Evaluation of Pointing Performance on Screen Edges

Caroline Appert, Olivier Chapuis, Michel Beaudouin-Lafon

Evaluation of Pointing Performance on Screen Edges

Caroline Appert[1,2,3] appert@lri.fr
Olivier Chapuis[2,3] chapuis@lri.fr
Michel Beaudouin-Lafon[2,3] mbl@lri.fr

[1]IBM Almaden Research Center
San Jose, CA, USA

[2]LRI - Univ. Paris-Sud & CNRS
Orsay, France

[3]INRIA
Orsay, France

ABSTRACT

Pointing on screen edges is a frequent task in our everyday use of computers. Screen edges can help stop cursor movements, requiring less precise movements from the user. Thus, pointing at elements located on the edges should be faster than pointing in the central screen area. This article presents two experiments to better understand the foundations of "edge pointing". The first study assesses several factors both on completion time and on users' mouse movements. The results highlight some weaknesses in the current design of desktop environments (such as the cursor shape) and reveal that movement direction plays an important role in users' performance. The second study quantifies the gain of edge pointing by comparing it with other models such as regular pointing and crossing. The results not only show that the gain can be up to 44%, but also reveal that movement angle has an effect on performance for all tested models. This leads to a generalization of the 2D Index of Difficulty of Accot and Zhai that takes movement direction into account to predict pointing time using Fitts' law.

Categories and Subject Descriptors

H.5.2 [**Information Systems**]: Information Interfaces and Presentation (e.g., HCI) – User Interfaces, Input Devices and Strategies

Keywords

Edge pointing, Screen Edges, Fitts' Law, performance modelling

General Terms

Human Factors, Experimentation, Performance

1. INTRODUCTION

Common graphical desktop environments display a number of interactive widgets along the physical edges of the screen. Microsoft Windows© and several X Window environments, e.g., GNOME and KDE, feature a *task bar*. This task bar contains buttons to navigate among application windows, and shortcuts to the files and applications used most often. It is also used to display notification icons, current time, sound controls or system status. Mac OS X© displays the menus of the foreground application and some notification icons in a *menu bar* that is always at the top of the screen. It also features a *dock* holding icons to quickly launch frequently used files and applications. Users can change the task bar or dock location but the system constrains them to one of the four physical edges of the screen (three for Mac OS X because of the menu bar).

Placing widgets along the edges makes it easier for users to organize their workspace, i.e., their windows and icons, in the central area of the screen without occluding these widgets. However, it also maximizes the distance between the working area and these "edge widgets". Since Fitts' law [9] predicts that the larger the distance between the cursor and a target, the longer the time to reach that target, edge widgets may impede pointing performance.

While pointing in the central screen area has been extensively studied and Fitts' law has been shown to hold in most cases, e.g. [8, 16], the situation with targets located on screen edges may be different. A typical pointing movement is composed of two main phases: an acceleration phase at the beginning of the movement and a deceleration phase at the end of the movement to stop the cursor within the target bounds [19]. Figure 1 (left) shows the typical profile of the speed curve for pointing at a "regular" target. When pointing at an edge target, however, users can take advantage of the physical boundary to stop the movement. They only have to stay within the bounds of the target along the direction collinear to the edge, while maintaining a high speed (prone to overshooting) along the main movement direction. Figure 1 (right) shows the expected speed curve when pointing at a target on a screen edge. Accordingly, edge pointing should be faster than "regular" pointing. The intuition that pointing at edge widgets should be faster has already been noted, e.g., [18], Chapter 4, or [5], but, to the best of our knowledge, it has never been empirically tested. Thus, we do not know if users can perceive the potential advantage and actually use edges in practice.

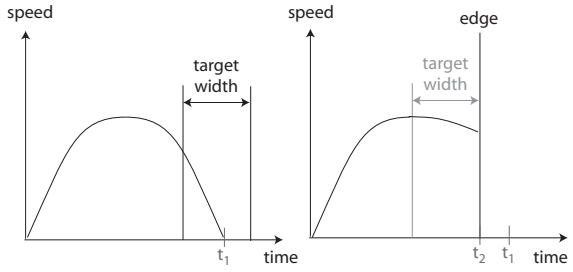


Figure 1: Speed curves for regular pointing (left) and for edge pointing (right).

C. Appert, O. Chapuis and M. Beaudouin-Lafon. Evaluation of Pointing Performance on Screen Edges. In AVI '08: Proceedings of the Working Conference on Advanced Visual Interfaces, 119-126, ACM, May 2008.
Authors Version

In this article, we present two experiments to better understand edge pointing and help interface designers in their desktop layout choices. The first experiment identifies the relevant factors involved in an edge pointing task and measures their effects on mouse movements and pointing performance. Results show that some factors such as cursor shape or movement direction have an impact on completion time and the use of edges. The second experiment quantifies the gain of using edges by comparing edge pointing with regular pointing and crossing [1]. It shows that movement angle has a strong effect on performance in all three cases and that differences between models increase with angle. We then propose a generalization of the 2D Index of Difficulty of Accot and Zhai [2] that captures the relation between pointing difficulty and movement direction to provide better predictions of pointing performance.

2. RELATED WORK

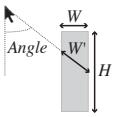
Regular pointing has been extensively studied and providing a full review of the literature is beyond the scope of this article. Since we are interested in identifying relevant factors that influence completion time of an edge pointing task, we give an overview of factors that have already been tested in regular pointing and the main findings of these studies.

The most common way of studying pointing is to measure movement time (MT) according to Fitts' Index of Difficulty (ID) on a one-dimensional pointing task [9]. Fitts' ID is a function of the ratio of two other factors: the distance to the target (D) and the width of the target (W):

$$MT = a + b \cdot ID, \quad \text{where} \quad ID = \log_2\left(\frac{D}{W} + 1\right)$$

This law means that the larger and the closer the target, the shorter the time required to point at it. Numerous studies have validated this model, see [14] for a review.

Over the past fifteen years, a number of studies have attempted to refine this model by taking into account other factors that might influence pointing performance in a realistic two dimensional environment. Since many targets are rectangular, a number of models of 2D pointing have been proposed. For example, MacKenzie and Buxton compared several models [15] and found that $ID_{W'}$ and ID_{min} were the most promising, with ID_{min} providing slightly better predictions :



$$ID_{W'} = \log_2\left(\frac{D}{W'} + 1\right) \quad (1)$$

$$ID_{min} = \log_2\left(\frac{D}{min(W,H)} + 1\right) \quad (2)$$

Accot and Zhai [2] criticized the similar importance attributed to target width and height in these models. They proposed a more complex model, noted ID_{az} in this article, that assigns a specific role to each of these two dimensions, and showed that it provides better predictions. They define target width as the side collinear to the movement direction, i.e., the amplitude constraint, and target height as the side orthogonal to the movement direction, i.e., the directional constraint. Each of these dimensions makes its own contribution to the task difficulty. In their study, $p = 2$, $\omega = 1$ and $\eta = \frac{1}{7.3}$ were the best values for the three free parameters:

$$ID_{az} = \log_2\left(\left[\omega\left(\frac{D}{W}\right)^p + \eta\left(\frac{D}{H}\right)^p\right]^{\frac{1}{p}} + 1\right) \quad (3)$$

Regarding movement direction, the ISO9241-9 standard for evaluating pointing devices [12] recommends to lay out targets in a circular pattern and to impose a specific order of appearance that forces participants to perform movements in every direction to obtain results that are valid whatever the movement direction. However, some studies have attempted to isolate and measure the effect of movement angle on completion time. Mackenzie and Buxton [15] used three different angles (0, 45 and 90 degrees) and found that moves along the horizontal and vertical axes were about the same while moves along the diagonal axis took 4% longer. Grossman and Balakrishnan [10] tested angles 0, 22.5, 45, 67.5 and 90 degrees and found that users were the fastest in horizontal movements. To our knowledge, the studies that have tested a wider range of angles have not given more fine-grained results. For example, Whisenand and Emurian [20] found that diagonal movements were slower than straight movements and that horizontal movements were the fastest. Hancok and Boot [11] and Boritz et al. [7] also tested angles all around the cursor with both left and right-handed users and found that movements to the right were the fastest with the right hand for right-handed users and a symmetric result for left-handed users.

Finally, a few studies have measured the effect of other factors such as target feedback or cursor shape. Akamatsu et al. [3] compared five different sensory feedback conditions (no feedback, auditory, colour, tactile, and a combination of the three). They found that feedback of any type decreases the final positioning time (between entering the target an selecting it) but has no significant effect on overall completion time. Regarding cursor shape, Po et al. [17] compared a circle cursor and four arrow cursors (upper-left, upper-right, lower-left and lower-right) and showed that (i) an arrow cursor is more efficient when it is oriented in the direction of movement and that (ii) a circle cursor is the most efficient on average and its performance is independent of the movement angle.

3. STUDY 1: RELEVANT FACTORS

The goal of Experiment 1 was to measure the effect of variables involved in an edge pointing task. First, since it is a pointing task, we tested the effect of two common variables: Index of Difficulty (ID) and movement angle. Second, we tested the effect of variables that can help users *feel* the edges. While interacting with a direct input device such as a stylus provides a physical feedback of screen edges, indirect input devices such as a mouse, which are commonly used with desktop interfaces, do not have this property. Let us see what happens in today's standard desktop interfaces.

The standard arrow cursor, which points toward the upper left, leads to a situation where only one pixel (the tip of the arrow) is (barely) visible when the cursor is located at the very bottom or right edge of the screen (Figure 2-c). This could reduce the potential gain of using edges since users have to perform a visual search to make sure that they are on the right target (and possibly additional movements to locate the cursor). This probably explains why, in such situations, additional feedback is added to the target under the cursor. For example, on Windows XP, the task bar icon under the cursor is slightly highlighted. On the contrary, Macintosh menus, which are always located at the top of the screen do not provide any additional feedback (until the mouse is clicked) since the cursor remains completely visible even when moved as far up as possible (Figure 2-b).

In this experiment, we considered three factors specific to edge pointing. First, we considered targets on the top (North) and bottom (South) edges. While the task bar and the Mac OS X dock can

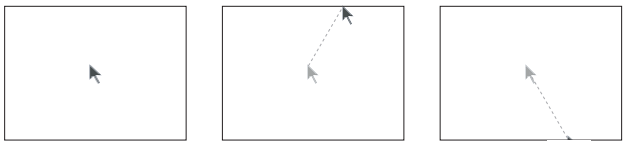


Figure 2: Edge and cursor visibility.

be on the left or right edges as well, we omit these cases to simplify the design and leave them for future work. The important point is that we have a condition where the arrow cursor almost disappears (South) and one where it is always visible (North). The second factor is target feedback: either targets are highlighted when the cursor is over them, or they are not. The third factor is the cursor type. We tested the traditional *arrow* cursor as well as a *circle* cursor. The *circle* cursor is symmetric and its hotspot is at its center. It remains visible whichever edge it is pushed against (Figure 3). This factor will help assess whether the observed effects of *Angle* and *Edge* are due to the *arrow* cursor orientation [17].

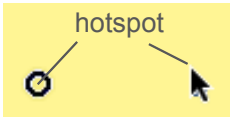


Figure 3: Circle cursor and Arrow cursor.

3.1 Apparatus

The experiment was run on a 2.66 GHz bi-processor PC running Linux with a Nvidia Quadro FX 1000 graphics card connected to a 1680 × 1050 LCD display (99 × 98 dots per inch). We used a standard optical mouse with the default linear X Window acceleration function. Our program was implemented in Java using the Touchstone run platform [13] and the SwingStates Toolkit [4].

3.2 Subjects

Twelve unpaid adult volunteers (11 male, 1 female), from 24 to 34 year-old (average 27.92, median 27), all right-handed, served in the experiment.

3.3 Task and Experiment design

Our experiment was a $2 \times 2 \times 2 \times 4 \times 7$ within-participant design. The following list summarizes the factors we tested:

- 2 *Cursor* conditions: *arrow* and *circle*,
- 2 *Feedback* conditions: *highlight* and *none*,
- 2 *Edge* conditions: *top* edge or *bottom* edge,
- 4 *Width* conditions: 20, 50, 100 and 200 pixels,
- 7 *Angle* conditions: -90, -60, -30, 0, 30, 60 and 90 degrees.

We used different angles and target widths to study edge pointing in a realistic context of use. -60, -30, 0, 30 and 60 degrees cover a good range of situations when the user is working at arbitrary screen locations. We also included -90 and 90 degrees to represent the frequent situation where a user moves the cursor horizontally to switch among window icons in a task bar or to explore different menus in the menu bar. For target widths, 20 pixels is roughly the size of a notification item while 50, 100 and 200 pixels represents a range of sizes for icons in the task bar or menus in the Mac OS X menu bar. To limit the number of trials, we used a fixed distance of 500 pixels and a fixed height of 20 pixels for the target (which is the typical height for a menu item or an icon in the task bar). Figure 4 illustrates the simple task participants had to perform: first click on a circular starting point and then click on the target. The next trial started only when the participant had clicked the target, i.e. every trial had to end successfully even if it included clicks outside the target.

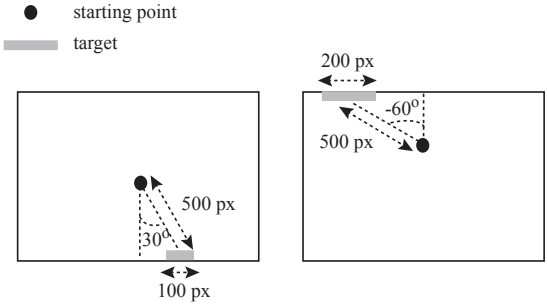


Figure 4: Two instances of the task used in Experiment 1. Left: $Angle = 30$, $Width = 100$, $Edge = bottom$. **Right:** $Angle = -60$, $Width = 200$, $Edge = top$.

We grouped trials into blocks according to the $Cursor \times Feedback$ condition, each block containing the 56 combinations $Edge \times Width \times Angle$. To counterbalance the presentation order of $Cursor \times Feedback$ blocks, we used a Latin Square to compute 4 presentation orders per participant, resulting in a design containing 4 trial replications per participant[1]. Each participant thus executed 16 blocks. Each block was divided into two series, one per *Edge* condition. We divided our participants into two groups of six participants. Participants in the first group performed these series in the order *top* then *bottom* while participants in the second group performed them in the order *bottom* then *top*. Within a series, participants had to perform 28 trials per $Width \times Angle$ condition, presented in a random order ($4 \times 7 = 28$ trials). Thus, the total number of logged trials in our experiment was: 16 blocks × 2 series × 28 trials × 12 participants = 10752 trials. Before starting the experiment, participants were instructed to point as fast and as accurately as possible and had to perform a series of trials with a sample of all the conditions they would face during the experiment.

Our software collected three main measures: completion time, number of "errors" (clicks outside the target) and click position.

3.4 Predictions

Before running the experiment, we made the following predictions:

H_1: *circle* cursor is more efficient than *arrow* cursor for $Edge = bottom$ because of the visibility problem caused by *arrow* cursor.

H_2: $Feedback = highlight$ is more efficient than $Feedback = none$ especially in condition $Cursor = arrow \times Edge = bottom$. Feedback should help users quickly perceive that they are in the target, making them more confident and avoiding a costly visual search for cursor location.

H_3: As in regular pointing, the larger the target, the shorter the completion time. Since we use a single target height and hypothesize

[1]Note that we do not use the same Latin Square for each participant so as to ensure that within a group of 6 participants, the $4! = 24$ possible orders are presented.

that participants will use edges to stop their movement, completion time should be a linear function of $ID_e = \log_2(1 + \frac{D}{Width})$. Contrary to Accot and Zhai [2] who defined width and height according to movement angle, we consider that target width is always the length of the target side collinear to the screen edge[2].

3.5 Results

We collected a total of 10752 trials. 690 of them included clicks outside the target (error rate = 6.41%). We did not remove these trials for our analyses since participants had to end each task successfully (errors are thus included in task completion time as a penalty).

There was a significant learning effect: *Block number* has a significant effect on completion time ($F_{31,341} = 2.4$, $p < 0.001$) and completion time decreases according to *Block number*. This should not affect the validity of our analyses since our counterbalancing strategy ensured that each condition appeared in every position across participants. This is supported by an analysis of variance that did not reveal an effect of presentation order on completion time: the interaction effect *Block number* $\times$ *Cursor* $\times$ *Feedback* on completion time is not significant.

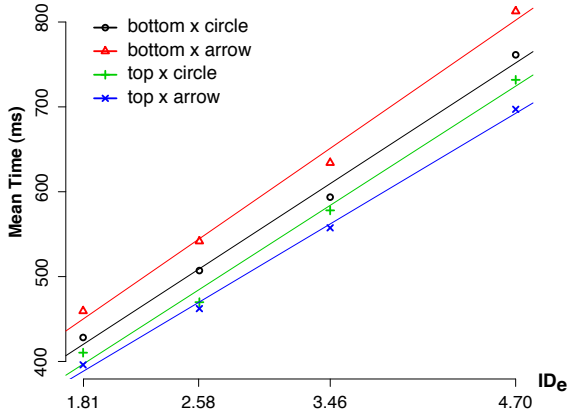


Figure 5: Mean time as a function of ID_e by *Edge* $\times$ *Cursor*

Analysis of variance did not reveal a significant effect of *Cursor* on task completion time but revealed a strong interaction effect of *Edge* $\times$ *Cursor* on task completion time ($F_{1,11} = 65.2$, $p < 0.0001$). Tukey post hoc tests showed that *bottom* $\times$ *circle* is significantly faster than *bottom* $\times$ *arrow* (a difference in mean of 40 ± 6 ms representing a speed up of 6.5%) and that *top* $\times$ *arrow* is significantly faster than *top* $\times$ *circle* (a difference in mean of 19 ± 6 ms representing a speed up of 3.4%). We found no significant difference between *bottom* $\times$ *circle* and *top* $\times$ *circle*. These results support hypothesis H_1. H_3 is also supported since we observed a significant simple effect of ID_e on task completion time ($F_{3,33} = 262.8$, $p < 0.0001$). Figure 5 illustrates these results.

Hypothesis H_2 however is rejected since there was no significant effect of *Feedback* on task completion time (nor any significant interaction effect of *Feedback* with any other factor on completion time). Analyses of the number of errors revealed that *Feedback* has a significant effect on number of errors ($F_{1,11} = 20.1$, $p = 0.0009$) with participants making significantly more errors in the *Feedback=highlight* condition (6.5%) than in the *Feedback=none* condition (5.1%). It seems that providing feedback by highlighting the object under the cursor is more disturbing than helpful in our experimental task. This is consistent with Akamatsu et al. [3] who observed no significant effect of feedback on completion time. We also observed a significant effect of ID_e on number of errors ($F_{3,33} = 12.8$, $p < 0.0001$). This is not surprising since clicking in a small target requires more precision than clicking in a large one. A linear regression of completion time as a function of ID_e, $MT = 208 + 114.ID_e$, shows a high correlation with an adjusted $r^2 = 0.992$ (we consider here 4 mean completion times per ID_e).

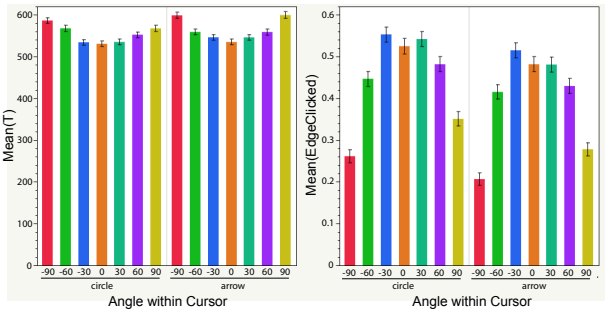


Figure 6: Mean time (left) and mean number of clicks along the edge (right) by *Angle* $\times$ *Cursor*

Our analyses showed that movement direction is an important factor for edge pointing. First, participants were faster at pointing upward than downward since we observed a significant effect of *Edge* on task completion time ($F_{1,11} = 45.8$, $p < 0.0001$). Second, performance varies according to the movement angle: *Angle* has a significant effect on task completion time for both edges ($F_{6,66} = 11.8$, $p < 0.0001$ for $Edge = bottom$ and $F_{6,66} = 4.1$, $p < 0.0014$ for $Edge = top$). This is probably because it is easier to use edges to stop when the movement is orthogonal to the edge, i.e. when *Angle* is close to 0. Comparing the mean completion time and the mean number of times users stopped on an edge according to the angle of movement supports this interpretation (Figure 6): participants stop more often on an edge for angles close to 0. Note that these results differ from those on regular pointing, in which users are faster in horizontal movements than in vertical ones [10, 15, 20].

We also observed an *Angle* $\times$ *Cursor* interaction effect on completion time ($F_{6,66} = 6.2$, $p < 0.0001$ for $Edge = bottom$ and $F_{6,66} = 2.7$, $p = 0.0202$ for $Edge = top$). This interaction effect seems stronger for $Edge = bottom$ probably because this edge suffers from the cursor visibility problem. Finally, there was a significant *Angle* $\times$ ID_e interaction effect ($F_{18,198} = 3.7$, $p < 0.0001$ for $Edge = bottom$ and $F_{18,198} = 4.1$, $p < 0.0001$ for $Edge = top$). The comparison of mean completion times across *Angle* $\times$ ID_e conditions revealed that performance is less sensitive to angle for easy pointing tasks, i.e., low IDs, than for difficult ones.

This first experiment revealed that edge pointing exhibits some differences with previous results on regular pointing, especially regarding the effect of movement angle. This is a motivation to further study edge pointing in order to better understand its underlying model and compare it with other models for target selection.

[2]Actually, using Accot and Zhai's definitions of width and height would have been confusing since we would have had to swap these two variables according to the value of the *Angle* factor. Indeed, Accot and Zhai use two movement angles (vertical and horizontal) and define target height as the directional constraint and target width as the amplitude constraint. In our case, the range of angles is much larger so one target dimension cannot be mapped directly to one of these constraints, as illustrated by Figure 12.

4. STUDY 2: PERFORMANCE GAIN

The goal of Experiment 2 is to identify a model for edge pointing. Our approach consists in comparing edge pointing with well-known models such as regular pointing or crossing, as well as a model that has not been studied yet, i.e., pointing a *Semi-Infinite* target, and that we hypothesize to be close to edge pointing.

4.1 Candidates for a model

Regular Pointing. In this model, based on Fitts' law, the user has to stop within the bounds of a finite target and click to select it. Edge pointing follows this model if users do not use edges to stop their movement.

Crossing+Click. Crossing was introduced by Accot and Zhai [1]: A target is a segment, and selection consists in overshooting the target with the pen down. Accot and Zhai showed that crossing a segment whose width is W can be more efficient than pointing a target of width W. Crossing follows Fitts' law but has lower empirical coefficients (a and b) when the segment is orthogonal to the movement direction. Crossing and edge pointing share the following property: the user does not have to perform the last part of the movement which consists in precisely stopping within the target. While crossing seems a good candidate to model edge pointing, crossing does not require a click to select the target (the selection is completed as soon as the user has crossed the segment). Therefore, we compare a variant of crossing that we call *Crossing+Click* which consists in first crossing the target and then clicking to actually select it. A pilot experiment revealed that it was hard for participants to know which target side they had to cross and to be sure that they had actually crossed it when the target was on the edge. Thus, in this experiment, we used a black line to indicate which side to cross (Figure 7-a) and the target was highlighted as soon as it had been crossed.

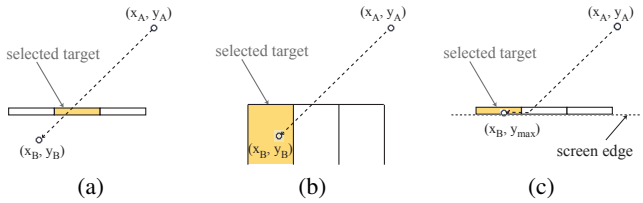


Figure 7: Selection by crossing (a), by pointing a semi-infinite target (b) and by edge pointing (c).

Semi-Infinite Pointing. A close look at current implementations of edge pointing in standard desktop environments shows that mouse movements along the x-axis are still taken into account once the top or bottom of the display is reached. We therefore introduce semi-infinite pointing. Figure 7 illustrates the difference with crossing. If a target is selected by crossing, only the position on the x-axis at crossing time is taken into account. This means that if the cursor has a diagonal trajectory and its speed would make it stop further along the edge, the part of the movement beyond the edge is ignored. On the contrary, if a target is selected by edge pointing, it is the x-position of the cursor when the click occurs that is taken into account to determine which target is selected. Therefore, in an edge pointing task, targets can be seen as semi-infinite, i.e., they are not bounded along the orthogonal direction of the edge. As mentioned earlier, pointing at targets with various W/H ratios has already been studied but only on a limited set of angles (0, 45 and 90 degrees in [15] and 90 degrees in [2]). Each study yielded a formula (ID_{min} and ID_{az}) that does not include the angle of movement.

We hypothesize that edge pointing is close to pointing a semi-infinite target, i.e., pointing a target with a W/H ratio close to zero, and that both models (ID_{min} and ID_{az}) do not capture this configuration properly since Experiment 1 has revealed a significant effect of angle of movement on movement time in edge pointing.

4.2 Task and Experiment design

We used the same hardware and software as in Experiment 1. Eight participants, all having already completed Experiment 1, also served in Experiment 2. The task also consisted in selecting a target but under different model conditions (Figure 8).

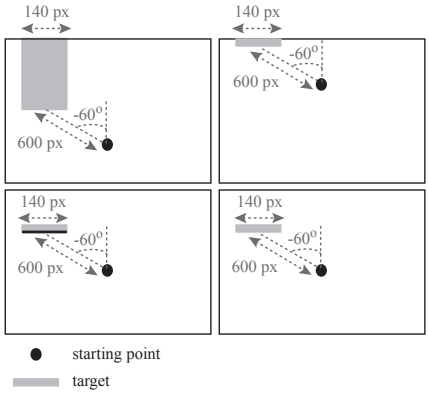


Figure 8: The 4 *Model* conditions for a given *Edge* × *Width* × *Distance* × *Angle* condition. Clockwise from upper-left: *Semi-Infinite*, *Edging*, *Pointing* *Crossing+Click*.

To limit the length of the experiment and focus on the study of the underlying model, we did not include *Feedback* and *Cursor* as factors in this experiment. Participants had to perform target acquisition tasks with a circle cursor and no feedback. This allowed us to study a wider range of ID. Our experiment was a $4 \times 2 \times 3 \times 2 \times 7$ within-participant design with the following factors:

- 4 *Model*: *Pointing*, *Crossing*, *Semi-Infinite* and *Edging*,
- 2 *Edge*: *top* edge or *bottom* edge,
- 3 *Width*: 35, 70 and 140 pixels,
- 2 *Distance*: 300 and 600 pixels,
- 7 *Angle*: -90, -60, -30, 0, 30, 60 and 90 degrees.

The trials were grouped into 12 blocks, 4 *Model* conditions repeated 3 times. Each *Model* block was divided into two sub-blocks, one per *Edge* condition and each of these sub-blocks contained 3 *Width* × 2 *Distance* × 7 *Angle* = 42 trials. The target height was 320 pixels in the *Semi-Infinite* condition while it was 20 pixels in all other conditions. To counterbalance the presentation order of conditions, we created 4 groups of 2 participants and computed 12 presentation orders for the *Model* condition using three Latin Squares. We concatenated 3 orders to compose a sequence of 12 blocks so we obtained 4 sequences, one per group of two participants. Within a group, one participant saw this sequence with sub-blocks in the order $Edge = bottom$ then $Edge = top$ while the other participant saw this sequence with sub-blocks in the order $Edge = top$ then $Edge = bottom$. Finally, the 42 trials of a *Model* block were presented in a random order. To summarize, the total number of logged trials in our experiment was: 12 blocks × 2 sub-blocks × 42 trials × 8 participants = 8064 trials. As in Experiment 1, participants were instructed to acquire the target as fast and as accurately as possible and had to perform a series of trials with a sample of all the conditions before starting the experiment.

4.3 Results

Before analyzing the results, we first checked that participants did not use the physical edge of the screen in the *Semi-Infinite* condition in order to avoid a confound with the *Edging* condition. The cursor reached the edge in only 0.34% of the trials and 99% of mouse clicks occurred within the first 250 pixels of the 320-pixels target.

Learning effect and error rate (6.05%) were similar to the ones observed in Experiment 1. Here again, our design counterbalanced learning effects since we did not observe a significant interaction effect of *Block number* $\times$ *Model* on completion time.

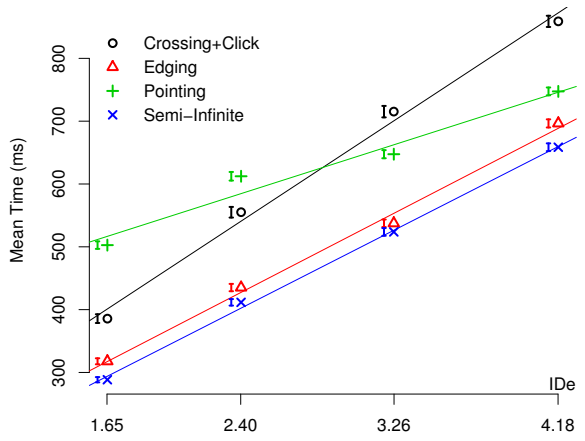


Figure 9: Mean time as a function of ID_e by *Model* (error bars are shown to the left of each symbol).

Analysis of variance revealed a significant effect of *Model* ($F_{3,21} = 141.3$, $p < 0.0001$) and ID_e ($F_{3,21} = 440.3$, $p < 0.0001$) on task completion time. We also observed a significant *Model* $\times$ ID_e interaction effect on task completion time ($F_{9,63} = 28.7$, $p < 0.0001$)[3]. Figure 9 illustrates these results: performance comparison among conditions depends on ID_e. First, Tukey post hoc tests showed that *Crossing+Click* is significantly faster than *Pointing* for easy tasks (i.e. $ID_e = 1.65$) and significantly slower than *Pointing* for difficult tasks (i.e. $ID_e = 4.18$). This result is consistent with Accot and Zhai [1]. Second, the difference between *Pointing* and *Edging* is larger for easy tasks than for difficult ones: Tukey post hoc tests showed that *Edging* is significantly faster than *Pointing* for all ID_e values, but the difference between mean completion times is 36.8% for $ID_e = 1.65$ while it is only 6.8% for $ID_e = 4.18$. Focusing our analyses on the *Edging* and *Semi-Infinite* conditions, we still observe a significant effect of *Model* ($F_{1,7} = 19.4$, $p = 0.0031$) and ID_e, but no *Model* $\times$ ID_e interaction effect ($F_{3,21} = 2.6$, $p = 0.0803$) on completion time[4]. Tukey post hoc tests showed that *Semi-Infinite* is significantly faster than *Edging* with a difference in mean of 26 ± 6 ms, this difference being almost similar across ID_e.

In summary, *Edging* and *Semi-Infinite* seem to follow a similar underlying model for ID_e and only differ by a small constant. *Pointing* and *Crossing* seem to follow different models.

Contrary to Experiment 1, we found no significant effect of *Edge* on completion time ($F_{1,7} = 2.9$, $p = 0.1339$), and no significant interaction effect of *Edge* with any other factor on completion time. This difference between the two experiments is probably due to the use of a single symmetric *circle* cursor. This allows us to simplify our analyses by considering *Angle* without distinguishing the *Edge* conditions. We found a significant effect of *Angle* ($F_{6,42} = 13.1$, $p < 0.0001$) and a significant ID_e $\times$ *Angle* interaction effect on completion time. Here again, we observe that movement time depends on movement direction (*Angle*) especially for easy selection tasks whatever the *Model* condition (*Model* $\times$ ID_e $\times$ *Angle* interaction effect was not significant).

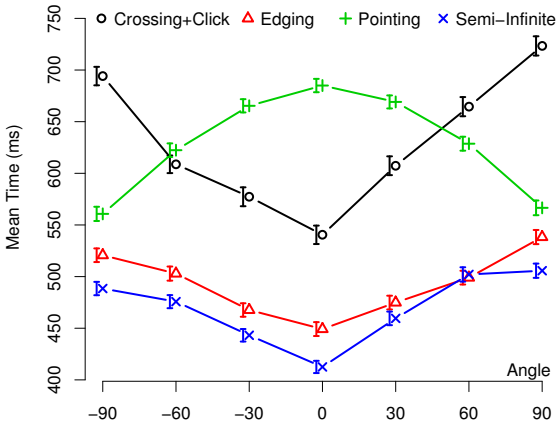


Figure 10: Mean time as a function of *Angle* by *Model* (error bars are shown to the left of each symbol).

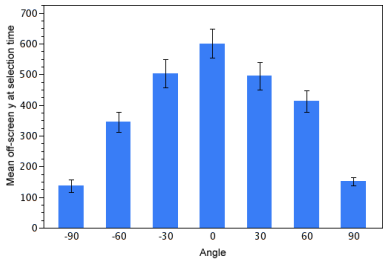


Figure 11: Mean cursor off-screen y-coordinate at target selection time according to *Angle* (*Model* = *Edging*).

Analysis of variance also revealed a significant *Model* $\times$ *Angle* interaction effect ($F_{18,26} = 19.7$, $p < 0.0001$) on completion time as illustrated in Figure 10. First, *Pointing* is faster for horizontal movements than for vertical movements, while *Crossing+Click* is faster for vertical movements than for horizontal movements. These results are consistent with the ones reported in previous work: [10, 20] showed that pointing is faster for horizontal movements than the two other angles they tested and [1] showed that crossing an orthogonal goal is faster than crossing a collinear goal in a continuous movement. Second, differences between *Pointing* and both *Edging* and *Semi-Infinite* are higher for vertical movements (i.e. *Angle* close to zero). For instance, Tukey post hoc tests showed that *Edging* is significantly faster than *Pointing* for *Angle* $= 0$ (a speedup of 34.0%) while there is no significant difference for *Angle* $= \pm 90$. This is probably due to the "virtual" target height in the *Edging* condition that offers a lower *amplitude constraint* for angles close to 0 than for angles close to 90 or -90. The histogram in Figure 11 supports this interpretation: it plots

[3] A finer analysis considering *Width* and *Distance* separately showed that this interaction effect was mainly an effect of *Width*.

[4] And no significant *Model* $\times$ *Width* and *Model* $\times$ *Distance* interaction effects when we consider *Width* and *Distance* separately.

the "virtual" y-coordinate[5] of the cursor at target selection time according to *Angle* in the *Edging* condition and shows that participants stopped their movement further away for angles close to 0 (i.e. vertical movements).

In summary, *Edging* and *Semi-Infinite* seem to follow a similar underlying model for *Angle* while *Pointing* seems to follow a different one. This result also supports our hypothesis regarding the similarity between the underlying models of *Edging* and *Semi-Infinite*.

5. DISCUSSION

The first important finding of this study is that users do take advantage of edges to facilitate target acquisition. Our analyses reveal that acquiring a target on an edge is similar to acquiring a target with a very large height: completion times for both tasks follow a similar function in terms of ID_e (Figure 9) and *Angle* (Figure 10).

The second important finding is that pointing at a target on an edge is quite different from pointing at the same target in the middle of the screen. First, the relationship between movement time and ID_e is different for the two conditions: while in both cases it is an increasing function of ID_e, differences between regular pointing and edge pointing are much larger for low ID_e values than for high ones (Figure 9). Second, the relationship between movement time and movement direction is different: for edge pointing, movement time seems to be a linear *increasing* function of the absolute value of *Angle* while for edge pointing, it seems to be a linear *decreasing* function of the absolute value of *Angle*. This results in performance differences between regular pointing and edge pointing between $+33 \pm 49$ ms (i.e. 4.4% of movement time) and -278 ± 20 ms (i.e. 44.6% of movement time).

As far as we know, the only model that takes movement direction into account is $ID_{W'}$ (eq. 1), which was introduced with ID_{min} (eq. 2) by Mackenzie and Buxton [15]. In their study, $ID_{W'}$ was shown to be less accurate than ID_{min}. Accot and Zhai raised issues with both models and introduced ID_{az} (eq. 3). The table below reports the linear regressions of completion times as a function of ID using each of these models (we consider the 42 mean completion times per condition *Angle* × *Distance* × *Width* for a given *Model*[6]):

Model	$MT = a + b.ID_{W'}$			$MT = a + b.ID_{min}$			$MT = a + b.ID_{az}$		
	a	b	r^2	a	b	r^2	a	b	r^2
Edging	205	127	0.80	80	144	0.90	-37	166	0.92
Semi-Inf.	178	129	0.80	59	143	0.87	-59	166	0.90
Pointing	233	106	0.73	-215	188	0.71	-253	181	0.76

Since ID_{az} contains three free parameters, we tested different combinations for $\omega \in [0,10] \times \eta \in [0,10] \times p \in \{0,1,2\}$ with a step of 0.1 for ω and η. Since the simple values $\omega = \eta = p = 1$, corresponding to $ID = log_2(\frac{D}{W} + \frac{D}{H} + 1)$, did not provide noticeably worse correlation coefficients, we use these values in the table.

Once again, these results support the hypothesis that *Edging* and *Semi-Infinite* follow the same underlying model but differ from traditional *Pointing*. The correlation coefficients however are not as good as for regular pointing, calling for a more detailed analysis.

[5]Even though the cursor is graphically blocked on the edge, we recorded input events directly from the mouse to compute the "virtual" off-screen location at target selection time.

[6]For *Edging* and *Semi-Infinite*, we approximate "infinite" height to 250 pixels, i.e. the height of the area that contains 99% of mouse clicks in the *Semi-Infinite* condition (see Section 4.3).

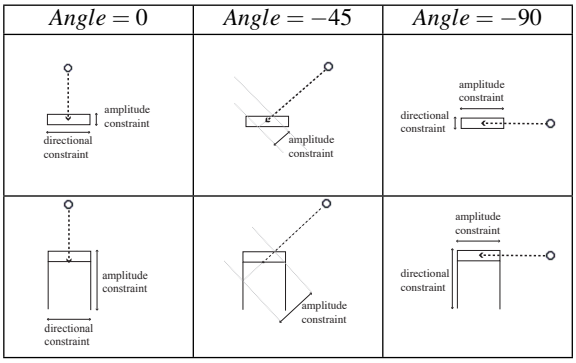


Figure 12: Amplitude and directional constraints for regular pointing (top) and edge pointing (bottom) according to movement direction.

Let us come back to the notions of *amplitude* and *directional* constraints defined by Accot and Zhai [2]. The *Amplitude* constraint is the interval within which the user must stop along the movement direction while the *Directional* constraint is the interval within which the user must stop along the direction orthogonal to the movement. In their study, Accot and Zhai only evaluated non-diagonal movements, so these constraints were simple functions of target width and target height. Figure 12 suggests that taking movement direction into account should help describe the task more accurately. We propose to introduce movement direction in the ID_{az} model based on two ideas mentioned by Accot and Zhai [2]: (i) satisfying an amplitude constraint takes more time than satisfying a directional constraint and (ii) the shortest side must dominate the ID.

To this end, we add a term that emphasizes the contribution of the shortest side to the ID, and we make this term a function of $|Angle|$. Figures 10 and 12 show that the larger the difference between the orientation of the shortest side and movement direction, the smaller the amplitude constraint. In the *Edging* and *Semi-Infinite* conditions (where W is the shortest side), this difference is an *increasing* function of $|Angle|$ while in the *Pointing* condition (where H is the shortest side), this difference is a *decreasing* function of $|Angle|$. We therefore propose the following model where the $|Angle|$ term captures the relationship between orientation of the shortest side and movement direction:

$$ID_{Angle} = \log_2\left(\frac{D}{W} + \frac{D}{H} + f(|Angle|).\frac{D}{min(W,H)} + 1\right)$$

$$f(|Angle|) = 0.6 \times sin(|Angle|) \text{ for } Edging \text{ and } Semi\text{-}Infinite$$

$$f(|Angle|) = 0.6 \times cos(|Angle|) \text{ for } Pointing$$

The table below and Figure 13 show that ID_{Angle} provides much better predictions than the other models studied above:

Model	$MT = a + b.ID_{Angle}$		
	a	b	r^2
Edging	-57	156	0.97
Semi-Inf.	-82	156	0.96
Pointing	-335	191	0.96

To select the functions $f(|Angle|)$, we balanced a trade-off between simplicity and prediction accuracy after systematically considering the following functions: $\{x \times |Angle|,\ x \times sin(|Angle|)\}$ for *Edging* and *Semi-Infinite* and $\{x \times (\frac{\pi}{2} - |Angle|),\ x \times cos(|Angle|)\}$ for *Pointing*, with $x \in [0,10]$ with a precision of 0.05.

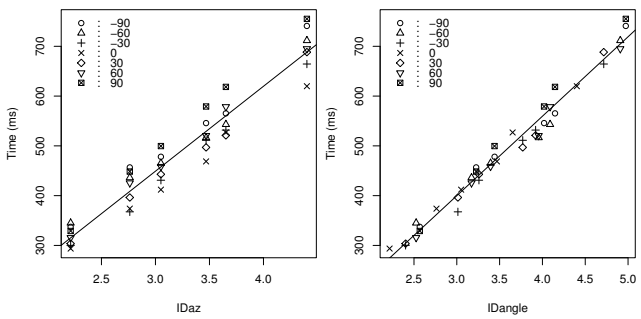


Figure 13: Movement time as a linear function of ID_{az} (left) or as a linear function of ID_{Angle} (right) for *Edging*.

6. CONCLUSION AND FUTURE WORK

We have presented an empirical study to better understand pointing at targets on screen edges. We have also proposed an analysis of the differences between regular pointing and edge pointing and shown that the angle of movement affects the amplitude constraint for a rectangular target. In order to account for these differences with Fitts' law, we have extended Accot and Zhai's definition of index of difficulty (ID) for bivariate pointing. While our model provides better predictions in the study presented here, its validity must be further tested by considering larger sets of target heights and edge orientations, i.e., left and right as well as top and bottom.

Having assessed the effect of various factors on edge pointing performance and compared different pointing models, we can draw the following recommendations to improve current desktop environments. First, the cursor should always be visible even when located on a screen edge. We have shown that a circular cursor shape does improve performance but that target highlighting does not. Other alternatives worth exploring in future work include displaying a small halo around the cursor when it is on the edge [6] or having virtual edges within a few pixels of the physical edges so that the cursor does not move to the physical edge and stays visible. Second, we encourage the use of edges for placing a widget if this does not significantly affect its average distance to the cursor in a typical context of use. The edge creates a "semi-infinite" target that can be acquired up to 44% faster than a regular target at the same distance in the central screen area. Third, we found that movements orthogonal to a given edge, i.e., with a zero angle, afford better performance. Designers should therefore lay out frequently used "edge widgets" close to the center of the edge. Note, however, that we have not tested the special case of corners, which are probably even faster to acquire than edge widgets.

Another research direction for this work is to explore whether "virtual" edges, such as the borders of a window, that would block the cursor under certain conditions, could also improve selection time in specific situations. Obviously, it is important to clearly identify when and how to activate and deactivate such virtual edges so that the user can easily access the rest of the screen. One idea would be to activate the virtual edges while *transient* graphical components, e.g., a pop up menu, are displayed.

7. ACKNOWLEDGEMENTS

We wish to thank Emmanuel Pietriga, our experiment participants and the anonymous reviewers for their feedback. Caroline Appert was supported by a grant from the French Lavoisier program.

8. REFERENCES

[1] J. Accot and S. Zhai. More than dotting the i's — foundations for crossing-based interfaces. In *Proc. CHI '02*, pages 73–80. ACM, 2002.

[2] J. Accot and S. Zhai. Refining Fitts' law models for bivariate pointing. In *Proc. CHI '03*, pages 193–200. ACM, 2003.

[3] M. Akamatsu, I. S. Mackenzie, and T. Hasbroucq. A comparison of tactile, auditory, and visual feedback in a pointing task using a mouse-type device. *Ergonomics*, 38(4):816–827, 1995.

[4] C. Appert and M. Beaudouin-Lafon. Swingstates: Adding state machines to the swing toolkit. In *Proc. UIST '06*, pages 319–322. ACM, 2006.

[5] AskTog. A quiz designed to give you Fitts, 1999. http://www.asktog.com/columns/022DesignedToGiveFitts.html.

[6] P. Baudisch and R. Rosenholtz. Halo: a technique for visualizing off-screen objects. In *Proc. CHI '03*, pages 481–488. ACM, 2003.

[7] J. Boritz, K. S. Booth, and W. B. Cowan. Fitts's Law Studies of Directional Mouse Movement. In *Proc. GI '91*, pages 216–223. Canadian Hum.-Comp. Comm. Soc., 1991.

[8] S. K. Card, 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. *Human-computer interaction: a multidisciplinary approach*, pages 386–392, 1987.

[9] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *J. Exper. Psych.*, 47:381–391, 1954.

[10] T. Grossman and R. Balakrishnan. A probabilistic approach to modeling two-dimensional pointing. *ACM Trans. Comput.-Hum. Interact.*, 12(3):435–459, 2005.

[11] M. S. Hancock and K. S. Booth. Improving menu placement strategies for pen input. In *Proc. GI '04*, pages 221–230. Canadian Hum.-Comp. Comm. Soc., 2004.

[12] ISO. 9241-9 Ergonomic requirements for office work with visual display terminals (VDTs)-Part 9: Requirements for non-keyboard input devices. *Inter. Org. for Standard.*, 2000.

[13] W. E. Mackay, C. Appert, M. Beaudouin-Lafon, O. Chapuis, Y. Du, J.-D. Fekete, and Y. Guiard. Touchstone: exploratory design of experiments. In *Proc. CHI '07*, pages 1425–1434. ACM, 2007.

[14] I. S. MacKenzie. Fitts' law as a research and design tool in human-computer interaction. *Hum.-Comput. Interact.*, 7:91–139, 1992.

[15] I. S. MacKenzie and W. Buxton. Extending Fitts' law to two-dimensional tasks. In *Proc. CHI '92*, 219–226. ACM, 1992.

[16] I. S. MacKenzie, A. Sellen, and W. Buxton. A comparison of input devices in element pointing and dragging tasks. In *Proc. CHI '91*, pages 161–166. ACM, 1991.

[17] B. A. Po, B. D. Fisher, and K. S. Booth. Comparing cursor orientations for mouse, pointer, and pen interaction. In *Proc. CHI '05*, pages 291–300. ACM, 2005.

[18] J. Raskin. *The Humane Interface: New Directions for Designing Interactive Systems*. Addison-Wesley, 2000.

[19] N. Smyrnis, I. Evdokimidis, T. Constantinidis, and G. Kastrinakis. Speed-accuracy trade-off in the performance of pointing movements in different directions in two-dimensional space. *Experimental Brain Research*, 134(1):21–31, 2000.

[20] T. G. Whisenand and H. H. Emurian. Some effects of angle of approach on icon selection. In *Proc. CHI '95*, pages 298–299. ACM, 1995.