DGP-TR-2004-003

April, 2004

Visual Features and Interference in Pressure Widget

Gonzalo Ramos, Ravin Balakrishnan

Visual Features and Interference in Pressure Widgets

Gonzalo Ramos, Ravin Balakrishnan
Department of Computer Science
University of Toronto
bonzo | ravin@dgp.toronto.edu

ABSTRACT
Pressure widgets are user interface elements that exploit the capabilities of pressure-sensing technology present in digitizer tablets and interactive pen-based display devices. They provide users with a visual indication of the amount of pressure being applied, as well as meaningful feedback as to the consequences of varying pressure in these widgets. We present empirical work that investigates how changes in key visual design dimensions of pressure widgets affect their usability. Our results indicate that variations in visual design, including changes in pressure to visual attribute mappings, can have significant impact on a pressure widget's usage speed, accuracy, and interference between the pressure and the spatial x-y movement components of the stylus. Our results can be used to refine the design of pressure widgets.

KEYWORDS: input, pen-based interfaces, isometric input, pressure widgets.

INTRODUCTION
Devices that support pressure-sensitive input, such as the TabletPC, or digitizing tablets, such as the Wacom Intuos, are quickly becoming more affordable and widespread. However, existing applications and user interfaces for these devices do not take full advantage of the potential degrees of freedom available to them, particularly in situations where a stylus is the primary or only mode of input. For instance, most styli currently used with tablet computers can provide an additional continuous degree of freedom that corresponds to the amount of pressure a user applies with the stylus' tip on the tablet's surface. To date, this pressure input has typically only been used by some drawing and image manipulation programs, like Adobe Photoshop, to control parameters of the active brush, such as stroke thickness or colour opacity.

In our previous work [10, 11], we introduced *Pressure Widgets* and presented the results of an experiment that investigated human ability, when provided with both partial and full feedback, to perform discrete target selection tasks by varying a stylus' pressure. Four different selection techniques, each involving an action performed with the stylus, were also examined. This study revealed that, for the tasks presented to the users, the presence of continuous visual feedback is necessary in order to achieve proper pressure control. When providing full visual feedback, we found that using pressure to select an item from a discrete set of options is not only feasible, but also practical, as long as the number of discrete targets available is less than or equal to six. This same study also revealed that different selection techniques significantly affected the users' performances. We showed that these differences in performance are a product of the intrinsic nature of the various selection techniques, whether it is the stylus movement, changes in the pressure that is applied, or the activation of the stylus' barrel button. Our observations and experimental results revealed that some users either inadvertently move the stylus while varying pressure, or unintentionally change the level of pressure when performing a selection gesture. We call this interaction between pressure and spatial x-y movement *self-interference* or *interference*, for short. Because *interference* can have a negative impact on the usability of pressure widgets, it is important to consider whether there are any factors that contribute to its presence, and whether it is possible to reduce interference, or better still, eliminate it. Our previous research proposes visual design dimensions and suggestions for the design of pressure widgets; however, it does not formally study how changes in these dimensions affect the interference phenomenon, influence the users' performances, and ultimately facilitate the use of pressure as an input channel. An answer to these issues can assist designers and may enable them to explore and select appropriate designs for pressure widgets.

In this paper, we address these issues first by reviewing the relevant literature. We then discuss pressure widgets and their visual design parameters, along with how such parameters can be related to input coming from the pressure channel. Building on this discussion, we then propose a number of discrete pressure widget designs, and examine how they tie into the aforementioned visual design parameters. We then evaluate these widgets in a controlled experiment that investigates how different visual design parameters affect users' interactions and performances. Next, we present the results of our study, and discuss the impact that variations in design parameters have on the pressure widget's usage speed, accuracy, and interference. We also use our experimental data to determine bounds on certain types of interaction, and discuss how these bounds can be used by designers. We conclude with a discussion of the implications of the experiment's results for the design and use of pressure widgets, and with suggestions for potential future research in this area.

DGP-TR-2004-003

RELATED WORK

Previous research studying pressure as a potential input channel in user interfaces include application prototypes that explore both the design and informal testing of interactions based on pressure, as well as formal studies investigating the boundaries of human performance and human capabilities when interacting with pressure.

Herot et al. [6] explored the ability of the human finger to apply pressure and torque to a computer screen. Their conclusions highlight the benefits of direct manipulation and the importance of continuous, real-time, visual feedback for the set of interactions they developed.

Buxton et al. [2] also explored the potential of touch-sensitive technologies and recognized the possibilities of the interactions they enabled. They also demonstrated how continuous pressure can be used to control the width of a brush tool in a painting application; a technique that now is commonly found in applications such as Adobe Photoshop.

In his thesis, Zhai [16] quantifies, through a series of experiments, the effects of controlling various dimensions of 6-dof input devices for 6-dof manipulation and tracking tasks. Most relevant to us is his observation that isotonic devices perform best when used for position or zero-order control, while isometric devices are best suited for rate or first-order control. Despite this research, it is interesting to observe that isometric devices continue to be used for position control, such as the IBM's Trackpoint joystick.

Srinivasan et al. [13] sought to measure human ability to control contact force against a rigid object, while determining the impact of different sensory feedback (i.e., the presence vs. the absence of visual feedback). While their results offer some insight into human performance for the particular task measured, their conclusions do not provide enough information about the impact that different *types* of visual feedback have on user performance.

Lécuyer et al. [8] carried out a series of experiments that compared the stiffness discrimination between a virtual spring and an equivalent actual spring using a SpaceBall isometric device. Their findings reveal that, with appropriate visual feedback, an isometric device can be used to simulate haptic information, thus offering the user the illusion of using a non-isometric device. In other related research, Tan et al. [14] show how knowledge of human biomechanical, sensorimotor, and cognitive abilities can guide the design of force-reflecting haptic interfaces.

Raisamo [9] studied four pressure-based area selection techniques available at an information kiosk that was equipped with a pressure-sensitive screen. He reported that even though pressure-based methods were challenging to control, the slowest pressure-selection technique ranked almost as high as a baseline direct manipulation technique. His conclusions suggest that appropriately designed pressure-sensitive interaction techniques could be a practical alternative to standard movement-based methods.

Komerska et al. [7] developed a haptic widget, which controls the viewpoint of a large 3D data space. When designing this haptic widget, special emphasis was put on both visual and haptic feedback. This widget shares many of the same design principles as pressure widgets. Both seek to inform the user not only of the current state of their input devices, but also of the possible available interactions.

In the context of a pen-based system, we show in [10] how the use of pressure can allow for a richer interaction vocabulary that can be used to explore the fluid control, navigation and annotation of digital video. Our system, called *LEAN,* uses pressure-sensitive widgets to navigate through the video stream, to select video frames, and to attach annotations to the system's workspace.

Today, it is possible to find many commercial isometric input devices that use pressure in some meaningful way. Examples include the SpaceBall controller, IBM's Trackpoint joystick, the DualShock2 controller for the Sony PS2 gaming console (with buttons that sense the pressure a player applies), and styli on digitizing tablets or Tablet PCs. New emerging technologies are also capable of sensing, at least to some degree, the pressure a user applies. Examples include Rekimoto's SmartSkin [12] and Mitsubishi's DiamondTouch table [3].

The literature we reviewed tends to agree that in order for pressure to be used in a meaningful way, user interfaces need to provide visual feedback. However, as we indicated in our introduction, there is no consensus as to what form this visual feedback should take.

PRESSURE WIDGETS

Pressure widgets are new user interface elements that exploit the capabilities of pressure-sensing input technology, as found in interactive display devices, or digitizer tablets. These widgets provide users with a visual indication of the amount of pressure being applied, as well as meaningful feedback as to the consequences of varying pressure. Pressure widgets can be *continuous*, if they control a continuous value, e.g., the speed of a video, or the opacity of an object; or *discrete*, if they are used to select an element or value contained in a finite, small set, e.g. a color from a palette, or the typeface of a font.

This paper studies discrete pressure widgets, and will refer to them simply as pressure widgets. We will begin with a description of the different factors that characterize pressure widgets in general and then discuss how these factors can be used to guide us in the visual design of specific pressure widgets.

Design Factors

We differentiate between two distinctive visual elements in pressure widgets: *cursor* and *target(s)*. We informally define *cursor* as the visual feature that indicates what item will be chosen if a selection occurs. We will refer to *targets* as the visual representation of the set of items available for selection.

Users' interactions with pressure widgets can be divided into two stages: *navigation*, where users apply the right amount of pressure so that the cursor indicates the desired target; and *selection*, where users effectively confirm picking the target identified in the navigation stage, and thus complete the interaction.

Our previous research [11] studies interaction with one type of discrete pressure widget and, along with other design recommendations, identifies the need to minimize *self-interference*. Interference is an undesirable feature, since most people find it difficult to both simultaneously move a stylus and effectively control the pressure applied with it. If a person's objective is to control pressure, a sudden spatial movement may be disruptive. Conversely, a carefully planned motion may be disrupted by a sudden, even intentional, change in pressure. These behaviors can be traced back to the open-loop nature of pointing or handwriting tasks, as reported by Woodworth and cited by Elliott et al. [4]. In the context of pressure widget design, there are three factors that could potentially impact on the amount of interference:

The widget's visual feedback: An improper widget visual design may lead to erratic pressure control, e.g. a person may not feel compelled to vary the pressure applied with the stylus, or a person may feel compelled to move the stylus, both of which cause interference.

The widget's selection method: A widget must provide a way for the user to select the desired target. For example, the selection method may rely on the x-y variation of the same stylus with which a user applies pressure for navigation. Such a selection method may disrupt the navigation task, causing the pressure applied to inadvertently vary before or during selection. Also, in this case, it is difficult for a system to identify the moment at which the navigation phase ends and the selection phase begins; i.e., the moment at which a person makes the decision to select, but prior to that decision being translated into a change in the stylus' parameters. For the purposes of this paper, we explicitly remove the selection phase from the tasks, in order to focus our study on how a pressure widget's visual feedback affects the navigation phase.

Tracking is factor that we will take into consideration in the design of pressure widgets. This factor can assume two values, *t_on* and *t_off*, and is of particular interest because of its potential impact on interference. A widget with *t_on* will tightly follow the stylus' x-y position. An example of such behavior can be found in Tracking Menus [5]. Conversely, a widget with *t_off* will maintain its location independent of the stylus' x-y position. This behavior can be found, for example, in context menus that are usually invoked with a right-click command in many applications.

Coupling and Mapping

As shown in [11], users benefit if they are provided with an indication of the amount of pressure they are applying. In order to do this we map the pressure value reported by the stylus to control either the widget's *cursor* or *targets*. We call this mapping *pressure coupling.* When the pressure is coupled with the widget's targets we call it *target coupling* (*TC*), and when it is coupled with the widget's cursor we call it *cursor coupling* (*CC*).

For both of these couplings, we consider three particular attributes of a widget's visual element that can be affected by changes in pressure: *position* if the coupling produces changes to an element's x-y coordinates; *scale* if the coupling produces changes in an element's size or scale; or *angle* if the coupling produces changes in an element's angle or orientation. We will refer to this relationship as *pressure mapping*. Using the above terminology, we can say for a particular widget that pressure is *mapped to* position, and *coupled with* target. When *mapping* and *coupling* both refer to the same input, in our case *pressure*, we will extend the use of the term *coupled*, e.g. for the previous example we can say that position is *coupled with* target, or conversely target is *coupled with* position.

Widget Designs

Here, we will present three types of pressure widgets, which exercise the design factors described previously.

The *Flag* is a widget (Figure 1) composed of a set of rectangular targets laid out vertically one after the other that give it an overall rectangular shape. In this widget, pressure is mapped to *position*, or more specifically, to the *vertical position*. For the *Flag*, *position* can either be coupled with *cursor* or with *target.* When *position* is coupled with *cursor*, we simply call the widget *Flag* (*F*). When *position* is coupled with *target*, we call the widget *Moving Flag* (*MF*). Though similar in appearance, *F* and *MF* exhibit different behaviors. *MF* keeps the cursor always "fixed" in the same position on the screen, while the target moves in concert with changes in pressure. This is unlike *F*, where the cursor moves vertically with changes in pressure, while the target is fixed. This type of widget has the property of clearly differentiating the low and high end of the pressure spectrum as opposites, which are graphically represented as the opposing minor sides of the widget's bounding rectangle.

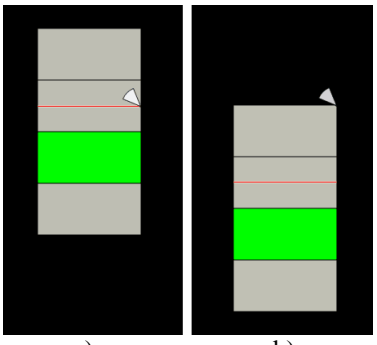


Figure 1: The Flag widgets. a) Moving Flag (*MF*): targets coupled with (vertical) position; b) Flag (*F*): cursor coupled with (vertical) position. The white wedges indicate the position of the stylus.

The *Pie* widget (Figure 3) consists of a set of sectors, or *targets*, which are laid around a circular path. The *cursor* for this widget is composed of a line or *needle* that pivots about the center of the widget, and points at one of the widget's sectors. The navigation phase on this widget is reinforced by an animation that magnifies and slides the sector, pointed to by the needle, in the direction at which it is currently aiming. For the *Pie* widget, pressure is mapped to *angle* and pressure can be coupled either with *cursor* or *targets*. When the *angle* is coupled with *cursor*, we will simply refer to the widget as *Pie* (*P*). When the *angle* is coupled with *target* we will call the widget *Rotating Pie* (*RP*). As was the case with the *Flag* widget, these two *Pie* variations behave differently depending on with what element (targets or cursor) the pressure is coupled. But unlike *F*, users of *P* may have to follow the cursor no further than the widget's radius, instead of a potentially distant target along a linear path. The circular nature of this widget makes it difficult to differentiate between the extremes of the pressure space, i.e., it is difficult, upon first glance, to establish where the low or the high ends of the pressure spectrum are for those targets positioned on a closed circumference. For this reason, we introduce a gap in the path where the targets lie separating the beginning and ending points of the target range. We also considered, as an alternative design, a *spiral-like* layout (Figure 2), where, in addition to the aforementioned gap, the beginning and ending targets were at a different radii, and all targets in between were along the spiral path defined by these two extreme radii. We chose to set this design aside, however, because pilot studies revealed that users were distracted and even confused by this design.

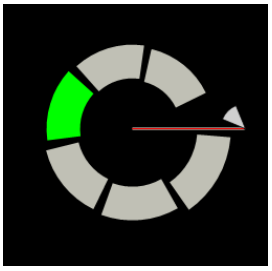
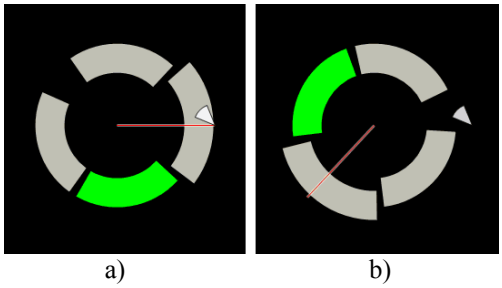


Figure 2: Early design for the Pie widget with 6 targets. Notice the spiral-like shape.



a) b)

Figure 3: The Pie widgets. a) Rotating Pie (*RP*): targets coupled to angle; b) Pie (*P*): cursor coupled to angle. Notice the large gap between the first and last target. The white wedges indicate the position of the stylus.

The *Bullseye* (Figure 4) consists of a set of concentric circular sectors, which correspond to different pressure level intervals, and a ring shaped cursor that changes its diameter depending on the amount of pressure exerted by the user. For this design, pressure maps to *scale,* and is coupled to the *cursor*. As the pressure changes, the cursor expands or contracts, falling into one of the rings.

The *Bullseye* widget was originally designed as a full circle. However, pilot studies revealed that users occluded with their hand the part of the interactive graphic display with which they were interacting. Consequently, we decided to cut a "wedge" out of it, on a region that will depend on the handedness of the user. We believe that by not occluding the widget, users will not have to move their hand in an effort to discover any significance in what they are occluding, and not moving one's hand reduces interference. Note that a usable *Bullseye* widget that couples pressure with *target* has yet to be developed.

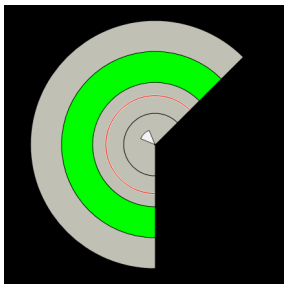


Figure 4: The Bullseye widget (*B*). The white wedge at the centre of the widget marks the position of the stylus.

Design Summary

A way to summarize the factors involved in the design of pressure widgets and the specific widget designs that we have presented, is to use a *design matrix*. This matrix, originally proposed in [11], can assist designers in the creation of pressure widgets, the same way taxonomies can help to identify and describe the nature of existing and potential input devices [1]. In the particular case of pressure widgets and the pressure channel, this *design matrix* has two dimensions: *coupling* and *mapping*. The matrix, shown in Table 1, allows us to inspect the design space of pressure widgets. By exploring all possible combinations of cells, one can imagine both the different visual designs and behaviors for pressure widgets.

	Pressure Mapping		
Coupling	***Position***	***Angle***	***Scale***
Cursor	-Flag	-Pie	-Bullseye
Targets	-Moving Flag	-Rotating Pie	

Table 1: *Design matrix* for the design of pressure widgets. Combinations of cells will describe the behavior for a particular pressure widget.

EXPERIMENT

Goals

Our goal is to investigate how the different visual design factors of pressure widgets, as previously discussed, affect users' interactions and performances. We are particularly interested in learning whether any of these attributes decrease interference, improve pressure control, foster faster target acquisition, and/or reduce error rates.

Apparatus

We used a Wacom Cintiq 18-SX interactive LCD graphics display tablet with a wireless stylus that has a pressure-sensitive isometric tip. It reports 1024 levels of pressure, and has a binary button on its barrel. The stylus does not provide any distinguishable haptic feedback. The experiment was done in full-screen mode, at the display's native resolution of 1280 by 1024 pixels, with a black background color. The tablet's active area was mapped onto the display's visual area in absolute mode. The experiment software ran on a 2GHz P4 PC with the Windows XP Professional operating system.

Task and Stimuli

A serial target acquisition and selection task was used. The stylus pressure and movement was used to control the behavior of different types of pressure widgets. We used the five different widgets described in the previous section: *Flag* (*F*), *Moving Flag* (*MF*), *Bullseye* (*B*), *Pie* (*P*) and *Rotating Pie* (*RP*). We also investigated the effects of *tracking* as applied to each of these widget designs.

During each experimental trial, one of the widget's targets was highlighted in green, and the user's task was to apply the appropriate amount of stylus pressure to make the cursor appear in that target, at which point the target's color changed from green to red. Participants confirmed the selection of the target by pressing the space-bar with their non-dominant hand. This isolated any interference that the selection event may introduce if it were coming from the stylus itself.

Participants

Seven female and eight male volunteers, 18-34 years old, participated in the experiment. All were right-handed and had little to no prior experience using pressure-sensitive devices such as the stylus used in the experiment.

Procedure and Design

A within-participants full factorial design with repeated measures was used. The independent variables were *widget* (*F*, *MF*, *B*, *P* and *RP*), *tracking* (*t_on*, *t_off*), and *breadth* or number of potential targets (4, 6, 8). For each trial, we collected all the stylus data events (position, pressure, and time). This allowed us to measure the time it took to perform a task, the result of the task (i.e. success or failure) the changes in the stylus' spatial position and extra information such as the number of times the cursor enters and leaves a target before the participant selects it. If a selection task was completed unsuccessfully (i.e. a selection was made outside the designated target), an audible beep was played, and the system presented the same selection task until it was completed successfully. With this procedure, participants cannot simply "race through" the experiment just so they could complete it as quickly as they may have, had we allowed them to continue to the next trial without correcting an erroneous selection. Also, this design provides us with a set of final selections for all conditions that are effectively error-free. Timing began the moment the stylus came into contact with the tablet's surface, (i.e. the tablet reported a pressure > 0) and it ended when a selection was made.

Based on the results from [9] we decided to use a simple hysteresis transfer function, designed to help users achieve better pressure control and to decrease overshooting effects. With this function, users need to overshoot or undershoot by more than a fraction *k* of the pressure interval *w*. For our experiments we set $k = 1/3$. Figure 5 shows the hysteresis function for a widget with *breadth* = 4.

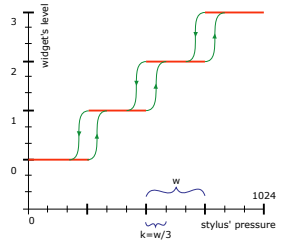


Figure 5: Hysteresis transfer function for breadth=4.

Participants were randomly assigned to 5 groups of 3 participants each. In each group, participants were exposed to all five widget conditions, whose order of appearance was balanced using a Latin square. For each widget condition, participants were asked to complete two sessions of 8 blocks each. The first session had *t_on*, and the second session had *t_off*. Each block consisted of 3 selection trials of for all 3 *breadth* conditions, repeated 4 times for a total of 36 trials per block. Presentation of trials within a block was randomized. In summary, the experiment consisted of:

15 participants x
5 widget conditions x
2 tracking conditions x
8 blocks x
3 selection tasks x
3 breadth conditions x
4 repetitions
= 43,200 target selection trials.

Prior to performing trials for each widget condition, participants were given a short warm-up set of trials to familiarize themselves with the widget. Participants were then instructed to perform the upcoming tasks as quickly and accurately as possible. Participants could take breaks between trials, and breaks were enforced during changes between widgets. The experiment lasted approximately 2 hours per participant. A short questionnaire was administered at the end, to gather subjective opinions.

Performance Measures

The dependent variables were *navigation time* NT – defined as the time from when the stylus came into contract with the tablet's surface until the user selected a target; *distance traveled* D – defined as the sum of pixels that the stylus traveled on both the horizontal and vertical direction during a trial; and *number of crossings* NC – defined as the number of times the cursor enters or leaves a target for a particular trial, (e.g., NC = 3 for a task where the user overshoots once and then reacquires the target). This last metric tells us about the degree of pressure control that participants exerted. We also computed the *error rate* ER – defined as the percentage of trials for a particular condition that resulted in an erroneous selection.

RESULTS

A trial was considered an outlier if the time it took to complete the task was beyond 2 standard deviations from the mean *NT*. A total of 1630 outliers were discarded, representing 3.7% of the data collected.

Target vs. Cursor Coupling

We first study the effects of target coupling (*TC*) versus cursor coupling (*CC*) by analyzing two separate groups of widgets: the *position* group (*F* and *MF*), and the *angle* group (*P* and *RP*).

For the *position* group, analysis of variance showed a significant main effect on *ER* for widget ($F_{1,21} = 37.8$, $p < .0001$). That is, users consistently made fewer errors when using *TC* (4%) than when using *CC* (6.5%). *NT* was also significantly different across widgets in the group ($F_{1,21} = 41.17$, $p < .0001$). Selections made using *TC* are 40ms faster. For the same group, *D* was significantly different across widgets ($F_{1,21} = 710.69$, $p < .0001$). Here we observe that users move the stylus less with a widget that uses *CC* (8 pixels) than with the same widget using *TC* (12 pixels). This result is very interesting, and is the reverse of what was expected from *TC* in [10]. *NC* was not significantly different for both widgets, indicating that *NC* was independent of either forms of position coupling. Figure 6 illustrates these effects.

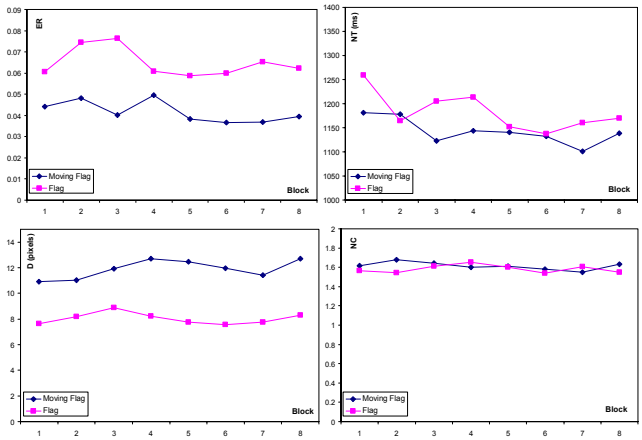


Figure 6: ER, NT, D, and NC by block for the *position* group.

In contrast to this result are the subjective impressions of users. Participants were asked to rate widgets on their "ease of use", with 1 being best and 5 being worst. For this question the overall ranking was *MF, P, F and RP* with *MF* beings rated as the best widget.

For the *angle* group, analysis of variance shows a main effect on *ER* for widget ($F_{1,21} = 32.31$, $p < .0001$). Users consistently made fewer errors when using *CC* (3.2%) than when using *TC* (5%). *NT* was also significantly different across widgets ($F_{1,21} = 234.28$, $p < .0001$). *CC* results in selections that are 100ms faster than using *TC*. It is interesting to notice that there was no significant difference in the distance traveled by the stylus, *D*, across widgets. Even though the stylus traveled an average of 10 pixels, this distance was not influenced by either *TC* or *CC*. It is also noteworthy to observe that, unlike the position group, *NC* significantly differed across widgets ($F_{1,21} = 13.77$, $p < .001$). Users made fewer crossings when interacting with *TC* (1.5) than when using *CC* (1.6). It this case, users' subjective impressions agree with the observed data, with participants rating *P* "easier to use" than *RP*. Figure 7 illustrates these effects.

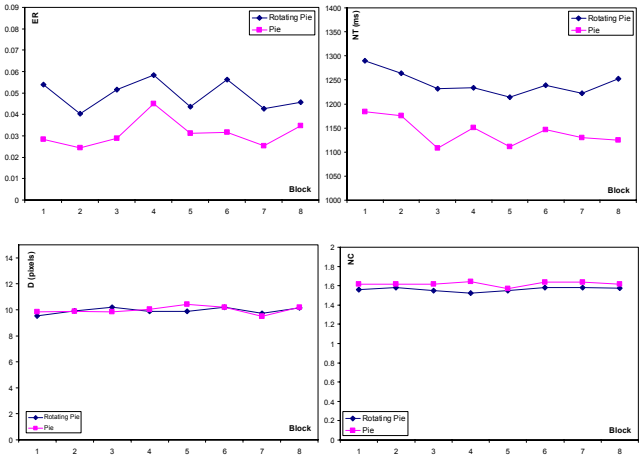


Figure 7: ER, NT, D, and NC by block for the *angle* group.

To summarize, the choice of target versus cursor coupling influences user performance along various measures including speed, error, interference, and control (*NC)*. This impact, however, differs depending on whether the pressure maps to *position* or *angle* (unfortunately, we did not have the chance to test this difference for pressure to *size* mapping, as we have yet to the design a usable *Bullseye* widget that maps pressure with target size).

Position vs. Size vs. Angle

For a fixed widget element, we are also interested in the effect that different types of mappings (*position, size* and *angle*) has on the interaction, depending on coupling. Thus, we analyze two separate groups of widgets: The *TC* group (*MF* and *RP*), and the *CC* group (*F, B* and *P*).

For the *TC* group, analysis of variance showed a significant main effect on *ER* for widget ($F_{1,21} = 5.33$, $p = .05$); where

users consistently made fewer errors when using *position* mapping (i.e., MF) (4%) than when using *angle* mapping (i.e., RP) (5%). Also for the same group the differences in *D* are significant for widget ($F_{1,21}$ = 192.67, p < .0001), namely the stylus traveled more with *position* than with *angle* (an average of 12 pixels vs. 10 pixels). Our analysis also shows a significant main effect on *NT* for widget ($F_{1,21}$ = 219.52, p < .0001), making *angle* an average of 100ms faster than *position*. The analysis also shows significant differences on *NC* across widgets ($F_{1,21}$ = 10.51, p < .001). However, a closer look at the data reveals that there is no main effect on *NC* across *block* for *angle*, but there is an effect for *position.* Users improve pressure control when using *angle* until they reach a level comparable with *position*. This observation is reinforced by the fact that differences in *NC* across widgets in this group are not significant after the 4th block. Figure 8 illustrates these effects.

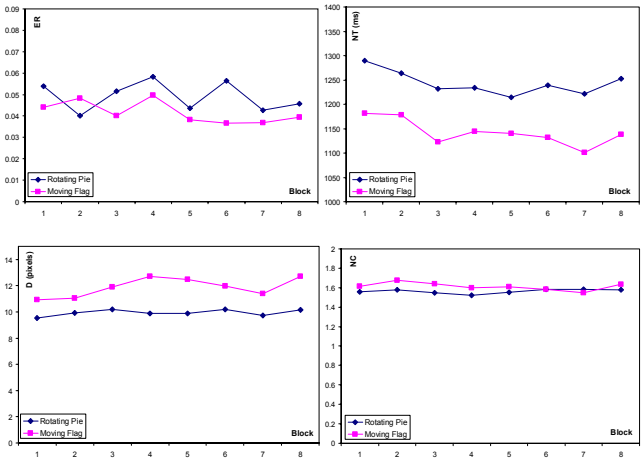


Figure 8: ER, NT, D and NC by block for the *TC* group.

For the *CC* group, analysis of variance showed a significant main effect on *ER* across widgets ($F_{2,42}$ = 45.44, p < .0001); users made fewer errors with *angle* (i.e., P) (3%) than with *size* (i.e., B) (5.8%) or *position* (i.e., F) (6.5%). *D* was also significantly different across widgets ($F_{2,42}$ = 247.25, p < .0001), with *angle* being the condition that caused the stylus to travel the most (10 pixels), followed by *position* (8 pixels) and *size* (7.6 pixels). There was also a significant effect on *NT* for widgets ($F_{2,42}$ = 40.99, p < .0001). However, pairwise means comparison shows that *size* and *angle* are not significantly different, and *position* is significantly different from *size* and *angle* ($F_{1,21}$ = 79.27, p < 0.0001). The average means indicate that *angle* is faster than *size* by 10ms, which is faster than *position* by 60ms. There was a significant effect on the number of crossings *NC*, for widget ($F_{2,42}$ = 7.81, p < .001). However, pairwise means comparison shows that only *size* and *angle* are significantly different (p < .001). The average number of crossing for *position*, *size* and *angle* are 1.6, 1.56 and 1.62 respectively. Figure 9 illustrates these effects.

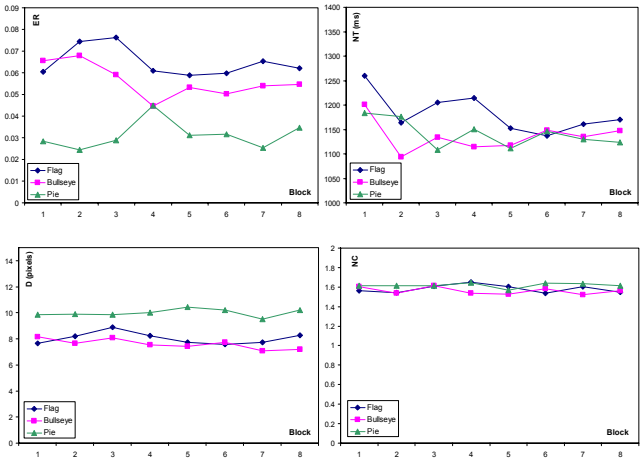


Figure 9: ER, NT, D and NC by block for the *CC* group.

To summarize, the choice of mapping pressure to *position*, *angle,* or *size* has a significant influence on various performance measures. However, as was the case for the previous section, this impact varies depending on the element to which coupling is applied (*cursor* or *target*).

The Effects of Tracking

To study the effects of the *tracking* condition we examine how its levels (*t_on*, and *t_off*) affect *ER, D, NT* and *NC* for each widget separately.

For *MF,* analysis of variance showed no significant main effect on *ER* for *tracking*. When considering the first six blocks, our analysis shows no significant main effect on *D* for tracking. However, the inclusion of the last two blocks makes the differences in *D* significant for tracking ($F_{1,21}$ = 5.29, p < .05). For these last blocks, the stylus travels an average of 10 pixels for the *t_on* condition, and an average of 12 pixels for the *t_off* condition. There was also a significant effect on *NT* for tracking ($F_{1,21}$ = 26.68, p < .0001), with *t_off* resulting in 40ms faster targeting than *t_on* across all blocks. There is also a significant effect on *NC* ($F_{1,6}$ = 26.68, p < .0001). Figure 10 illustrates these effects.

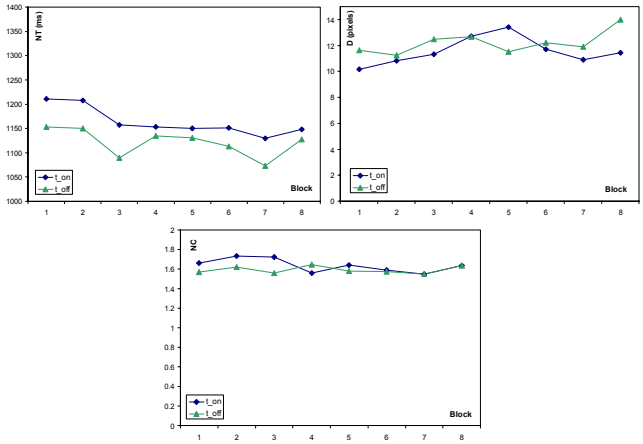


Figure 10: Effect of tracking on NT, D, and NC by block for the Moving Flag (*MF*) widget.

For *F,* analysis of variance showed no significant main effect on *ER*, or *D* for *tracking*. However, differences in *NT* were significant for *tracking* ($F_{1,21}$ = 64.81, p < .0001). Targeting was 80ms faster with *t_off* than with *t_on* across all blocks. Analysis of variance also shows significant differences on *NC* for tracking ($F_{1,21}$ = 26.68, p < .05). On average, there were fewer *crossings* with *t_off* (1.5) than with *t_on* (1.6). Figure 11 illustrates the significant effects.

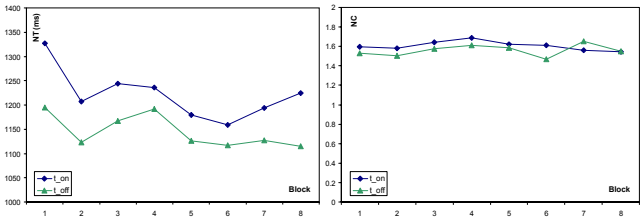


Figure 11: Effect of tracking on NT and NC by block for the Flag (*F*) widget.

For *B,* analysis of variance showed no significant main effect on *ER* or *NC* for *tracking*. There was a significant effect on *NT* for tracking ($F_{1,21}$ = 42.56, p < .0001). Tasks performed with *t_off* were on average 60ms faster than tasks with *t_on* cross all blocks. There was also a significant effect on *D* for tracking ($F_{1,21}$ = 20.37, p < .0001). The stylus traveled an average of 8 pixels with *t_on*, and an average of 7 pixels with *t_off*. Figure 12 illustrates the significant effects.

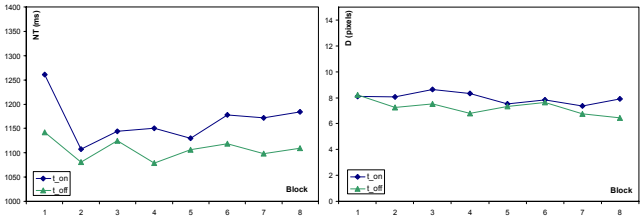


Figure 12: Effect of tracking on NT and D by block for the Bullseye (*B*) widget.

For *RP*, analysis of variance shows no significant main effect on *ER* or *NC*. The analysis also shows that significant differences in *D* occur only on the last four blocks ($F_{1,9}$ = 32.01, p < .0001). There was a significant effect on *NT* for tracking ($F_{1,21}$ = 37.02, p < .0001). Results show that targeting tasks with *t_off* were 50ms faster than with *t_on*, across all blocks. However, while *NT* degraded over time for the *t_off* condition, it improved substantially over time in the *t_on* condition. Figure 13 illustrates the significant effects.

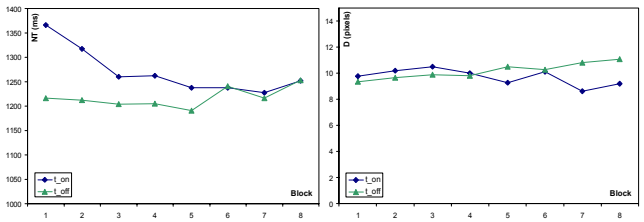


Figure 13: Effect of tracking on NT and D by block for the Rotating Pie (*RP*) widget.

For *P*, analysis of variance shows no significant main effect on *ER, D* or *NC* for tracking. However, there was a significant effect on *NT* for tracking ($F_{1,21}$ = 57.28, p < .0001). Targeting was about 70ms faster with *t_off* than with *t_on* across all blocks. Figure 14 illustrates this effect.

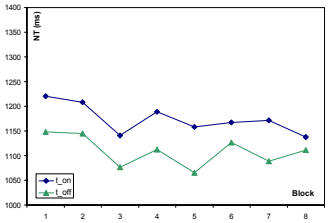


Figure 14: Effect of tracking on NT by block for the Pie (*P*) widget.

In summary, we can draw a number of conclusions from our results. First, for all widgets, tracking has no influence on the error rate of the selection tasks. Also, while tracking had no effect on *D* for widgets with *TC* for the first half of the trials, it later affected *D* in the same way for both widgets in the *TC* group. For the case of widgets with *CC*, tracking affected *D* only for the case of cursor coupled with size (i.e., Bullseye widget *B*). When this occurs, *D* is less with *t_off*. Tracking had a significant impact on *NC* only for the *position* group (*F* and *MF*), which we believe makes sense, since users may get confused trying to identify the source of the widget's change in position. Finally, tracking had the same influence on *NT* for all widgets where selections made with *t_off* were consistently faster that selections made with *t_on*.

The Effects of Breadth

Our previous research [11] indicates that performance measures should degrade as *breadth* increases. Our current results also exhibit this trend. Analysis of variance shows that *breadth* had a significant effect on all performance measures: *ER* ($F_{2,42}$ = 106.36, p < .0001), *NT* ($F_{2,42}$ = 1200.48, p < .0001), *D* ($F_{2,42}$ = 51.93, p < .0001) and *NC* ($F_{2,42}$ = 1064.08, p < .0001). However there were no interactions between *breadth* and *widget* for *ER, NT,* and *NC*, meaning that performance measured by these metrics degraded at the same rate for all widgets. That was not the case for *D*, where there was a significant *widget* x *breadth* interaction ($F_{8,168}$ = 4.82, p < .0001), meaning that the traveled distance of the stylus increased at different rates for different widgets (Figure 15).

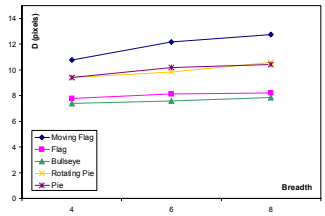


Figure 15: Effect of breadth on D for all widgets.

Overall Performance Trends over Blocks of Trials

There were no significant variations in *ER* across blocks, for all widgets. There was also no significant *widget* x *block* interaction for *ER,* indicating that regardless of visual feedback, users maintained a high level of performance through the entire experiment (94% - 97%).

Analysis of variance shows that from block 3 onwards there is no significant variation in *NT* across blocks for all widgets. This suggests that there is a point at which users stop improving in selection speed for a particular widget.

Our study also shows no significant variation in *NC* across blocks, and no significant *widget* x *block* interaction for *NC*. This makes us suspect that, in spite of the presence of visual feedback, pressure control remains at fixed levels (between 1.55 and 1.6) for the duration of the experiment. We observe that the average values of *NC* for the Flag (*F)* were lower (averaging 1.6) than the ones reported in [11] where *NC* was on average 2.8. We attribute this improvement to the use of the hysteresis transfer function (Figure 5). This improvement leads us to believe that there are perhaps more opportunities for improving pressure control by designing an appropriate transfer function than by producing a radically new visual design.

There are no significant variations in *D* across blocks, for *RP* and *P*. However, differences in *D* are significant across blocks for *F* ($F_{7,147} = 4.84$, $p < .0001$), *MF* ($F_{7,147} = 3.87$, $p = .001$) and *B* ($F_{7,147} = 2.97$, $p < .005$). From looking at the data, we see that *D* increases across blocks for *MF*, suggesting that some fatigue occurs during the experiment. Figure 16 illustrates these effects.

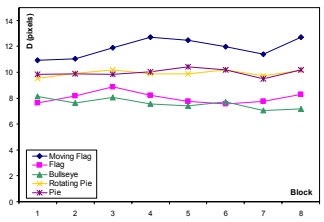


Figure 16: D per block for all widgets.

Subjective Evaluation

In our study, we gathered subjective information from participants through the post-experiment questionnaire.

All users quickly learned how to manipulate the widgets. Participants were asked on a 7-point "agree – strongly disagree" Likert scale if they felt they knew how to use a particular widget after a few trials. Answers averaged above 5 (in the "agree – strongly agree" interval) for all widgets except the *Rotating Pie*. This last widget scored at 4.7, falling in the "somewhat agree – agree" interval.

Our experimental results are also consistent with the participants' subjective evaluations of their performances. We asked participants to rate on a 7-point "agree – strongly disagree" Likert scale if they believed their pressure control degraded through the experiment. The average result was 1.45, falling in the "disagree – somewhat disagree" interval.

Diversity in participants' personal preferences surfaced when we asked them which widget they prefer to use, or which one they find visually attractive. For these questions, participants rated each widget numerically from 1 to 5 (with 1 being "best"). Even though the differences in the scores were not significant, participants rated *MF* highest (2.2) on preference and *RP* last (3.4). This assessment was reversed for visual attractiveness, where *RP* was ranked highest (2.2) and *MF* was ranked second lowest (3.2), only better than *F* (3.7).

Users almost unanimously commented as to how quickly they were able to select a target when it was the last one on the pressure intervals. Similar to the case of target selection in x-y space with a target of infinite width, users performed a fast, ballistic increase in pressure that reached the last interval without the risk of overshooting. In this regard, the recommendation made by Walker et al. [15] remains true: borders are effective in decreasing selection time. However, unlike spatial movement, pressure space has only one "effective" upper limit, or border. Still, this observation can help us choose what actions to map to different pressure intervals, e.g. the last interval of a discrete pressure widget could be assigned to a frequently used operation.

DISCUSSION

Designers possess different goals when selecting or creating user interface elements. For example, while some may want to place more emphasis on speed or error rate, others may favor aesthetics or user preference. It is not uncommon to find situations, however, where these goals may be mutually exclusive, e.g., a widget with low error rates may also have a slow selection speed. Recognizing this potential conflict, and building on the results of our experiment, we suggest the following heuristics for the visual design of pressure widgets:

To decrease interference, CC or angle mappings are better. Decreasing interference is important in order to isolate unintentional from intentional variations in the pressure a users applies with a stylus. Designs in which the targets' attributes did not change in response to variations in pressure resulted in less interference than those that did. Designs where pressure was mapped to angle, i.e. *Pie* and *Rotating Pie* also served to decrease interference, though to a somewhat lesser degree.

To achieve low error rates at fast speeds, Moving Flag and Pie are best. Both these pressure widget designs exhibited comparable error rates at below 5%, along with demonstrating the fastest selection speeds. These widgets were also ranked the top two "easiest to use" designs, in the users' subjective evaluations.

A cross between these heuristics makes *Pie* the overall best design, as it rates well in terms of interference, speed, accuracy and ease of use.

Our research also reveals measurable interaction upper bounds that can be used as a reference to assist practitioners

in the design of pressure widgets. Through our experiment, we observed that the distance traveled by the stylus, while the participants were applying pressure, does not exceed 12 pixels. By analyzing the data we collected from the user trials, we believe that some of this distance (in the order of 4 pixels on average) can be explained by the action of "landing" the pen onto the tablet's surface. Thus, discarding the few very first incoming packets from the stylus can potentially eliminate part of the interference. Being able to provide an upper bound to the amount of a stylus' movement, when interacting with a particular pressure widget, is a measurable contribution of this paper. With this result, for example, one can add to the widget an appropriate orthogonal selection technique using the stylus' movement in x-y space; or consider a widget design with a known, but accountable interference level, such as the *Moving Flag*.

CONCLUSIONS AND FUTURE WORK

We have presented a study that investigates how different pressure widget attributes affect a widget's usability. The results of our experiment indicate that a widget's design can impact significantly on metrics such as interference, speed, and error rate. We are able to suggest heuristics to help designers select the visual design features of pressure widgets that best accommodate their design goals: speed, error rate, or interference. From the results of our experiment, we are also able both to quantify concrete upper bounds on the interference coming from the stylus interaction, and to suggest how designers can use this information to create complementary selection techniques or reconsider previously rejected designs.

Future work in this area includes the design and subsequent study of pressure widgets used in the selection of continuous values. Also worthy of study is the question of how current user models for menu navigation and selection times could be applied to the analysis of pressure widgets. Finally, the use of an interactive graphics display in our experiment has further reinforced the need for user interface elements that compensate for the potential occlusion caused by the user's hands on the display.

ACKNOWLEDGMENTS

We thank all who participated in the experiment, and members of the Dynamic Graphics Project lab (www.dgp.toronto.edu) at the University of Toronto.

REFERENCES

1. Buxton, W. (1983). Lexical and pragmatic considerations of input structures. *Computer Graphics*, *17*(1). p. 31-37.
2. Buxton, W., Hill, R. and Rowley, P. (1985). Issues and techniques in touch sensitive tablet input. *ACM SIGGRAPH*. p. 215-224.
3. Dietz, P. and Leigh, D. (2001). DiamondTouch: a multi-user touch technology. *ACM UIST*. p. 219-226.
4. Elliott, D., Helsen, W.F. and Chua, R. (2001). A century later: Woodworth's (1899) two-component model of goal-directed aiming. *Psychological Bulletin*, *127*(3). p. 342-357.
5. Fitzmaurice, G., Khan, A., Pieké, R., Buxton, B. and Kurtenbach, G. (2003). Tracking menus. *ACM UIST*. p. 71-79.
6. Herot, C. and Weinzapfel, G. (1978). One-point touch input of vector information for computer displays. *ACM SIGGRAPH*. p. 210-216.
7. Komerska, R., Ware, C. and Plumlee, M. (2002). Haptic interface for center-of-workspace interaction. *IEEE Virtual Reality - Haptics Symposium*. p. 352-353.
8. Lécuyer, A., Coquillart, S., Kheddar, A., Richard, P. and Coiffet, P. (2000). Pseudo-haptic feedback: Can isometric input devices simulate force feedback? *IEEE Virtual Reality Conference*. p. 83-90.
9. Raisamo, R. (1999). Evaluating different touched-based interaction techniques in a public information kiosk. *Conference of the CHI Special Interest Group of the Ergonomics Society of Australia*. p. 169-171.
10. Ramos, G. and Balakrishnan, R. (2003). Fluid interaction techniques for the control and annotation of digital video. *ACM UIST*. p. 105 - 114.
11. Ramos, G., Boulos, M. and Balakrishnan, R. (2004). Pressure Widgets. To appear in *ACM CHI*.
12. Rekimoto, J. (2002). SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. *ACM CHI*. p. 113-120.
13. Srinivasan, M.A. and Chen, J. (1993). Human performance in controlling normal forces of contact with rigid objects. *Advances in Robotics, Mechatronics, and Haptic Interfaces, ASME*, *49*. p. 119-125.
14. Tan, H., Srinivasan, M., Eberman, B. and Chang, B. (1994). Human factors for the design of force-reflecting haptic interfaces. *Proceedings of the ASME Winter Annual Meeting*. p. 353-359.
15. Walker, N. and Smelcer, J.B. (1990). A comparison of selection time from walking and pull-down menus. *ACM CHI*. p. 221-226.
16. Zhai, S. (1995). Human performance in six degree-of-freedom input control. Ph.D. Thesis, *Department of Industrial Engineering*, University of Toronto.