Position vs. Velocity Control for Tilt-Based Interaction

Robert J. Teather*
McMaster University

I. Scott MacKenzie**
York University

Abstract

Research investigating factors in the design of tilt-based interfaces is presented. An experiment with 16 participants used a tablet and a 2D pointing task to compare position-control and velocity-control using device tilt to manipulate an on-screen cursor. Four selection modes were also evaluated, ranging from instantaneous selection upon hitting a target to a 500-ms time delay prior to selection. Results indicate that position-control was approximately 2× faster than velocity-control, regardless of selection delay. Position-control had higher pointing throughput (3.3 bps vs. 1.2 bps for velocity-control), more precise cursor motion, and was universally preferred by participants.

Keywords: Fitts' law, pointing, position-control, velocity-control.

Index Terms: H.5.2. Information interfaces and presentation (e.g., HCI): User Interfaces – evaluation/methodology.

1 Introduction

Tilt control is now available in virtually all mobile devices and tablets, and in many game controllers. The ubiquity of this technology arises from the low cost of accelerometers coupled with the interaction possibilities afforded by the technology. Most mobile devices change their display orientation upon rotating the device. Many mobile games are tilt-based, for example, racing games where the player tilts the device like a steering wheel, and "marble maze" games where tilting rolls a ball to simulate gravity.

Although tilt control intuitively *seems* natural for the above examples, it tends to underperform relative to direct touch [2, 11, 30], another universal technology. One wonders if game developers might intentionally choose tilt to *increase* game difficulty. It is possible that minor changes to the control scheme such as using position instead of velocity control could offer better user performance. To assess this hypothesis, we used a widely accepted paradigm of pointing device evaluation to assess tilt performance in a general interaction task.

Our research focuses on an on-going investigation of human performance aspects of tilt control as an interaction primitive. Motivated by our previous work [17], the major objective of this work is to compare different control modes and different selection modes. Specifically, the research described herein is the first to compare position-control and velocity-control for tilt-based input in a pointing task.

We also expand our previous work [17] which only compared two selection modes: immediate selection when the cursor enters the target, and delayed selection after a 500-ms dwell time upon target entry. The rationale for the delay is that in any real user interface, target disambiguation is required. However, the delay duration limits the upper bound of performance. Hence, we look at several selection delays to determine a setting that minimally impacts performance and does not interfere with the user's subjective impression of the system.

* teather@mcmaster.ca
** mack@cse.yorku.ca

2 Related Work

2.1 Tilt-Based Interaction

Tilt control has long been of interest to HCI researchers. We present previous work in several areas. See Table 1.

2.1.1 UI Tasks

Early tilt research focused on list/menu navigation [21], scrolling [1, 6, 19, 24], document browsing [3], and changing display orientation [9]. The cited papers all present implementation *examples*; performance was not quantified. Here, we focus on user performance for basic input control using tilt for target selection.

Wang et al. [27] used vision-based motion tracking instead of an accelerometer for tilt control. They proposed several tasks using tilt, including a pointing task, game control, and text entry. The pointing task, which used Fitts' law [4], is the closest to our evaluation. However, they only investigated 1D cursor control in four directions, possibly due to sensor imprecision. Performance as indicated by Fitts' throughput was about 1 bps. The authors speculated that accelerometer-based tilt control may offer better performance due to lower processing requirements. Other work confirms that tilt-based pointing and scrolling conform to Fitts' law, but did not report pointing throughput [22]. Later research [17] using accelerometer-based tilt control indicates that, for multi-directional pointing, throughput is as high as 2.5 bps.

2.1.2 Text Entry

Several researchers investigated tilt for text entry [13, 20, 23, 28]. Wigdor and Balakrishnan [28] used device tilt to disambiguate letter selection and reported that their technique was faster but more error-prone than MultiTap. Unigesture [23] partitioned letters into seven "zones" corresponding to seven tilt directions, with an eighth zone for SPACE. Like T9, Unigesture uses dictionary-based disambiguation to determine the word the user is entering. GestText [13] is similar as it partitions text into zones. Two options for partitioning were examined, with a matrix-based layout (like Unigesture) found to be superior. TiltType [20] is also similar, but accesses letter groups via physical buttons and uses tilt to disambiguate. The technique was not evaluated, though. While these studies suggest promise in tilt-based text entry, how well the results generalize to other tasks is unclear.

2.1.3 Games

Tilt-controlled games are increasingly common and popular on mobile devices. Tilt offers an alternative to touchscreen-based controls, which perform poorly relative to physical controls [2, 11, 30]. This performance discrepancy is largely attributed to the absence of tactile feedback in touchscreens [29].

Much of the work on tilt control for games is qualitative, for example, looking at user experience [5, 26]. However, there is also quantitative work [2, 18]. Browne and Anand [2] compared tilt to touchscreen controls for a shooting game on an Apple iPod Touch. Participants could play the game significantly longer with tilt than with touch controls. However, this was in part because the virtual buttons did not support multi-touch; i.e., participants could not move and shoot simultaneously, unlike with tilt.

First Author	Tilt Control	Device	Task	Main Findings
Browne [2]	Velocity	Smartphone	2D shooter game	Tilt performed better and more preferred than swiping or on-screen buttons
Gilbertson [5]	Velocity	Mobile Phone	3D tunnel game	Participants had higher scores with tilt than keypad
Henrysson [7]	Velocity	Mobile Phone	3D rotation	Tilt slower than device motion and keys
Henrysson [8]	Velocity	Mobile Phone	3D rotation	Tilt faster than touchscreen, slower than keys
Hynninen [11]	Velocity	Smartphone	First-person shooter games	Tilt performed worse than virtual joystick controls on touchscreen
Jones [13]	Position (gestures)	Wiimote	Text entry	Entry speed of around 5 wpm after four days with two difference soft keyboard layouts
MacKenzie [17]	Velocity	Tablet	ISO pointing task	Tilt conforms to Fitts' law, lower performance than mouse
Medryk [18]	Velocity	Smartphone	Pong-like game	Touchscreen (swiping) faster and higher scores than tilt
Oakley [19]	Velocity and position	Handheld computer	Menu Navigation	Tilt using position control faster than velocity control
Sad [22]	Position	PDA with tilt sensor	Scrolling, pointing	Tilt control in both tasks conforms to Fitts' law
Sazawal [23]	Position	Custom	Text entry	Mainly qualitative, error counts also reported
Valente [26]	Position	Mobile Phone	Navigation via audio	Tilt control feasible for non-visual games
Wang [27]	Velocity	Mobile Phone, tilt from camera	Multiple tasks, pointing	Fitts' pointing throughput of about 1.1 bps, differences based on motion direction
Wigdor [28]	Position	Mobile Phone	Text entry	TiltText 23% faster, about 2x as error prone as multitap

Table 1: Overview of empirical performance studies on tilt control.

Other researchers report contradictory results. Medryk and MacKenzie [18] found tilt was inferior to touchscreen control: Participants had worse game scores and accuracy, and took longer to complete levels in a *Pong*-like game. This may be due to the nature of the touch-control. Instead of displaying virtual joysticks and buttons, the game was controlled with swipe gestures. Similarly, Hynninen [11] found that tilt-control was inferior to touch-based virtual joystick control in a first-person shooter game.

The results on tilt-based gaming are mixed and difficult to generalize due to the complexity of the game tasks. The effectiveness of tilt control may thus be task-dependent.

2.1.4 Mobile Augmented Reality

The power, portability, and availability of multiple sensors (including cameras) make smartphones an interesting platform for mobile augmented reality (AR). Several studies investigated the use of tilt for object manipulation in a phone-based AR system. These systems typically use the device's accelerometers, but sometimes instead use the camera to detect device motion [27].

Henrysson et al. [7] used phone tilt to control 3D object rotation, comparing four techniques for single-axis rotation. The techniques included two using phone tilt, one using phone displacement, and one using keys. They found that the tilt methods were slowest. A later study on object rotation [8] compared tilt to key/joystick and touchscreen input. Consistent with earlier results, the authors found key-based input superior to tilt. However, tilt outperformed the touchscreen method, suggesting positive benefits in using tilt for input control.

Other researchers [10] report that moving objects in an AR environment using the movement/tilt of the device was faster than direct or indirect touch-based interfaces. Selection was slower using tilt than direct touch, however.

2.1.5 Summary

Of the studies cited above, only Oakley et al. [19] compared velocity and position tilt control. They used a menu navigation task. We seek to better understand these two control types in more general tilt-based interaction. Overall, few studies seem to consider position-control at all, likely because velocity-control is a more reality-founded control style. Our work is motivated on the hypothesis that position-control will offer better performance and could be employed in new or re-imagined tilt-based designs.

2.2 Fitts' Law and Pointing

Our evaluation employs Fitts' law [4], which models the speed-accuracy tradeoff in rapid aimed movements. The prediction form of Fitts' law is

$$MT = a + b \times \log_2\left(\frac{A}{W} + 1\right) \tag{1}$$

where MT is movement time. The constants a and b are empirically derived via linear regression. The log term is the index of difficulty (ID, in bits) where A and W are the amplitude (distance to the target) and width (target size), respectively. Essentially, the model implies that far, small targets are harder to hit than near, large targets. Harder tasks take longer, as reflected by MT. This has been formalized in an international standard for pointing device evaluation, ISO 9241-9 [12]. Figure 1 depicts the standard's 2D pointing task.

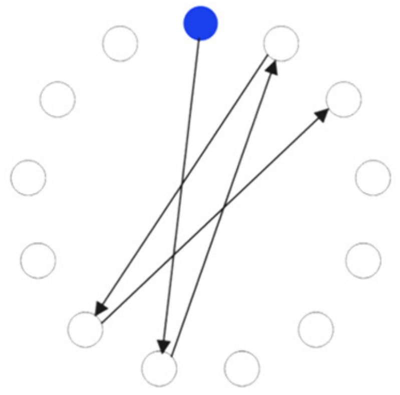

Figure 1: ISO 9241-9 2D tapping task with 13 circles (12 targets). Arrows indicate the first five targets.

The standard suggests using "effective" measures for width and amplitude to better account for user performance. This offers some advantages [15, 25], but effective measures are not used in the current study because the task precludes missing targets. Consequently, and similar to previous work [17], we instead calculate throughput using presented ID and average MT as

$$TP = \frac{ID}{MT} \tag{2}$$

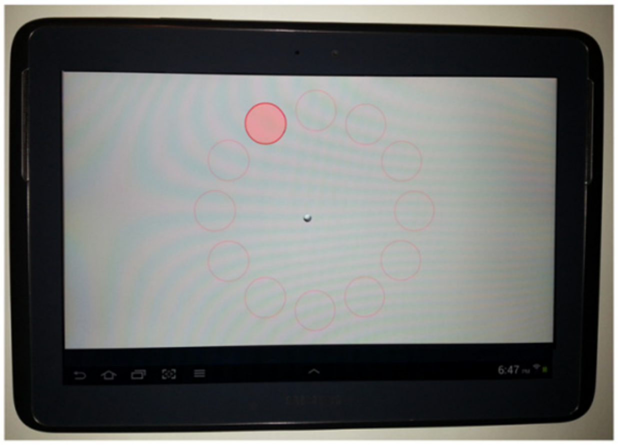

Figure 2: Samsung *Galaxy Note 10.1* tablet running the experimental software.

Finally, while throughput quantifies performance, it does not explain performance differences. Hence, we also employ accuracy measures (target re-entries, movement variability, and errors) [16] to help explain performance differences.

3 Position- and Velocity-Control

The main factor studied in our experiment is control mode. We investigated two options: position-control and velocity-control. Both use the tablet tilt but result in dramatically different experiences. Velocity-control is reminiscent of marble-maze games, as the cursor (a ball) increases its movement speed based on the amount of device tilt. Position-control is quite different: The farther the device is tilted, the farther from center the cursor is positioned. There is a direct correspondence between tilt angle and cursor position. Upon leveling the device, the cursor returns to the display center. The algorithms to determine ball position in each control mode are explained in the Apparatus section.

Previous work has shown that position-control affords higher performance than velocity-control in a tele-manipulation task [14]. Also, tilt-based position-control was superior to velocity-control for menu selection [19] on a mobile phone. There appears to be no prior empirical evaluation comparing tilt-based velocity- and position-control in general point-select tasks, though.

4 Methodology

4.1 Participants

Sixteen participants (8 male) took part in the study. Ages ranged from 18 to 47 ($\mu = 25$, $\sigma = 7.3$ years). All were undergraduate students enrolled in an introductory computer course. Participants were not regular tilt users – on a five point scale (1 meaning they never used tilt control and 5 meaning they used it every day), the mean response was a 3.1 (*SD* = 1.4).

4.2 Apparatus

The experiment was conducted on a Samsung *Galaxy Note 10.1* tablet with Google's *Android 4.1.2 (Jelly Bean)* OS. See Figure 2. The display resolution was 1280 × 800 pixels and measured 260 mm (10.1") diagonally. Pixel density was 149 pixels/inch.

Software was developed in Java using the Android SDK. Tilt control used the device's orientation sensor, fusing data from the accelerometer, gyroscope, and magnetometer. The sensor sample rate was 100 Hz. Two different methods were used to convert device pitch and roll to tilt magnitude and tilt angle, resulting in the two control modes investigated.

4.2.1 Calculating Ball Position[1]

Both control modes computed tilt magnitude and angle as:

$$tiltMag = \sqrt{pitch^2 + roll^2} \tag{3}$$

$$tiltAngle = \operatorname{asin}\left(\frac{roll}{tiltMag}\right) \tag{4}$$

These data controlled the ball direction and speed in velocity-control mode, and the ball position in position-control mode. In velocity-control mode, the ball "rolled" according to the tablet tilt. The ball velocity (v, in pixels/second) was a linear function of tilt magnitude and a programmable tilt gain setting:

$$v = tiltMag \times tiltGain \tag{5}$$

With each sample, the ball displacement (*dBall*, in pixels) was calculated as the product of the velocity and time since the last sample (*dt*, in seconds):

$$dBall = dt \times v \tag{6}$$

As an example, if the tilt magnitude was 3° and tilt gain was set at 50, then velocity was 3 × 50 = 150 pixels per second. If the sample occurred 20 ms after the previous sample, the ball was moved 0.02 × 150 = 3.0 pixels in the direction of the tilt angle. Based on previous results [17], tilt gain was fixed at 100.

The second control mode used position- or absolute-control, and behaved somewhat like an isotonic joystick. The farther participants tilted the tablet, the farther the ball was positioned from the home position. "Centering" the tablet – setting it down, or otherwise positioning it flat – re-centered the ball. Ball position was determined according to the following equations:

$$dBall = tiltMag \times tiltGain \tag{7}$$

$$offset = (dBall * \sin(tiltAngle), dBall * \cos(tiltAngle)) \tag{8}$$

$$ballPos = \text{center} + \text{offset} \tag{9}$$

The *offset* vector was added to the center position to determine the ball position. Tilt gain for position-control was fixed at 20. This resulted in a reasonable or nominal user experience, comparable to velocity-control with a tilt gain setting of 100.

4.2.2 Task and Target Parameters

The software implemented a variation on the ISO 9241-9 [12] 2D targeting task. See Figure 3. The task required tilting the tablet to hit the highlighted target with the 20-pixel diameter ball. Three target amplitudes (*A*) were used: 131, 263, and 526 pixels. Three target sizes (*W*) were used: 42, 63, and 105 pixels. See Figure 4. Note that the targets were effectively smaller, since the ball had to be completely inside the target for selection. Consequently, we subtract the ball diameter from the target diameter in the calculation of *ID*.

The *A-W* conditions yielded nine combinations of *ID* calculated according to Equation 1 and subject to the modification discussed above. *IDs* ranged from 1.35 to 4.64 bits. For each condition, there were 13 targets (12 selections), yielding 12 recorded trials per sequence. Each trial ended upon successful selection of the target, hence missing targets was impossible. In addition to the two control modes detailed above, four target selection delays were studied. The first was 0 ms delay. In this mode, selection occurred as soon as the ball was completely inside the target. The remaining selection modes required keeping the ball (completely) inside the target for a specified duration. Extending previous work [17], we used three non-zero levels of selection delay: 500, 400, and 300 milliseconds.

[1] These equations are a fast and accurate simplification of the exact equations, which require converting Euler angles to an axis-angle representation. The error is < 1 degree for all tilt angles < 45 degrees, which is well within the range of angles used.

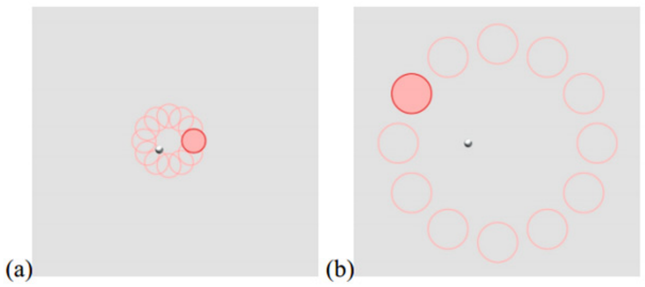

Figure 3: The task performed by participants. (a) *A* = 131 pixels, *W* = 63 pixels. (b) *A* = 526 pixels, *W* = 105 pixels.

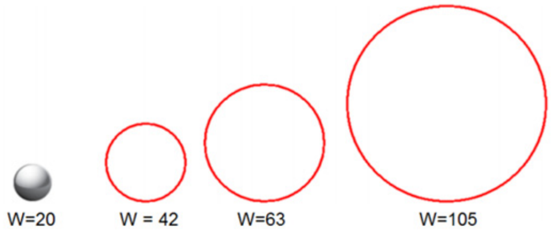

Figure 4: The user-controlled ball (cursor) relative to the three target sizes used in the experiment.

4.3 Procedure

After giving informed consent, the procedure and software were demonstrated to participants. The study was performed seated, and participants held tablet however they felt comfortable. See Figure 5. At the start of the experiment, participants were given 12 practice trials with each of the extreme selection delays (i.e., 0 ms and 500 ms) in the starting control mode. At the halfway point the control mode changed, and participants were given the same number of practice trials with the same selection delays.

Participants completed a questionnaire after the experiment. This included questions on control smoothness, mental and physical effort, and general preference of selection delay.

4.4 Design

The experiment used a repeated-measures design with the following independent variables and levels:

Control mode:	Position, Velocity
Selection delay:	0, 300, 400, and 500 ms
Target amplitude:	131, 263, 526 pixels
Target width:	42, 63, 105 pixels

Half of the participants started with position-control, and the other half started with velocity-control. Control mode switched at the halfway point. Selection delay was counterbalanced within control mode according to a balanced Latin square.

The dependent variables were movement time, target re-entries (for non-zero selection delays), movement variability, maximum tilt angle, and throughput. Results of the post-experiment questionnaire are also presented.

Overall, the experiment took about one hour per participant. Each participant completed 2 control modes × 4 selection delays × 3 amplitudes × 3 widths × 12 selections = 864 trials total, or 13,824 trials over all 16 participants.

5 Results

Statistical reports for movement time (*MT*), throughput (*TP*), target re-entries (*TRE*), movement variability (*MV*), movement error (*ME*), and maximum tilt (*MaxTilt*) are shown in Table 2.

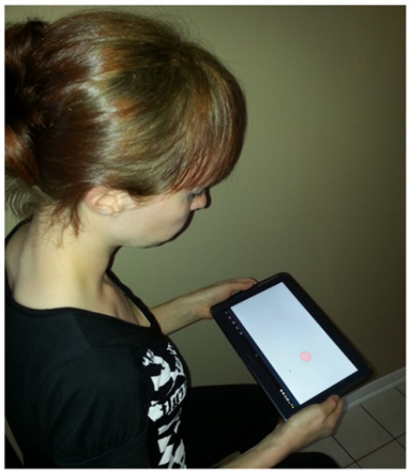

Figure 5: Participant performing the selection task.

5.1 Movement Time

The grand mean for movement time was 2325 ms. There was a significant main effect on movement time for both control mode and selection delay. See Table 2. Overall, movement times were 1555 ms for position-control and 3095 ms for velocity-control; i.e., movement time was about 50% lower for position-control than for velocity-control. A Tukey-Kramer post hoc test indicated that 0 ms selection delay was significantly faster than all others, while the 300 and 400 ms delays were significantly faster than the 500 ms delay. See Figure 6.

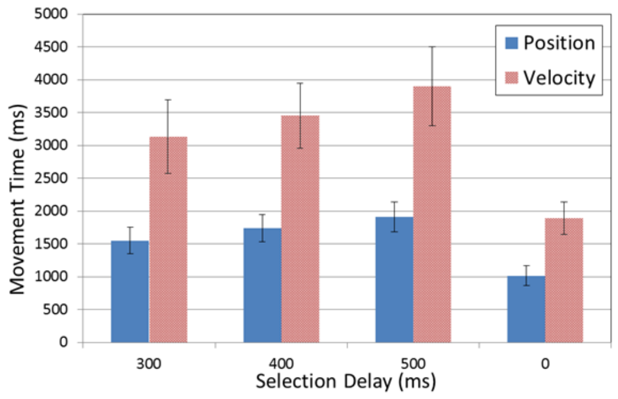

Figure 6: Movement time by control mode and selection delay. Error bars show ±1 *SE*.

There was also a significant interaction effect between control mode and selection delay on movement time. The difference in movement time with the 0 ms delay was more substantial for velocity-control than for position-control.

5.2 Throughput

Throughput was calculated according to Equation 2. As mentioned earlier, effective width and amplitude were unavailable since the task precluded missing the targets.

The main effects and interaction effect for control mode and selection delay were significant. See Table 2. Throughput scores are shown in Figure 7. Position-control with 0 ms selection delay had the highest throughput, at 3.3 bps. For comparison, most ISO-conforming pointing studies on desktop systems report mouse throughput of around 4 to 5 bps [25, Table 4]. Thus, throughput for position-controlled tilt input is about 25% lower than for mouse input. Conversely, velocity-control had an overall

Effect		*MT*		*TP*		*TRE*		*ME*		*MV*		*MaxTilt*	
Name	*df*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*	*F*	*p*
(C)ontrol Mode	1,15	230.7	*	1115.2	*	30.9	*	53.9	*	23.7	*	1409.1	*
(S)election Delay	3,15	78.3	*	207.2	*	25.2	*	84.5	*	36.7	*	35.9	*
C × S	3,45	24.7	*	55.3	*	3.1	.06	23.3	*	15.2	*	48.3	*

Table 2: Statistical effects for accuracy measures. Control mode and selection delay are main effects, while C × S is the interaction. Significant effects are indicated with * for $p < .0001$.

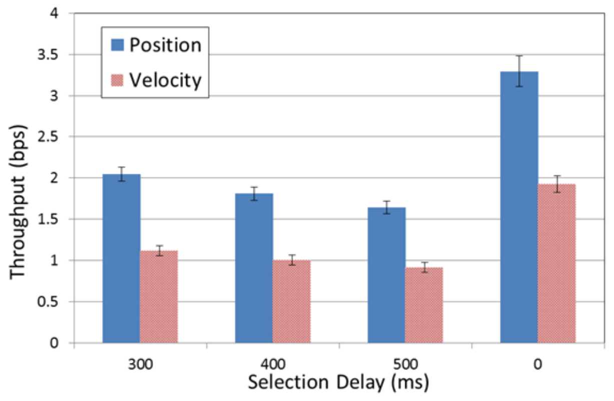

Figure 7: Throughput by control mode and selection delay. Higher is better. Error bars show ±1 *SE*.

throughput of 1.2 bps which was significantly lower than position-control (and about 70% lower than mouse throughput).

Unsurprisingly, throughput is lower for higher selection delays. The worst condition was velocity-control with 500 ms selection delay. Even the 300 ms delay was substantially worse than the 0 ms delay. This suggests that alternative target disambiguation techniques may fare better than the delayed timeout used. These delays may also be too short for UIs with distracter targets.

5.3 Accuracy

Since all trials ended with selection of the target, error rates were unavailable for the study. Instead, we examined other accuracy measures (target re-entries, movement variability, and movement error) which correlate with pointing performance [16]. Statistical data for these metrics are reported in Table 2.

5.3.1 Target Re-Entries (TRE)

Target re-entries is the count (averaged per 12 selections) of how frequently the cursor left and then re-entered the target. In a "perfect" selection task, *TRE* is 0, i.e., there is one target entry. *TRE* gives an indication of control problems, and may be more pronounced for smaller targets. See Figure 8 for *TRE* scores.

Aside from the results shown in Table 2, there were significant interaction effects between control mode and target size ($F_{2,30} = 13.1$, $p < .0001$) and between selection delay and target size ($F_{4,60} = 18.0$, $p < .0001$) (not given in Table 2). Evidently, participants had a much harder time selecting the smallest targets with velocity-control than with position-control, and also with longer selection delays. The worst overall score resulted from the combination of velocity-control, a 500 ms selection delay, and a 42-pixel target. *TREs* for this condition exceeded 20 per sequence of 12 selections.

5.3.2 Movement Variability and Movement Error

In a perfect selection task, the cursor would move in a straight line to the target, irrespective of the actual distance along the task axis (the line between subsequent targets). Movement variability (*MV*)

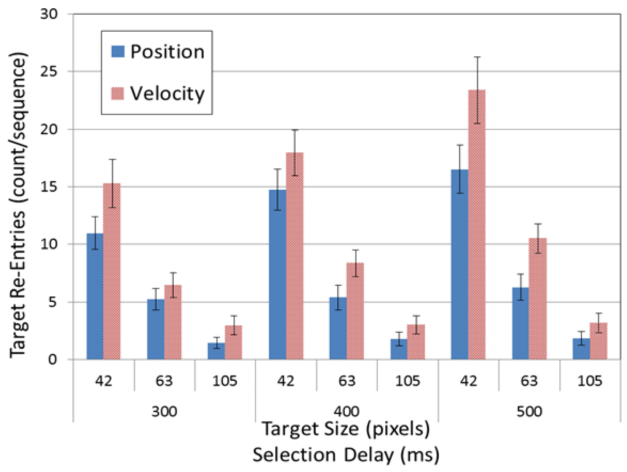

Figure 8: Target re-entries by control mode, selection delay, and target size. Lower is better. Error bars show ±1 *SE*.

represents this path straightness; the lower the score, the straighter the path. Movement error (*ME*) is the average distance of the motion path from the task axis, and may reflect difficulty in maintaining an optimal path. Control mode and selection delay had significant main effects on both *MV* and *ME*. The interaction effects were also significant. See Table 2.

Both movement variability and error were lower with position-control than velocity-control. Interestingly, velocity-control with 0 ms delay had the highest scores for both movement variability and error. While this suggests participants had more difficulty moving the ball in a straight line in this condition, reckless tilting of the tablet may be a better explanation. This is discussed further in the motion analysis section below. This path inefficiency is reflected in the overall higher movement times and lower throughput scores reported above. Conversely, position-control with 0 ms selection delay was not significantly worse than with other selection delays. See Figures 9 and 10.

Overall, position-control offered superior handling. Participants had difficulty keeping the ball close to the task axis with velocity-control, especially with the 0 ms delay. Motion path analysis (see below and Figure 13) corroborates this result.

5.3.3 Maximum Tilt

The degree to which participants tilted the tablet was also recorded. Note that position-control *required* a certain amount of tilt, as determined by the target distance (relative to the center of the tablet) and the tilt gain. With velocity-control, however, the amount of tilt depends on the user's *strategy* – greater tilt yields higher ball velocity. The main effects and interaction effect of control mode and selection delay on maximum tilt were both statistically significant. See Table 2.

Tilt angles were significantly and substantially lower for velocity-control than for position-control. See Figure 11. Velocity-control with 0 ms selection delay exhibited significantly larger tilt angles than all other delays for that control mode.

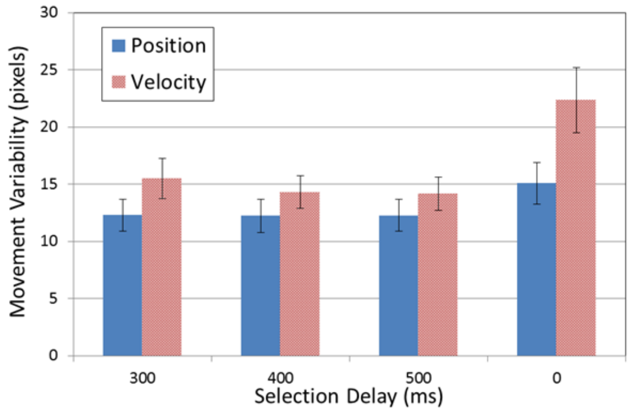

Figure 9: Movement variability by control mode and selection delay. Error bars show ±1 *SE*.

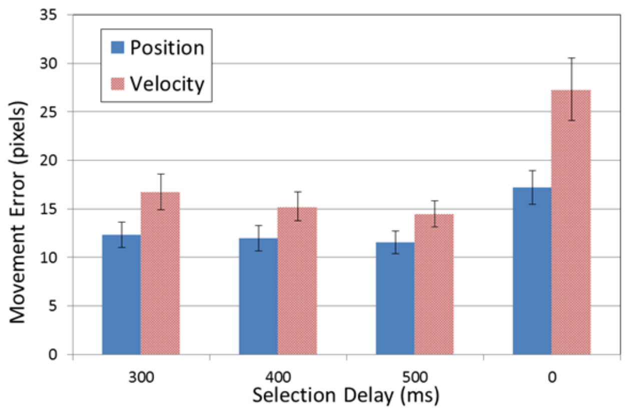

Figure 10: Movement error by control mode and selection delay. Error bars show ±1 SE.

Position-control required approximately twice as much tilt as velocity-control. The maximum tilt for position-control across selection delays was essentially flat. As noted above, maximum tilt with position-control is determined more by target location than by user strategy. Despite requiring greater degrees of tilt, position-control also offered higher performance. This suggests that the degree to which participants tilted the device was not directly linked to performance. This is likely linked to the tilt gain setting.

5.3.4 Motion Path

Figure 13 depicts typical motion paths of the ball in four conditions (0 ms and 500 ms selection delays for both control modes). Red dots indicate where selections occurred, and the blue line indicates the cursor/ball motion path. Velocity-control yielded erratic motion, with large target overshoots and corrections. This is supported by higher *MV* and *ME* scores and overall worse performance. Conversely, motion is fairly accurate with position-control; this likely explains the higher overall performance.

Line thickness in Figure 13 increases with the amount of tilt (i.e., thick lines indicate more tilt than thin lines). With position-control (Figure 13c and d), lines are thicker farther away from the center, as greater tilt is required to position the ball farther from the tablet center. Conversely, for velocity-control the lines are thinner near targets. Participants were more careful upon approaching the target, and reduced the tablet tilt to slow the ball movement, improving control for the feedback-guided selection. Conversely, when moving the ball between targets, they would tilt the tablet to greater degrees to traverse the distance more quickly.

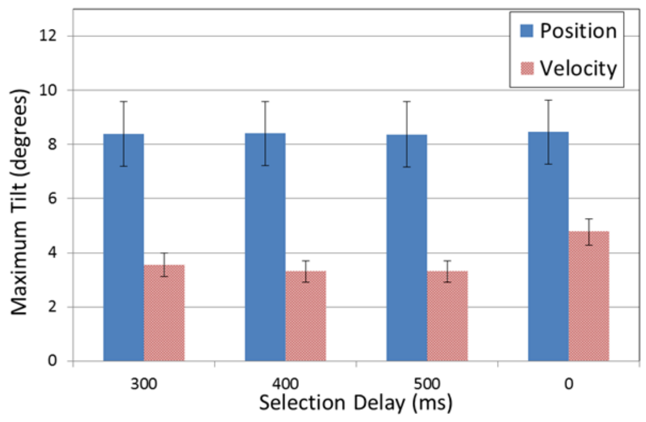

Figure 11: Maximum tilt angle by control mode and selection delay. Error bars show ±1 *SE*.

Figure 13b (velocity-control with 0 ms selection delay) shows the most erratic motions, and are typical of the condition. Participants would recklessly tilt the device to quickly traverse the distance between targets. This behaviour is especially evident in the line thickness near the center of Figure 13b. The movement appears somewhat ballistic and frequently resulted in the ball moving *through* the target, followed by a correction to adjust the ball's direction for the next target.

5.3.5 Subjective Questionnaire Results

Participants also completed a questionnaire, the results of which were compared using *t*-tests. Position-control scored significantly higher in all questions, see Figure 12. These results indicate that position-control was also preferred by the study participants.

Participants also provided comments on the experimental conditions. One participant reported that their lower arms were more tense while using the velocity mode and that the position mode was more relaxing. Another noted, "If I had to use velocity-control for any real task, I would stop using that app". These comments help explain the difference in participant preference for the two control modes, while underlining the measured performance differences.

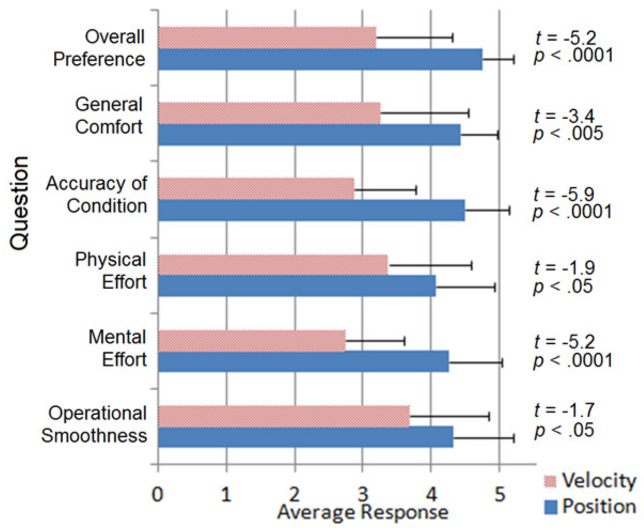

Figure 12: Survey responses and statistical reports by question. Higher scores are more favorable. Error bars show ±1 *SD*.

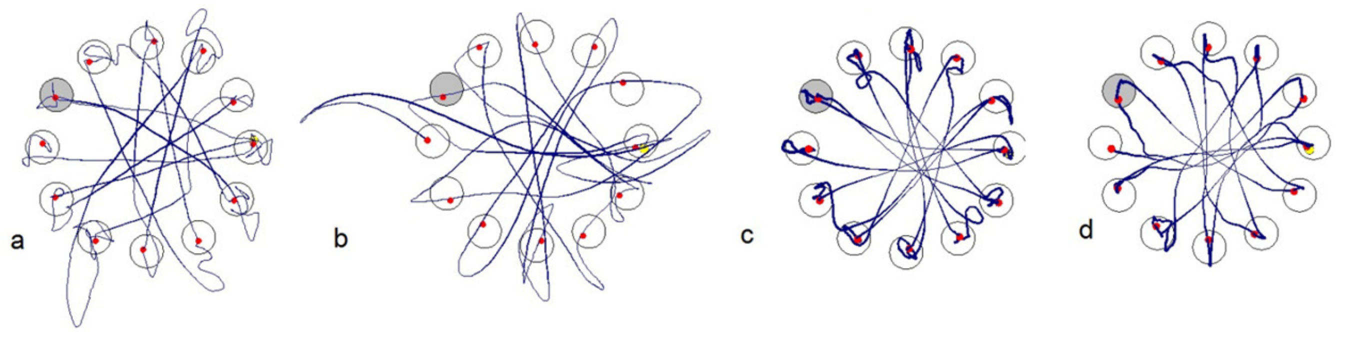

Figure 13: Motion paths for (a) velocity-control with 500 ms delay, (b) velocity-control with 0 ms delay, (c) position-control with 500 ms delay, and (d) position-control with 0 ms delay.

The survey also asked participants for their selection delay preference. Eleven preferred no selection delay, four preferred a 300 ms delay, and one preferred a 400 ms delay. It is likely that the strong preference for instantaneous selection is simply due to the higher performance it offers. Of course, this selection mode is not always feasible (e.g., in the presence of distracter targets)..

6 Discussion and Limitations

Our results suggest that, in applications where there is a choice, position-control tilt input is preferred. Overall, participants performed better with and generally preferred position-control over velocity-control. This may be due to the higher attention demand in the velocity-control mode, where constant visual attention is required to adjust the ball's motion. Conversely, in position-control, attention demand is somewhat lower as participants can leverage proprioception without worrying that the ball will venture off-course. This is reflected in the analyses of motion, movement variability, and error. Attention demand differences are also reflected in participants' assessment of higher mental effort with velocity-control.

It is worth noting that position-control is unsuited to situations with unbounded scrolling, for example, map or document navigation. Position-control is thus dependent on the control range. This can be mitigated by increasing the tilt gain or through clutching. Of course, after a point the screen simply tilts out of a viewable range. Velocity-control does not have this limitation.

Participants also tended to prefer shorter selection delays. Of course, this result is contingent with the task. Typical UIs include multiple targets yielding the possibility of selecting the wrong target. Our study did not include distracter targets. While performance was highest with a 0 ms delay, such a technique is clearly impractical for real-world use. Similarly, the some selection delays (e.g., 300 ms) may be too short to avoid accidental selections. We plan to study this further in an experiment that includes distracter targets.

Finally, we also acknowledge that touch-control is the predominant control style on mobile devices. Tilt control is typically used only in limited situations, or in conjunction with touch. As we did not directly compare these two control styles, we can only speculate on performance differences in selection tasks. It is likely that touch control would offer superior performance to tilt control for the most common interactions in mobile user interfaces. That said, we believe that the present investigation indicates that position-controlled tilt is worth further consideration. It would be interesting to implement position-control for games which commonly use velocity-control and compare scores between these. We are also considering other possible uses of position-control tilt, including phone dialers and text entry. These are topics for future study, however.

7 Conclusions

We presented a study investigating important aspects of tilt-based interaction. This study represents (to our knowledge) the first comparison between position- and velocity-control for general tilt-based interaction. Results indicate that although it required greater amounts of tilt, position-control offered significantly better performance than velocity-control and was also strongly preferred by participants. Not only was this mode faster and yielded higher throughput, it also offered much smoother control as indicated by the analysis of several motion metrics.

The results also indicate that instant selection upon entering a target (i.e., 0 ms selection delay) afforded better performance than a delayed selection. While this is not surprising, such a selection mode is not always practical. For example, some means of target disambiguation is required if the cursor must cross other targets to reach the intended target. While the selection delays studied here suggest faster delays offer better performance, we note again that these results must be considered in light of the fact that no actual distracter targets were present in our study.

7.1 Future Work

The study used nominal gain settings chosen from pilot testing. This is one obvious parameter to evaluate further in future work. In particular, and as suggested by participants, a dynamic gain level is worth considering.

A long term objective is to further investigate what kinds of tasks benefit more from velocity tilt control versus position tilt control. For example, one might consider a marble-maze style game using position-control, and compare this to velocity-control.

Acknowledgements

Thanks to the anonymous reviewers for their helpful suggestions on improving the paper. This work was supported by NSERC.

References

[1] J. F. Bartlett, Rock'n'Scroll is here to stay, *IEEE Computer Graphics and Applications*, *20*, 2000, 40-45.

[2] K. Browne and C. Anand, An empirical evaluation of user interfaces for a mobile video game, *Journal of Entertainment Computing*, *3*, 2012, 1-10.

[3] P. Eslambolchilar and R. Murray-Smith, Tilt-based automatic zooming and scaling in mobile devices – a state-space implementation, *Proceedings of Human Computer Interaction with Mobile Devices and Services - MobileHCI 2004*, (Berlin: Springer, 2004), 120-131.

[4] P. M. Fitts, The information capacity of the human motor system in controlling the amplitude of movement, *Journal of Experimental Psychology*, *47*, 1954, 381-391.

[5] P. Gilbertson, P. Coulton, F. Chehimi, and T. Vajk, Using “tilt” as an interface to control “no-button” 3-D mobile games, *Computers in Entertainment (CIE)*, *6*, 2008, 38.

[6] B. L. Harrison, K. P. Fishkin, A. Gujar, C. Mochon, and R. Want, Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI '98*, (New York: ACM, 1998), 17-24.

[7] A. Henrysson, M. Billinghurst, and M. Ollila, Virtual object manipulation using a mobile phone, *Proceedings of the International Conference on Augmented Tele-Existence - ICAT 2005*, (New York: ACM, 2005), 164-171.

[8] A. Henrysson, J. Marshall, and M. Billinghurst, Experiments in 3D interaction for mobile phone AR, *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia - GRAPHITE 2007*, (New York: ACM, 2007), 187 - 194.

[9] K. Hinckley, J. Pierce, M. Sinclair, and E. Horvitz, Sensing techniques for mobile interaction, *Proceedings of the ACM Symposium on User Interface Software and Technology - UIST 2000*, (New York: ACM, 2000), 91-100.

[10] W. Hürst and C. van Wezel, Gesture-based interaction via finger tracking for mobile augmented reality, *Multimedia Tools and Applications*, *62*, 2013, 233-258.

[11] T. Hynninen, First-person shooter controls on touchscreen devices: A heuristic evaluation of three games on the iPod touch, *M.Sc. Thesis,Department of Computer Sciences, University of Tampere, Tampere, Finland*, 2012, 64 pages.

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

[13] E. Jones, J. Alexander, A. Andreou, P. Irani, and S. Subramanian, GesText: Accelerometer-based gestural text-entry systems, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI 2010*, (New York: ACM, 2010), 2173-2182.

[14] W. S. Kim, F. Tendick, S. R. Ellis, and L. W. Stark, A comparison of position and rate control for telemanipulations with consideration of manipulator system dynamics, *IEEE Journal of Robotics and Automation*, *3*, 1987, 426-436.

[15] I. S. MacKenzie and P. Isokoski, Fitts' throughput and the speed-accuracy tradeoff, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI 2008*, (New York: ACM, 2008), 1633-1636.

[16] I. S. MacKenzie, T. Kauppinen, and M. Silfverberg, Accuracy measures for evaluating computer pointing devices, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI 2001*, (ACM, 2001), 9 - 16.

[17] I. S. MacKenzie and R. J. Teather, FittsTilt: The application of Fitts' law to tilt-based interaction, *Proceedings of the 7th Nordic Conference on Human-Computer Interaction - NordiCHI 2012*, (New York: ACM, 2012), 568-577.

[18] S. Medryk and I. S. MacKenzie, A comparison of accelerometer and touch-based input for mobile gaming, *International Conference on Multimedia and Human-Computer Interaction - MHCI 2013*, (Ottawa, Canada: International ASET, 2013), 117.1-117.8.

[19] I. Oakley and S. O'Modhrain, Tilt to scroll: Evaluating a motion based vibrotactile mobile interface, *Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, (New York: IEEE, 2005), 40-49.

[20] K. Partridge, S. Chatterjee, V. Sazawal, G. Borriello, and R. Want, TiltType: Accelerometer-supported text entry for very small devices, *Proceedings of the ACM Symposium on User Interface Software and Technology - UIST 2002*, (New York: ACM, 2002), 201-204.

[21] J. Rekimoto, Tilting operations for small screen interfaces, *Proceedings of the ACM Symposium on User Interface Software and Technology - UIST '96*, (New York: ACM, 1996), 167-168.

[22] H. H. Sad and F. Poirier, Evaluation and modeling of user performance for pointing and scrolling tasks on handheld devices using tilt sensor, *Advances in Computer-Human Interactions - ACHI 2009*. (New York: IEEE, 2009), 295-300.

[23] V. Sazawal, R. Want, and G. Borriello, The Unigesture approach: One-handed text entry for small devices, *Proceedings of Human Computer Interaction With Mobile Devices - MobileHCI 2002*, (Berlin: Springer, 2002), 256-270.

[24] D. Small and H. Ishii, Design of spatially aware graspable displays, *Extended Abstracts of the ACM SIGCHI Conference on Human Factors in Computing Systems - CHI '97*, (New York: ACM, 1997), 367-368.

[25] R. W. Soukoreff and I. S. MacKenzie, Towards a standard for pointing device evaluation: Perspectives on 27 years of Fitts' law research in HCI, *International Journal of Human-Computer Studies*, *61*, 2004, 751-789.

[26] L. Valente, C. Sieckenius de Souza, and B. Feijo, Turn off the graphics: Designing non-visual interfaces for mobile phone games, *Journal of the Brazilian Computer Society*, *15*, 2009, 45-58.

[27] J. Wang, S. Zhai, and J. Canny, Camera phone based motion sensing: Interaction techniques, applications and performance study, *Proceedings of the ACM Symposium on User Interface Software and Technology - UIST 2006*, (New York: ACM, 2006), 101-110.

[28] D. Wigdor and R. Balakrishnan, TiltText: Using tilt for text input to mobile phones, *Proceedings of the ACM Symposium on User Interface Software and Technology - UIST 2003*, (New York: ACM, 2003), 81-90.

[29] L. Zaman and I. S. MacKenzie, Evaluation of nano-stick, foam buttons, and other input methods for gameplay on touchscreen phones, *International Conference on Multimedia and Human-Computer Interaction - MHCI 2013*, (Ottawa, Canada: International ASET, 2013), 69.1-69.8.

[30] L. Zaman, D. Natapov, and R. J. Teather, Touchscreens vs. traditional controllers in handheld gaming, *Proceedings of the International Academic Conference on the Future of Game Design and Technology - FuturePlay 2010*, (New York: ACM, 2010), 183-190.