2007.
11. Karlson, A. K., Bederson, B. B., and SanGiovanni, J. AppLens and LaunchTile: two designs for one-handed thumb use on small devices. In *Proceedings of CHI '05*, ACM Press (2005), 201-210.
12. Kurtenbach, G. and Buxton, W. The limits of expert performance using hierarchical marking menus. In *INTERCHI '93*, ACM Press (1993), 482-487.
13. Parhi, P., Karlson, A. K., and Bederson, B. B. Target size study for one-handed thumb use on small touchscreen devices. In *Proceedings of MobileHCI '06*, ACM Press (2006), 203-210.
14. Perlin, K. Quikwriting: continuous stylus-based text entry. In *Proceedings of the UIST '98*, ACM Press (1998), 215-216.
15. Potter, R. L., Weldon, L. J., and Shneiderman, B.. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In *Proceedings of CHI '88*, ACM Press (1988), 27-32.
16. Sears, A. and B. Shneiderman, High precision touchscreens: design strategies and comparison with a mouse. *International Journal of Man-Machine Studies*, 43(4), (1991), 593-613.
17. Vogel, D. and Baudisch, P. Shift: a technique for operating pen-based interfaces using touch. In *Proceedings of CHI '07*, ACM Press (2007), 657-666.
18. Zhao, S., Agrawala, M., and Hinckley, K. Zone and polygon menus: using relative position to increase the breadth of multi-stroke marking menus. In *Proceedings of CHI'06*, ACM Press (2006), 1077-1086.