Optimal Parameters for Efficient Crossing-Based Dialog Boxes

Morgan Dixon, François Guimbretière, Nicholas Chen
Department of Computer Science
Human-Computer Interaction Lab
University of Maryland
{mdixon3, francois, nchen}@cs.umd.edu

ABSTRACT

We present an empirical analysis of crossing-based dialog boxes. First, we study the spatial constraints required for efficient crossing-based interactions in the case of a simple multi-parameter dialog box. Through a series of 3 tasks, we establish the minimal value of the landing margin, the takeoff margin, and the column width. We also offer an estimation of the role of stroke shape on user performance. After studying the reasons for errors during our experiment, we propose a relaxed crossing semantic that combines aspects of pointing and crossing-based interfaces.

To test our design, we compare a naïve dialog box implementation with our new implementation, as well as a standard point-and-click dialog box. Our results reveal that there is not a significant difference between the naïve crossing implementation and the standard point-and-click interface and that the new crossing semantic is faster than both the naïve crossing implementation and the point-and-click interface, despite a higher error rate.

Together these two experiments establish that crossing-based dialog boxes can be as spatially efficient and faster than their point-and-click counterpart. Our new semantic provides the first step towards a smooth transition from point-and-click interfaces to crossing-based interfaces.

Author Keywords
Crossing-Based Interfaces, Graphical User Interfaces, Dialog Box, Interaction Design.

ACM Classification Keywords
H5.2. User Interfaces: *Input devices and strategies (e.g., mouse, touchscreen)*; *Graphical user interfaces (GUI).*

INTRODUCTION

As the number of Tablet PCs in use steadily increases, it is important to better understand the performance characteristics of interfaces specifically designed for pen interactions. One class of interfaces comprises goal crossing-based interfaces in which all interactions are performed by crossing targets on the screen. Such interaction styles appear to have a great potential: Accot and Zhai [2] demonstrated that crossing tasks can be described by Fitts' law, with performance comparable to standard point-and-click tasks, and suggested that in certain scenarios, such as when targets can be cascaded, crossing interfaces might be faster than point-and-click interfaces. Taking note of this advantage, Apitz and Guimbretière [6] showed that one can design a complete application (a simple drawing program called CrossY) which relies only on goal crossing for all interactions. CrossY not only demonstrated that goal crossing-based interfaces could be as expressive as standard point-and-click interfaces, but it also illustrated several of the potential advantages of crossing-based designs. These include the ability to use a rich gesture set on top of interface elements such as the scrollbar (an approach reminiscent of the Gedrics system [12]), and the ability to compose several commands in one gesture.

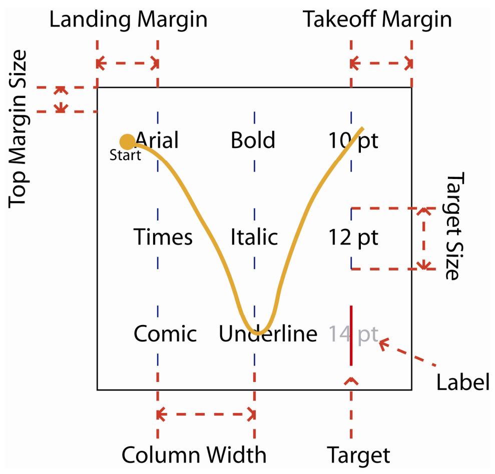


Figure 1: An example crossing dialog box with several spatial parameters labeled. The user selects targets from left to right, in this case crossing options Arial, Underline and 10 pt.

The latter feature is unique to the crossing style of interactions and could offer a significant speed advantage for pen-based interactions. However, Apitz and Guimbretière warned that designers might face a trade-off between speed of execution of composite commands [6], and the screen real estate required for these interactions. This problem arises from the fact that when users quickly select several options with a single gesture, they tend to be sloppier and require more space to move without triggering unwanted commands. Apitz and Guimbretière do not report any empirical evidence of the amplitude of this problem but suggest that this problem could be alleviated by using gesture based recognition mechanisms such the one used by the SHARK keyboard system [17].

In this paper we offer the first empirical evaluation of the space versus speed trade-off that might occur while designing a typical crossing-based dialog box. A simple example of such a dialog box used to select several text attributes is shown in Figure 1. Like in CrossY, users trigger an action by crossing the bar that is behind each label (as shown in Figure 1). To evaluate this setting, we first conducted an experiment designed to explore the minimum spatial requirements for efficient option selection in the dialog box, such as the ones shown in Figure 1. In particular we studied the impact of key parameters, including the required margins around the goals, the spacing of columns of targets as well as the overall cost of making complex selections in a three column box. Our results show that the spatial requirements for crossing-based interfaces are very similar to those of point-and-click interfaces. Based on our findings, we present guidelines for the design of space efficient crossing-based dialog boxes.

The observations gathered during our first experiment offered some insight on how to improve the performance of complex selections involving a middle goal either far below of far above the horizontal line formed by the first and last options selected. In such a situation, it is often difficult for users perform the change in direction when selecting the middle option. To address this problem, we propose to blend the point-and-click and the crossing semantics to offer a style of interaction which is more forgiving of common errors, such as crossing just above a target. To evaluate our solution, we conducted a second experiment designed to compare the speed performance of crossing-based and point-and-click interfaces. In this experiment we compared our new semantic to a naïve crossing-based interface and a standard point-and-click interface. Our results showed that in our setting, there was no significant speed difference between the naïve crossing-based interface and the point-and-click interface, whereas the new implementation offered a significant speed advantage despite a higher error rate. Our solution, which could be implemented as an extension of current point-and-click interfaces for pen-based computing, could significantly increase the efficiency of command selection in pen-based computing.

ANATOMY OF CROSSING-BASED DIALOG BOXES

The typical layout of a goal crossing-based dialog box is presented in Figure 1. In this example, users can select 3 different values for 3 text attributes (from left to right, Font, Style, and Size). Since possible values for each attribute are exclusive, the dialog box presents the possible values of a given attribute in a column layout. Although users could select the value for each attribute one at a time, the crossing paradigm [4] makes it possible to select several attributes using a single stroke as shown in Figure 1. This is the case we are considering in detail in this paper because results from Accot and Zhai [4] show that when the Fitt's law ID between targets in a dialog box are below 5 this is the fastest mode of operation. It is also unique to crossing-based interactions [6].

Now, consider the case where one would like to set the following font attributes in one stroke: Arial, Underline, and 10pt. First, we note that the work by Accot and Zhai on the performance of goal crossing [4] does not directly apply, since several effects, such as the presence of a limited landing margin and the need to travel at an angle, were not considered in their study. Therefore, we proceed, similarly to Pastel et al. [16], by decomposing the different steps required to perform this selection; highlighting how previous results might help predict the performance of each action. The first step is for the user to land near the start position to the left of the 'Arial' target. This part of the interaction can be modeled as a bivariate Fitts' law task [5] (or alternatively [13] if the landing region is not rectangular) in which both the landing margin (see Figure 1) and the height of the first target (including the distance between target and maybe the top margin size) will influence user performance. Then as users cross the target, the speed will be limited by the target width as studied by Accot [3] .

During the next step of the selection process, users have to travel toward the next target, 'Underline'. During this segment of the selection, one can expect two distinctive types of behavior: if the column width is small, the different options will appear as a tunnel users must travel through without crossing the boundary. In that case, user performance will be modeled by the Steering law for tunnels [3]. As the space between columns increases, Accot suggests in his thesis [3] that the Tunnel law will not apply anymore. Instead the tunnel becomes so wide that a normal "ballistic" movement has little chance to cross the borders, and users' interactions are modeled by Fitts' law with a possible influence of the direction of movement [9, 15]. The exact transition between the two will depend on the experimental setting.

Reaching the vicinity of the 'Underline' target, the user must now cross the target while performing a sharp turn. To our knowledge there have been no direct investigations of such interactions, but several related studies might help predict users' behavior. Pastel et al. [16] studied the influence of angle while navigating through corners, reporting that "users will round off corners while gesturing or negotiating menu hierarchies." Also, Cao and Zhai [7, 8] presented a new quantitative model for single stroke pen gestures, the Curves, Line Segments, and Corners (CLC) model, which demonstrated that in a free setting, it typically takes users within 40 milliseconds to draw a corner.

In the final stage of the selection, the user must aim at and cross the third target "10 pt" before lifting the pen. As before, user performance will be influenced by the distance between the columns and the size of the target. It will also be influenced by the size of the takeoff area, probably following the bivariate Fitts' law task [5] in which both the takeoff margin (see Figure 1) and the height of the target (including the distance between target and maybe the top margin size) will influence user performance.

In summary, to better understand how the layout of a crossing-based dialog box influences user performance, one has to consider:

- How the dialog box margins might influence landing and takeoff performance;
- How the distance between columns might influence user performance;
- How the need to perform sharp turns while selecting several commands in a row might influence user performance;

We proceed with the description of our first experiment designed to evaluate the impact of these parameters.

EXPERIMENT I

To keep the complexity of our experiment in check, we decided to divide our experiment into three tasks, each investigating one of the parameters described above. While this might hide potential interactions between these variables we felt that a full factorial design was unpractical and might not be necessary at this stage of our investigation.

In our first task, we presented users with a simple dialog box with one column of options to study the influence of the takeoff and landing margins. For our second task, we used a two column dialog box to study the influence of the distance between columns on user performance. Finally, for our third task, we focused on the effect of command composition by asking users to select 3 parameters.

To further simplify our design, we did not include the target height and the size of the top and bottom margins as variables in our experiment. With respect to the target size, we turned instead to existing systems, fixing all target heights to 18 pixels, with 9 pixels of vertical space between targets. This corresponds to the standard checkbox sizes used in Mac OS X [1]. With respect to the top and bottom margins, we investigated their influence during pilot studies and found that there were no notable effects on user behavior as long as they were wider than 3 pixels. For an added margin of safety, we fixed the top and bottom margins to 9 pixels, such that every target was surrounded equally by 9 pixels of space.

Although each the of three tasks were presented in successive blocks in one experimental session, in the following we will present each task and the corresponding results as separate sections for the sake of clarity.

Protocol and Participants

The three tasks were presented in blocks. Before completing each task, users completed a training session until they were comfortable with the task (typically around 45 trials). Subjects were asked to cross targets as fast as possible, but with precision. They were asked to keep an error rate below 6% and were given a visual warning when their error rate exceeded 4% (the error counter changed from yellow to red). Participants were allowed to rest as soon as they finished a trial.

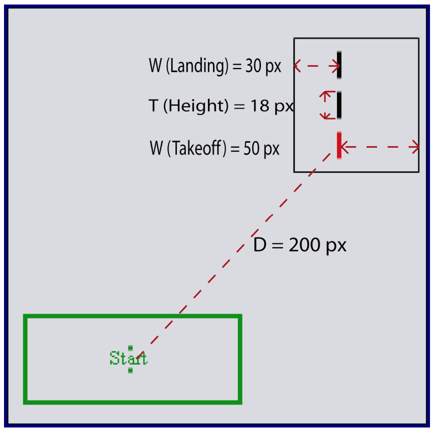


Figure 2: An example trial of the landing and takeoff test. The landing margin is 30 pixels and the takeoff margin is 50 pixels.

We recruited 12 participants (8 male, 4 female; age range 18-51 years). All subjects were right-handed. Subjects received $10 for their participation.

Apparatus

All subjects completed the tasks on a Toshiba Protégé Tablet PC, with 2 GB RAM and a 1.7 GHz CPU frequency. The diagonal of the screen was 307mm and the resolution was set to the tablet's native 1024 x 768 pixels (or a 0.24 mm pixel pitch). The tablet screen was folded so that the computer appeared as a slate and placed in portrait orientation. The test application was written in C# and all actions performed by users were logged.

Takeoff and Landing Margins

Our goal for this task was to understand the influence of takeoff and landing margins on user performance. To measure these effects experimentally, we created a simple task in which users must first cross a fixed starting target, then cross another target to the right of the starting point as shown in Figure 2. We fixed the distance between the starting point and the goal at 200 pixels, but to introduce variety, the target goal could be one of three targets in the dialog box, and the angle between the two targets and the horizontal varied from 0 to 80 degrees. During the experiment, we systematically varied the width of both the takeoff and landing margins and observed their effect on the task time, defined by the time from when users crossed the fixed target to when they lifted the pen after crossing the second target. To investigate possible interactions between the takeoff and landing areas, we decided to run a factorial design, crossing the two parameters. We picked the following values for each margin: 5, 10, 15, 20, 25, 30, 40, 50, 60, and 70 pixels, with 5 pixels being the smallest practical target (corresponding to an ID of 5 in our setting) and 70 being well into the performance plateau according

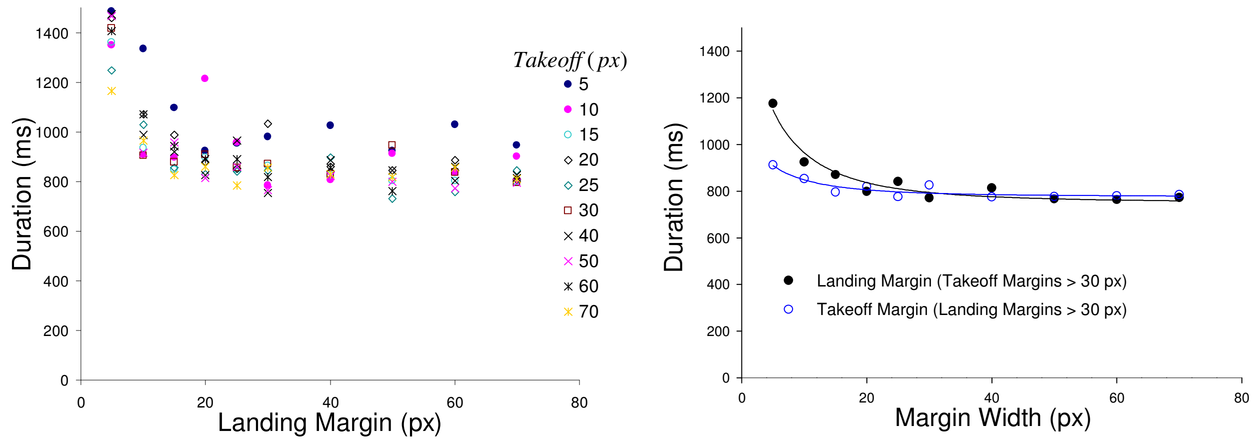


Figure 3: Left: The margin sizes (pixels) versus average task duration (ms). Each colored sequence represents the corresponding takeoff margin width. Right: The landing and takeoff widths excluding the opposing margin widths greater than 30 pixels.

to our pilot studies. The composition of the goal target varied randomly between trials to offer a more realistic setting. Each possible combination of margins was tested 3 times for a total of 300 trials.

Because the landing region can be described as a rectangular target [5], we expected that the users' total movement time (*MT*) would follow a the bivariate version of Fitts' law described by the following formula:

$$MT = a + b\log_2\left(\sqrt{\left(\frac{D}{W}\right)^2 + \eta\left(\frac{D}{T}\right)^2} + 1\right) \quad (1)$$

in which D is the distance between the fixed target and the goal target, W is the margin width, T is the target height (shown in Figure 2) and η is an experimentally determined constant. One would expect user performance to plateau after the margin size grows larger than the target height as the target height will become the limiting factor. We were expecting a similar effect to take place for the takeoff margin.

Results

Out of the 1197 trials, there were a total of 36 errors, yielding an error rate of approximately 3%. 27 (75%) of the errors were caused by users crossing the boundary of the box during a selection. The majority (21 occurrences) of these occurred within the 5-pixel takeoff margin setting. This makes sense as it becomes difficult to cross the target without exiting the box for such small takeoff margins. 9 (25%) of the errors were occurrences of users selecting targets in the wrong direction. This seems to be primarily because users would cross the goal target and then slightly backtrack because of the difficulty to aim in such a small region. We now proceed to look at the task durations with respect to the margin widths.

We examined the average duration for each combination of landing and takeoff margins, where the duration was defined from when the user presses down with the pen in the dialog box to when the user lifts the pen after successfully crossing the goal target. We looked at the duration of each user's median trial for every setting (excluding trials with errors) and averaged all participants to obtain an estimate for each task time. We removed the 5 pixel landing margin, 70 pixel takeoff margin setting from one user because his or her trial times were 10 times slower than any other user. This did not change the nature of our findings.

Our results are shown in Figure 3, left, in which we plot, for each takeoff margin, the total task time as a function of the landing margin size. As predicted, Figure 3, left, shows little change in performance beyond 30 pixels for both landing and takeoff margins, but the data is quite noisy. As a result, we decided to study the influence of landing and takeoff margin separately. To do so, for the landing margin (respectively takeoff margin), we ignored data points with a takeoff margin (respectively landing margin) smaller than 30 pixels and aggregated all data captured with a takeoff margin (respectively landing margin) greater than 30 pixels. The results are shown in Figure 3, right, super-imposed over the best fit of the bivariate pointing model (1) with D = 200 and H = 18 for clarity. For the landing margin, we again observed a clear plateau starting around 30 pixels (the duration does not change more than 50ms) which corresponds to the total target height if one includes the space between targets. For the takeoff margin, the plateau seems to come sooner just short of 20 pixels, which corresponds to the height of the target. Both results are in accordance with our predictions. We conclude that the minimum margin should be at least 50% larger than the target height (30 pixels in our case).

Target Column Width

In the second task, our goal was to determine the effect of the column width. Starting from the start target, participants were asked to cross the start target, lift the pen, move to the dialog box, and then in a single stroke select two highlighted (red) targets from left to right (Figure 4). While the start button is not strictly necessary for our

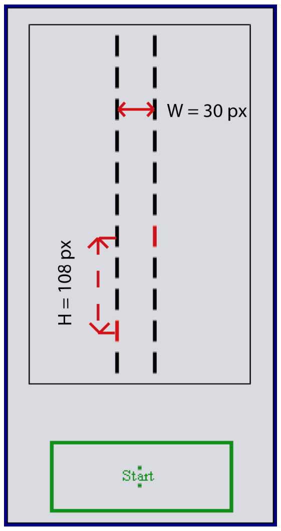


Figure 4: A sample trial of the column distance task. In this example, the distance between columns is 30 pixels.

measurement, our pilot studies showed that it has two beneficial effects: first, it limits potential hand occlusions by forcing users to always move back to a position where their hand is out of the way. Second, it prevents people from entering a "stride" mode in which they mechanically repeat the same movement, often missing the fact that a new combination of targets was presented. To observe the effect of column width over a range of IDs we varied the vertical distance between targets from 0, 54, 81, 108, 189, and 270 pixels (or 0, 2, 3, 4, 7, 10 vertical steps respectively) either upward or downward, a selection representative of common dialog boxes (and also seems to be the limit of users' patience for very narrow columns).

In this task, we methodically varied the column width and observed its effect on participant movement time from the time the user crossed the first target to the time the user crossed the second target. For the width variable, we tested 10, 15, 20, 25, 30, 40, 50, 60, and 80 pixels, with 10 pixels being the smallest practical width and 80 pixels entering the range where the task becomes a Fitts' law pointing task, according to our pilot studies. These choices meant that the ID of the task varied from 1 to about 27 in the case of a large vertical movement through a narrow column. Each setting was tested 6 times (3 in the upward direction and 3 in the downward direction) for a total of 270 trials.

As mentioned, users' performance can be described by either of two laws in this task. When the column width is very small, the surrounding options create a tunnel that the users must navigate through without touching the boundary, as it would trigger an unwanted option. In that setting, the movement time can be described by the Steering law [2], where *H* is the vertical distance between targets and *W* is the column width:

$$MT_{Accot's} = a + b\frac{H}{W} \quad (2)$$

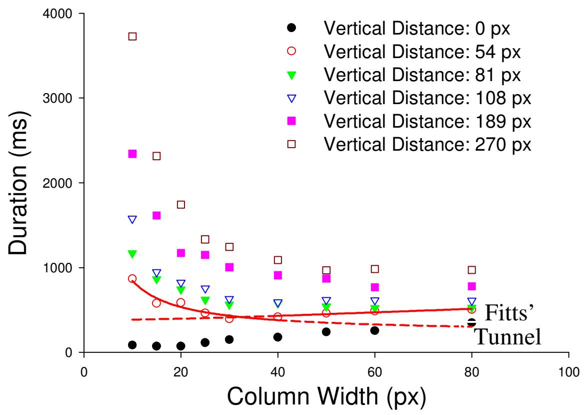


Figure 5: Column width vs. average duration. Each sequence represents a different vertical distance between targets.

But as the tunnel becomes larger (*H* increases), this law will no longer apply [3] since the tunnel will become so wide that it will not impede direct ballistic movement. In this case, Fitts' law [11] can be used to model the movement:

$$MT_{Fitts'} = a + b\log_2\left(\frac{\sqrt{H^2 + W^2}}{18} + 1\right) \quad (3)$$

Results

Out of the 1179 trials there were 70 errors (5.9%). 32 errors occurred from users lifting the pen before crossing both targets, 26 errors occurred from users crossing a wrong target, 10 errors occurred from users crossing targets from the wrong direction (right to left) and 2 errors occurred from users crossing outside of the dialog box.

For this task we used each participant's median performance, excluded error trials and averaged across all participants to obtain an estimation of the time it takes to complete the task for each cell. Figure 5 plots, for each vertical distance, the average movement time depending on the column width. As expected, the trace for 0 pixels of vertical distance follows a very different pattern from the other conditions because Accot's law does not play a role. For all others settings, one can see that for small inter-column widths, user performance follows the Steering Law, with performance improving as the inverse of the column width, as would be expected from (2). Yet, as the column width exceeds 40 pixels, user performance begins to follow Fitts' law. For small vertical distances (e.g. 54px, 81px), increasing *W* has a large effect on the total distance traveled, and one observes a point of inflection (user performance begins to decrease) as user performance transitions to Fitts' law (3). For clarity, we also plotted Accot's law and Fitts' law against the red 54 pixel series in Figure 5, making the curve dotted when the corresponding law does not apply. For larger vertical distances (e.g. 189px, 270px), *W* has a smaller effect on the total distance traveled so the transition to Fitts' law behavior appears more as a slowly rising plateau.

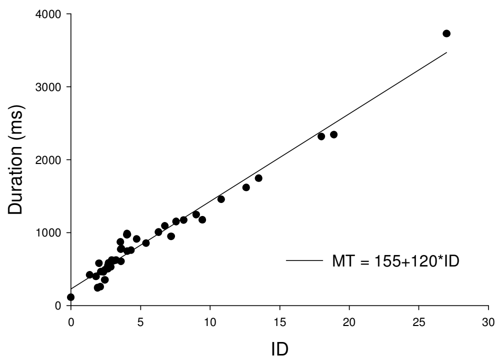


Figure 6: ID vs. task duration. For column widths under 40 pixels Tunnel law IDs were used, for those above 40 pixels, Fitts' law IDs were used.

To explicitly find the transition points between the Tunnel law and Fitts' law, we determined the experimental weights in the formulas (2) and (3) by performing linear regressions on the extreme scenarios (margin widths of 10 and 80 pixels). We then used these formulas to solve for each travel distance the point where:

$$MT_{Fitts'} = MT_{Accot's}$$

We found that the transition points ranged from 35 to 40 pixels. We plotted Accot's law IDs for settings less than or equal to 40 pixels and Fitts' law IDs for those greater than 40 pixels against the average task duration in Figure 6 and, as expected, we see a very strong fit ($r^2 = 0.96$). Based on these results, designers should expect that with standard target sizes and column widths passed 40 pixels, Fitts' law should be used to estimate user performance.

Composition

For the third task, our main goal was to observe the influence of the sharp angle users must make in order to select 3 targets in one stroke. To investigate this, users were required to select 3 highlighted targets in a 3 column dialog box as shown in Figure 7. Participants were asked to first cross the start target and then the three highlighted targets from left to right. We decided to include six rows of targets in this task since for larger IDs it is faster to lift the pen to perform selections [4], which seems appropriate since our study focuses on command selections in one stroke. Like in the previous task, the start target was not strictly necessary, but reduced the effect of occlusion and errors caused by users entering a "stride" in a given setting. In this task, the distance between columns was 80 pixels and the landing and takeoff margins were both 70 pixels, all of which were conservative values according to our pilot studies. Because the margin sizes and column widths are constant within this task, we were able to include textual labels on each target to provide a more realistic setting. During pilot studies, we did not notice any significant effect from the presence of labels. To avoid a combinational explosion, we did not repeat identical compositions. For example, in Figure 7, selecting "otter", "deer" and "cow" is considered identical to selecting "koala", "sloth", and "zebra" since these two

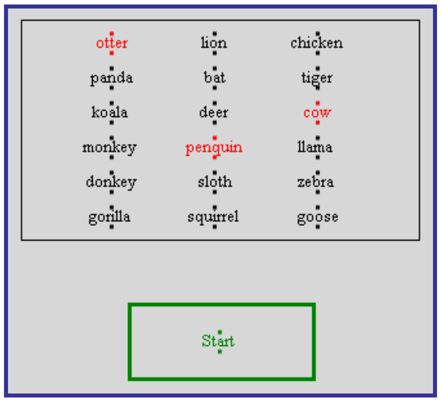


Figure 7: An example trial of the composition test. In this example, the user must cross the start target left to right.

interactions have congruent angles of inflection and general directions of travel. In some cases the performance of identical compositions may be affected by the bounding dialog box, such as in Figure 7 the border of the dialog box may affect the selection of "gorilla", "sloth", "zebra" differently than "panda", "lion", "chicken." In these situations we tested both compositions. We tested each composition 3 times to increase reliability, and there were 91 unique composition patterns, which yielded 273 total trials for this task. Before this task, we also informed users of the main techniques for crossing; we explained to users that one common technique is to cross orthogonally to the targets, or to cross by drawing a straight line between targets (shown in Figure 9).

With the complex selections in this task, there are two main difficulties for users. First, the distance between targets will be greater. This will increase the overall ID for the task. Second users must make a sharper turn while selecting the second target. Cao and Zhai [8] showed that in a free gesture setting, the time to draw a corner was typically less than 40 milliseconds (and practically negligible here). However, our setting is different in that goal constraints are imposed and users may decide to not strictly cross at a rigid angle. Instead, we expect that for steeper angles of inflection, the task will become more difficult. It is also the case that users may not need to make an inflection point, but the general direction of travel is at a nonzero degree angle from the horizontal (e.g. in Figure 9, if the user selects "Comic", "Italic", and "10 pt"), which may cause a decrease in apparent target size. Since we are interested in how the stroke complexity influences users' performance, we decided to consider the sum of the angles from the horizontal (e.g. in Figure 9 the angle created by moving from "Arial" to "Underline" plus the angle from "Underline" to "10 pt") as a proxy for complexity. To remove the influence of the distance between targets, we defined the time corrected for distance (TCD) of a task as:

$$TCD = Duration - ID * \left(\frac{Duration_{REF}}{ID_{REF}} \right) \quad (4)$$

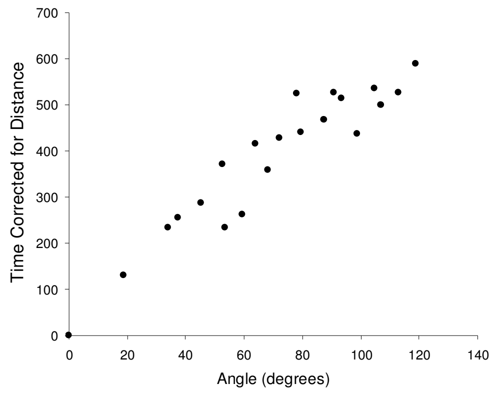


Figure 8: The sum of the angles from the horizontal that describe a composite command stroke vs. difficulty.

where $Duration_{REF}$ and ID_{REF} refer to the average duration and Fitts' ID for the 0 degree angle setting.

Results

Out of 1,139 trials in this task, there were 94 errors (8.25%). The most common error (occurring 62 times) was when users slightly missed a target, but continued with the interaction and crossed the successive target. The second most common error (occurring 22 times) was that users lifted the pen just before passing through the final target. Users also selected the wrong target 7 times and exited the box while crossing 3 times.

In this task we excluded error trials and again took the median trials from each user. We then looked at average task durations for each setting. We looked at the influence of the total stroke angle versus the task time corrected for distance (using $ID_{REF} = 4.89$ and $Duration_{REF} = 716.8$ ms in equation (4)), which is plotted in Figure 8. It is evident that the TCD of the task generally increases as the stroke angle increases. The angle influencing the TCD in this way makes sense because for steeper angles, the user must choose between either drawing long, s-shaped strokes, or by drawing straight strokes and greatly limiting the apparent target size, which is not present in Cao and Zhai's scenario.

Based on these results, designers should expect an approximately linear decrease in performance as strokes become more angular.

APPLYING THE RESULTS

From the results of this experiment, we can now determine the parameters that describe the optimal crossing-based dialog box. We give the minimum dimensions that should not hinder user performance in Figure 9. However, for design purposes, larger dimensions might be necessary. For the takeoff and landing margins, we suggest widths at least 50% larger than the target height (we suggest 30 pixels in our setting). Our second task revealed that column widths at least 9.6mm (40px) provide sufficient widths for dialog boxes sized similar to ours, however, note that this value is quite small compared to typical label sizes, which implies that most application designers should expect user behavior

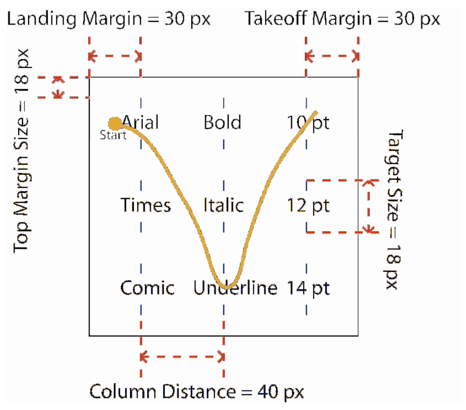


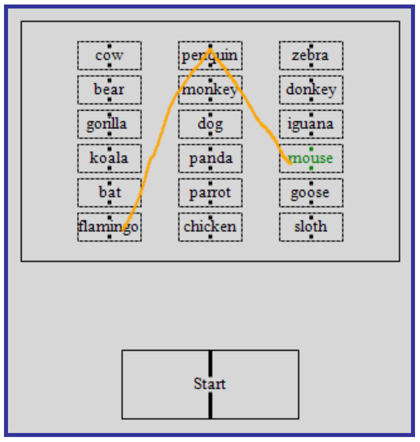
Figure 9: Design specifications for an efficient dialog box based on a target height of 18 pixels.

to be modeled by Fitts' law. We also recommend that the combinations which will be selected most often be as close as possible to a horizontal line to avoid the linear decrease in performance as strokes become more angular. Although we fixed our top and bottom margins to half of the target height in the third task, we suggest heights a bit larger, around the target height, as half was an extreme limit and design might demand a slightly bigger value to give users more space for the more difficult tasks against the border.

While Experiment I illustrated how to adjust the spatial parameters of the dialog box, we also noticed several aspects of the dialog box implementation that can be improved. One of the first issues was the difference between the possible stroke styles. During pilot studies, we noticed that a straight stroke style (as seen in Figure 10, left) can typically lead to faster task completion time. We also noticed that users typically crossed orthogonally to the targets during the third task, most likely because it provided the largest perceptual target size. Next, we noticed that it was often difficult to aim at the second target and negotiate the sharp angle simultaneously, which often caused users to slightly miss the second target. Last, we noticed that in many cases, users began crossing to the right of the first target or lifted the pen before crossing the last target (as shown in Figure 10, left, the user does not completely cross the "flamingo" and "mouse" targets).

A New Crossing Detection Algorithm

To address this problem, just as Lank et al. [14] allowed for sloppier selections when circling targets, we decided to relax our crossing semantic. First we assigned an invisible interaction box around each target as shown in Figure 10, left. In addition to detecting standard cross events, each interaction box detects three fundamental interactions: pressing down with the pen, lifting the pen, and creating a sharp angle. Detecting pen down events allows users to combine landing interactions and selecting their first target into one motion by allowing users to start on the target (or maybe a little bit to the right of it). Similarly, detecting pen up events allows for users to release the pen in the general vicinity of a desired target (or maybe a little bit left of it), giving a larger tolerance of error when selecting the last



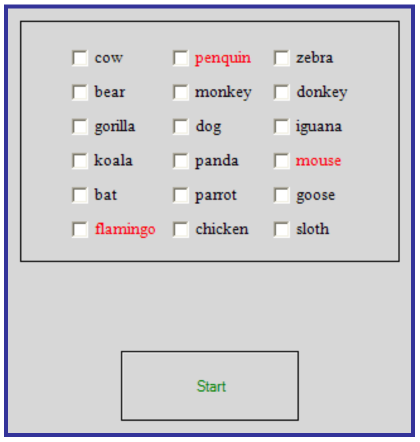


Figure 10: Left: A crossing scenario that our new backend would accept. Dotted rectangles show the regions (normally invisible) that detect pen-up, pen-down, and inflection point events. Right: An example trial of the point-and-click implementation.

target of a stroke. If the user wishes to select a single target, detecting pen down and pen up events will conflict. To solve this, our implementation ignores a pen up event that occurs on the same target directly after a pen down event. Finally, detecting a sharp angle makes it easy for users to make a selection while negotiating a turn through the middle target since the use of a box increases the apparent width of the target. For example, in Figure 10, left, the user only needs to draw an angle in the "penguin" region rather than specifically crossing the vertical target.

To detect sharp angles, we perform the Douglas-Peucker line simplification algorithm [10] on strokes within the interaction box, and issue a crossing event when an inflection point has been detected. Note that in the case where the three targets are aligned, there will be no inflection and if the user slightly misses the middle target, an error will occur. To reduce the likelihood of missing the middle target in this scenario, our algorithm adds four invisible pixels to each target (two above the target and two below). It should also be noted that our implementation is local to each target so it can be implemented using standard event-loop dispatching techniques.

Another important consideration is to ensure that our modifications do not hinder single target selections. For example, if the invisible boxes are too close horizontally, users may cross a single target, such as "dog" in Figure 10, left, and release the pen within the box surrounding "iguana," which would trigger an unwanted selection. To ensure that this does not happen, we first looked at the typical stroke widths from the first task of Experiment 1. For large landing and takeoff margins (60 and 70 pixels), the average stroke width was approximately 35 pixels (roughly 17 or 18 pixels before and after the goal). This suggests that an 18 pixel box is all that is necessary. But from the first task in Experiment 1 we also noted that for the landing margin, user performance plateaus at about 30 pixels. Balancing these two data points, we designed each box as 22 by 50 pixels. This configuration provides the maximal height (for targets of 18 pixels and 9 pixel spaces) and 30 pixels between regions assuming an 80 pixel column width.

Evaluating our Implementation

To verify the effectiveness of our new algorithm for multiple selections, we conducted a second user study where we evaluated user performance for each of three implementations: a traditional goal crossing implementation, a goal crossing implementation using our new algorithm and a standard point-and-click implementation (see Figure 10, right). Each dialog box consisted of 3 rows and 6 columns and we aimed to give each implementation identical spatial parameters. All implementations had 18 pixel visible target heights, 9 pixels between rows, and 80 pixel column widths. For the point-and-click implementation, we used check boxes for targets, with textual labels to the right of each target. To ensure that we are not providing an unfair advantage to the relaxed setting, we also added four invisible, active pixels (two above the target and two below the target) to each target in the point-and-click setting, which is common behavior in Windows. Also, according to Windows default behavior, the user may select the checkbox or to the right of the checkbox (summing to a width of 80 pixels for each target) to make a selection. The task and settings were identical to that of task 3 in our first experiment. For each implementation, participants performed the same 273 trials as in task 3 of Experiment I blocked together. We fully balanced the presentation of each technique to limit the influence of possible skill transfer.

12 participants (1 male, 11 female; age range 18 – 39 years; all right handed) were recruited for this study and they received a $10 compensation for their time. We used the same Tablet PC and the same apparatus settings as Experiment I. There were two experimenters who each ran a fully balanced sample of users.

Results

In our experiment, we timed each trial as beginning when the user presses the pen down inside the dialog box and ending when the user successfully finishes selecting all three targets and lifts the pen from the dialog box. We

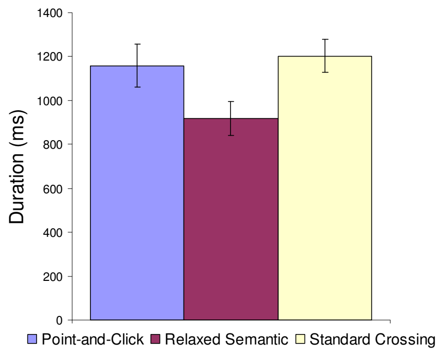


Figure 12: Task Type vs. Average Duration with 95% confidence intervals.

chose to ignore the time it takes for users to initially travel from the start target to the dialog box as it mainly represents thinking time. We did, however, include all of the time it took users to correct errors in our data. Since our work is focusing on one stroke selections, we required users to travel back to the start target and re-perform the entire trial after committing an error in order to force users to perform a fully successful selection at the end of each trial as a baseline reference. In our analysis, we used the Greenhouse-Geiser correction when sphericity could not be assumed and we used Bonferonni correction for post-hoc analysis. Also when performing our analysis of error rates, we noticed one user's error rates were more than triple the average for all three tasks due to frequently selecting the incorrect target. To limit the possible bias caused by this behavior, we ran an additional user as a replacement (using the same technique ordering and experimenter) and report these results here. This did not change the nature of our findings for either error rates or performance.

There was a significant difference in error rate among the three conditions ($F_{2,22} = 7.969$, $p = .002$, $\eta_p^2 = .420$, the error rates were 6.72% for the standard crossing, 8.27% for our new implementation and 4.37% for the point-and-click interface). Specifically the new implementation had significantly more errors than the point-and-click implementation ($p = .005$). The standard crossing implementation did not have a significant difference in error rates from the point-and-click ($p = .183$) and the new implementation ($p = .286$). It should be noted that lifting the pen before completely selecting all three of the targets is considered an error in the crossing conditions. In reality, we expect that users may prematurely lift the pen and then perform any additional strokes necessary; however, we did not allow this to force single stroke selections. Approximately 20% of the errors in the standard crossing condition and 23% of the errors in the relaxed condition were lifting errors. Since we are including time for error corrections, we believe that this effect has a limited impact on the validity of our results.

With respect to user performance, we removed outlier trials, which were trials exceeding three standard deviations from the mean duration for each setting. Figure 12 illustrates the average durations for each of the tasks. A repeated measure ANOVA on the task time revealed a significant difference between conditions ($F_{1.22,13.45} = 14.9$, $p = .001$, $\eta_p^2 = .575$). While the standard crossing implementation was not significantly faster than the point-and-click implementation ($p > .999$), the advanced implementation was significantly faster than both the standard crossing interface ($p < .001$) and the point-and-click interface ($p = .010$). These results validate our belief that the relaxed semantic offers better performance.

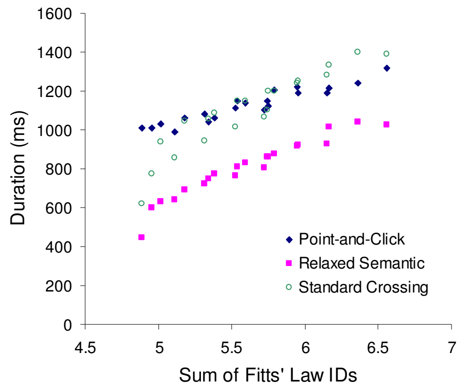


Figure 11: The sum of Fitts' law IDs from moving between targets vs. average duration (error cases removed).

It is possible that the sample used in this experiment may bias our result. Thus, to show how different settings influence the different techniques' performances, we plotted in Figure 11 the sum of the Fitts' law IDs between targets (e.g. the ID of moving from "flamingo" to "penguin" plus the ID from "penguin" to "mouse" in Figure 10) versus the average duration for each task for error free trials. On one hand, following Accot and Zhai [4], we see a changeover in performance between the standard crossing and point-and-click tasks. For small IDs, the standard crossing task outperforms the point-and-click task, while for large IDs the point-and-click task is faster. On the other hand, the relaxed semantic implementation seems to consistently outperform both for commonly observed IDs.

Of course, it might be the case that by improving the performance in composite command selections, we might have significantly degraded the ability to select one option at a time. To explore this, we conducted a short follow-up experiment to compare the error rates of when users select a single target from the center column of the dialog box in both crossing-based implementations. We asked 6 new participants to perform a total of 18 selections (selecting each target in the center column 3 times) using both implementations. Users were not told about the difference in functionality between implementations, and we balanced the order of presentation. Since the experiment was so short, participants did not receive any compensation for their participation. After removing outliers greater than three standard deviations from the mean, the average durations for each condition were within 6ms of each other and in both cases there were no errors. This suggests that our new algorithm improves multiple selection speeds without affecting single selection difficulty.

DISCUSSION AND FUTURE WORK

Together, our three experiments showed that the crossing-based versions of a dialog box have similar screen footprints and performance characteristics as a more traditional point-and-click dialog box. Our second experiment showed that blurring the distinction between the point-and-click semantic and the goal-crossing semantic can have a significant benefit on user performance during multiple selections.

We believe that our results have a strong implication for the deployment of crossing-based interfaces in the field. Our results imply that it might be possible to leverage the benefits of crossing-based interface within the framework of a more traditional point-and-click interface by changing the dispatch mechanism to accommodate the algorithm described above. In some systems such changes could be implemented through a simple update of the GUI base library. This implies that pen-based interface users might be able to smoothly transition from a fully point-and-click style of interaction to a mixed style of interactions with ease. Such a gradual approach will ease the acceptance of the crossing-based interface among users, compared to an abrupt change to a brand new interaction paradigm.

Our present work is only the first step in that direction, and we need to extend the external validity of our results by exploring more complex interactions (such as selecting more than 3 options) and conducting longitudinal studies of user performance in everyday tasks. To this end, we are planning to develop a new crossing-based interface toolkit that will make it easy for users to smoothly transition from point-and-click interfaces to crossing-based interfaces. Providing a drop-in implementation like this can also help decrease the dependence on point-and-click interfaces since it will make crossing-based components readily available, and thus cost effective, for designers.

CONCLUSION

In this paper, we provided a better understanding of the parameters influencing the performance of crossing interfaces. We explored the space-time tradeoff within the crossing-based dialog box and provided the first design rules indicating the optimal parameters for such a tradeoff. Finally, we proposed a new crossing-based interaction semantic that allows for faster and more fluid interactions. We believe our experiment results accompanied with our new algorithm will promote the deployment of crossing-based interfaces and thus strengthen pen-based interfaces.

AKNOWLEDGEMENTS

This work was supported by the NSF Grant 0414699 and by Microsoft Research as part of the Microsoft Center for Interaction Design and Visualization at the University of Maryland. We would like to thank Corinna Löckenhoff, Leianna Ridgeway, and Hyunyoung Song for providing many useful comments.

REFERENCES

1. Apple Developer Connection. 2006.
2. Accot, J. and S. Zhai. Beyond Fitts' Law: Models for Trajectory-Based HCI Tasks. *Proceedings of CHI'97*, pp. 295 - 302.
3. Accot, J., Les Tâches Trajectorielles en Interaction Homme-Machine—Cas des tâches de navigation., PhD thesis, Université de Toulouse 1. 2001
4. Accot, J. and S. Zhai. More than dotting the i's --- foundations for crossing-based interfaces. *Proceedings of CHI'02*, pp. 73 - 80.
5. Accot, J. and S. Zhai. Refining Fitts' law models for bivariate pointing. *Proceedings of CHI'03*, pp. 193 - 200.
6. Apitz, G. and F. Guimbretiere. CrossY: A Crossing-Based Drawing Application. *Proceedings of UIST'04*, pp. 3 - 12.
7. Cao, X. and R. Balakrishnan. VisionWand: interaction techniques for large displays using a passive wand tracked in 3D. *Proceedings of UIST'03*, pp. 173 - 182.
8. Cao, X. and S. Zhai. Modeling human performance of pen stroke gestures. *Proceedings of CHI'07*, pp. 1495 - 1504
9. Card, S.K., W.K. English, and B.J. Burr, Evaluation of Mouse, Rate-Controlled Isometric Joystick, Step Keys and Text Keys for text selection on a CRT. Ergonomics, 1978. **21**(8): p. 601 - 613.
10. Douglas, D.H. and T.K. Peucker, Algorithms for the reduction of the number of points required to represent a line or its caricature. The Canadian Cartographer, 1973. **10**(2): p. 112-122.
11. Fitts, P.M., The information capacity of the human motor system in controlling amplitude of movement. Journal of Experimental Psychology, 1954. **47**: p. 381 - 391.
12. Geissler, J. Gedrics: the next generation of icons. *Proceedings of INTERACT'95*, pp. 73 - 78.
13. Grossman, T., N. Kong, and R. Balakrishnan. Modeling pointing at targets of arbitrary shapes. *Proceedings of CHI'07*, pp. 463 - 472.
14. Lank, E. and E. Saund. Sloppy Selection: Providing an Accurate Interpretation of Imprecise Stylus Selection Gestures. *Proceedings of Computers and Graphics*, pp. 490 - 500.
15. MacKenzie, I.S. and W. Buxton. Extending Fitts' law to two-dimensional tasks. *Proceedings of CHI'92*, pp. 219 - 226.
16. Pastel, R. Measuring the difficulty of steering through corners. *Proceedings of CHI'06*, pp. 1087 - 1096.
17. Zhai, S. and P.-O. Kristensson. Shorthand writing on stylus keyboard. *Proceedings of CHI'03*, pp. 97 - 104.