# Comparing Order of Control for Tilt and Touch Games

Robert J. Teather
Dept. of Computing & Software
McMaster University
Hamilton, ON, Canada
teather@mcmaster.ca

I. Scott MacKenzie
Dept. of Electrical Engineering and Computer Science
York University
Toronto, ON, Canada
mack@cse.yorku.ca

## ABSTRACT
We conducted a study comparing two touch-based and two tilt-based game control methods using a Pong-like game over two one-hour sessions. Each input method was compared by order of control: position-control and velocity-control. Participants' performance was assessed for game-level reached and how frequently the ball was missed. Results indicate that order of control is a greater determinant of performance than input method. For both position-control modes (tilt and touch), participants reached game-levels roughly twice as high as with the velocity-control modes. Miss rates were about 40% higher with the velocity-control modes than with position-control.

## Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation, (e.g., HCI)**]: User Interfaces – *Input devices and strategies (e.g., mouse, touchscreen)*. K.8.0 [**Personal Computing**]: General – *Games*.

## General Terms
Performance, Human Factors.

## Keywords
Tilt and touch control; position-control; velocity-control; games.

## 1. INTRODUCTION
Modern mobile devices offer a variety of sensors and a rich set of interaction schemes. These usually include a touchscreen, cameras (often two), accelerometers, gyros, and magnetometers for tilt-control. Usually the touchscreen is the primary means of interacting with the device. This variety of control options has yielded new types of mobile games, enabling players to directly touch game elements. *Indirect* touch input is also sometimes used, e.g., for virtual joysticks and buttons (see Figure 1 for examples). This is similar to how soft keyboards are used for text entry on mobile devices. Virtual controls are regularly employed in mobile ports of console games, but are sometimes used in mobile-exclusive games too.

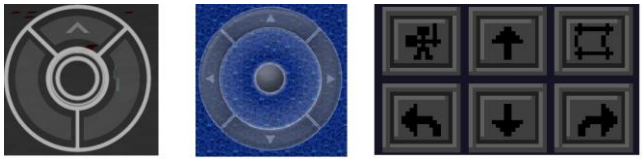

IE2014, December 02 - 03 2014, Newcastle, NSW, Australia

ACM 978-1-4503-2790-9/14/12…$15.00
http://dx.doi.org/10.1145/2677758.2677766

Tilt-control is also common, especially in games where tilting affords a natural interaction style [4]. Some games offer a choice of input method (tilt vs. touch). The question of which is more efficient is important to game developers attempting to enhance the game UI. To address this question, we compared tilt and touch input methods in a custom-developed Pong-like game where the paddle is controlled by device tilt or an on-screen virtual touch control. The pros and cons of using custom-developed games as experimental platforms (rather than commercial games, as in previous research) are detailed in Section 3.

Like physical joysticks, virtual joysticks (similar to the touch control used in our study) can operate in either a position mode or a velocity mode. These modes are examples of *order of control* and are referred to as *position-control* and *velocity-control*. Velocity-control is commonly associated with isometric joysticks, such as those found on game console controllers or between the G, H, and B keys on some laptop keyboards. Such joysticks have also been proposed for handheld terminals [18]. In contrast, isotonic joysticks are typically position-control devices; moving the joystick specifies a unique position. Input devices like the mouse also employ position-control – moving the device affects the position of the cursor.

Both position-control and velocity-control are reasonable control options for both tilt and touch input; all combinations have been employed in mobile games. See Table 1 for several recent examples. It is unclear, however, which of these two mappings is more appropriate for each of these input methods. We thus explore the design space of game input options on touch-based devices. Previous work has explored order of control with tilt for point-select tasks [19]; other work has compared tilt and touch input in games [2, 6, 14]. No previous work has investigated order of control and input method together. Hence, in addition to comparing tilt and touch control, our study also includes order of control. The goal is to determine which factor, order of control or input method, more strongly affects performance in games.



(a) (b) (c)

**Figure 1. Sample virtual (indirect touch) controls from (a) Revolab's *Undead Pixels*, (b) Square Enix's *Dragon Quest II*, and (c) Eric Kinkhead's *Quest Lord*. Such controls provide software simulations of "traditional" game console controllers or keyboards, but on touchscreens.**

**Table 1. Example mobile games and their control styles. Rank indicates Google Play Store ranking (and type) at time of writing. Games with a ranking of "N/A" do not appear in a top-500 list.**

| Game, Developer | Input Method | Order of Control | Scrolling | Rank | Game Style |
|---|---|---|---|---|---|
| *Angry Birds*, Rovio | Direct touch | Position | Yes | 21 (free) | Physics simulation: drag to launch birds |
| *Bit Trip Beat*, Gaijin | Direct touch, tilt | Position | No | N/A | Pong-like game: move paddle to intercept objects |
| *Grand Theft Auto* series, Rockstar | Indirect touch | Velocity | Yes | 19 (pay) | 3D open world game using virtual controls |
| *Hungry Shark*, FGOL | Indirect touch, tilt | Velocity | Yes | 109 (free) | Action: controls move shark through 2D world |
| *Marble Maze*, Hyperkani | Tilt | Velocity | No | N/A | Physics simulation: tilt device to rolls ball |
| *Minecraft*, Mojang | Indirect touch | Velocity | Yes | 1 (pay) | 3D open world game using virtual controls |
| *Need for Speed: Most Wanted*, Electronic Arts | Indirect touch, tilt | Position | Yes | 12 (pay) | Racing: tilt simulates steering wheel, position mapping to car wheel orientation |
| *Plants vs. Zombies*, Popcap Games | Direct touch | Position | No | 4 (pay) | Tower defense, place plants to obstruct zombies |

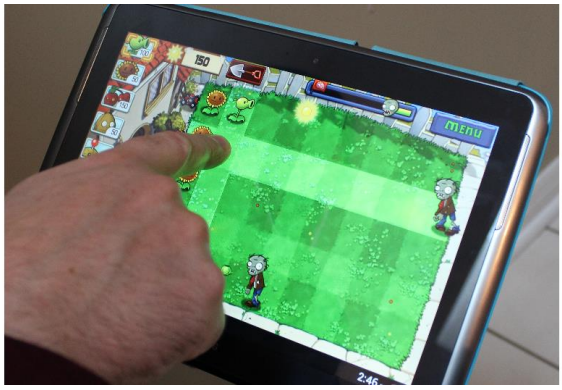



**Figure 2. Popcap's *Plants vs. Zombies* employs direct touch input. Players touch the plant icons on the left side of the display, then touch the location in the game to place a plant. This is a form of position-control touch input.**

**Figure 3. Rockstar's *Grand Theft Auto 3*, like many console game ports, uses virtual controls: software recreations of gamepads, but with game-specific buttons rather than the general purpose buttons found on physical gamepads.**

## 1.1 Considerations in Game Control

To better motivate this work, we first discuss predominant game control schemes in *actual* commercial games. We consider only commercial games here (rather than academic work) to present an overview of the state of the art in practical game control design. See Table 1 for a summary of these games.

### *1.1.1 Direct Touch Input*

Touch input is often used for direct interaction with game elements. See, for example, Popcap's *Plants vs. Zombies* (Figure 2), or Rovio's *Angry Birds*. The player directly touches characters/objects in the game – for example, touching positions to set a plant in *Plants vs. Zombies* (Figure 2). Similarly, players touch, drag, and release birds in *Angry Birds*, to launch them from a slingshot. Often, this form of touch input works similarly to a mouse. Both of the aforementioned games are available on desktop systems, and both effectively substitute the finger for the mouse pointer. Since this input method uses a direct mapping of the touched point to an in-game position, it is a form of position-control input.

### *1.1.2 Indirect Touch Input*

Indirect touch input employs virtual controls. Sample virtual controls are shown in Figure 1. This control style is often used in mobile games that were originally developed for game consoles. Console games are developed with specific (physical) gamepads in mind. Gamepads are designed to be generic enough to support many types of games and thus offer several general-purpose buttons, two analog sticks, and/or directional buttons. When porting console games to mobile platforms, it is easier to add virtual controls replicating a physical gamepad than to develop a completely new touchscreen-based control scheme. Consequently, virtual controls may necessitate fewer modifications to the underlying game code.

While indirect touch input is less common than direct touch input, it is still used in numerous popular games. Example games employing indirect touch include Mojang's *Minecraft*, and Rockstar Games' *Grand Theft Auto* series (Figure 3). Console emulators also provide a virtual version of the emulated console's gamepad. This is necessary as it is impossible to modify the game

code played by the emulator to enable direct touch input. Notably, at the time of writing, Neutron Emulation's *SuperGNES* – an emulator for Nintendo's *Super NES* game console – is ranked as the #5 most popular pay game on the Android Play Store.

### *1.1.3 Tilt Input*

Controlling games with physical motions was first popularized by the Nintendo *Wii* game console. Mobile devices quickly caught up with the release of Apple's *iPhone* in 2007. Today, many games employ tilt input. There are games where tilt input is more natural than touch input, such as *Marble Maze.* A typical marble maze game is depicted in Figure 4. Driving games, such as EA's *Need for Speed: Most Wanted*, also provide tilt input, where the device is used like a steering wheel to control the car's direction.

Although less common, some action games such as *Hungry Shark* or *Grabatron* (both from Future Games of London) offer both tilt and touch input options. In these games, tilt input uses a velocity-control mapping: tilting the device increases movement speed in the tilt direction, while simultaneously scrolling the view in the same direction. In contrast, Gaijin Games' *Bit Trip Beat* employs position-control tilt input – tilting the device controls the position of the paddle, rather than its movement speed.

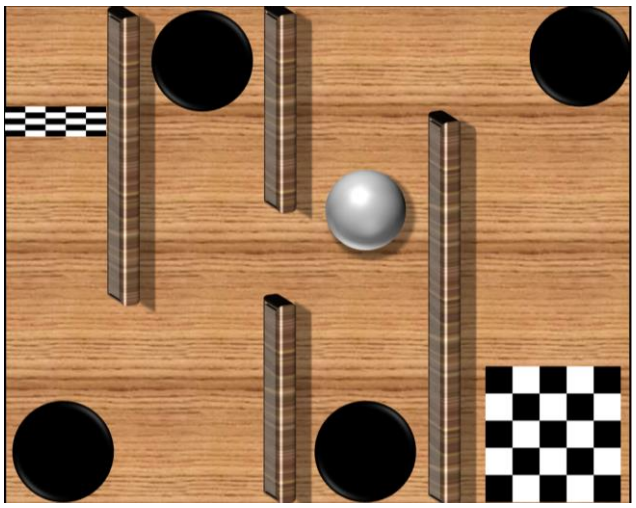


**Figure 4. Marble maze games employ tilt input and physical simulation to roll a ball through a maze. These simulate physical wooden ball puzzles where the objective is to roll the ball from one point (the checkered box) to another (the checkered strip) while avoiding obstacles (walls and holes).**

### *1.1.4 Scrolling vs. Non-Scrolling Games*

A final issue is view scrolling. This design choice is used in many modern games and (indirectly) influences the choice of input technique. In scrolling games, as the player moves their character through the environment, the "camera" moves with them. In contrast, non-scrolling games are those that do not move the viewpoint during gameplay. In other words, the scene is constrained to a single "screen" at any given time. The scene may change between levels though. *Bit Trip Beat*, *Plants vs. Zombies*, and *Marble Maze* are all examples of non-scrolling games.

In non-scrolling games, both position- and velocity-control mappings are feasible with both touch and tilt input. *Bit Trip Beat* uses a position-control mapping for both tilt and touch input and is the closest example to the game used in our experiment. *Marble Maze* uses a velocity-control mapping for tilt-input – a position-control tilt input mapping would also be feasible (if physically unnatural).

Scrolling games tend to constrain the choice of input method. For example, tilt-based position-control is impractical for scrolling games. One cannot uniquely specify an off-screen point with device tilt, as the screen typically tilts out of view before the intended point becomes visible. Direct touch is still feasible, however; game elements can be touched as they come onto the screen. Tilt-based velocity-control is also feasible in scrolling games. For example, *Hungry Shark* employs such an input method. Indirect touch input is similarly possible.

## 1.2 Contributions

We present what is (to our knowledge) the first empirical comparison of touch and tilt input across position- and velocity-control. The four control schemes investigated are thus:

tilt + position-control
tilt + velocity-control
touch + position-control
touch + velocity-control

The comparison of tilt vs. touch across order of control for game input is a novel aspect of our work. Previous work has compared touch and tilt [2, 6, 14] in games, and other work has investigated order of control for tilt input [19]. To date, this is the first study looking at both of these options in tandem for game input. Considering the variety in game control options (outlined in Section 1.1), the evaluation of these factors together makes sense.

Our goal is to assess which option offers the best performance for *non-scrolling* games using a Pong-like game where players move a paddle to intercept a bouncing ball. Position-control is standard for this style of game, but what is the potential for velocity-control? What are the performance trade-offs for touch vs. tilt input? These are open questions. We are also interested in user experience: What do users feel is the most enjoyable and engaging control scheme? Finally, we provide design guidelines for game developers considering which input techniques to use.

# 2. RELATED WORK

Previous work comparing velocity- and position-control indicates that matching the property sensed (i.e., position or displacement) to the property controlled (i.e., position, or velocity) produces the most effective mappings [8, 16, 23]. Oakley et al. [16] report that position-control offered faster tilt-based menu navigation. Similarly, Teather and MacKenzie [19] report that position-control offers higher performance in tilt-based point-select tasks. These results may not generalize to games though, which often use more complex tasks than pointing or menu navigation.

## 2.1 Comparing Touch and Tilt Input

There is relatively little research comparing touch and tilt input for games. Previous work yielded conflicting results and no strong "take-away" message. Thus, we also consider work comparing these input methods in non-game contexts.

The popularity of touch input may be in part due to finger dexterity. Tilt input mainly uses less dexterous joints such as the wrists [1, 24]. Nevertheless, researchers report that tilt input allows for faster text entry than multi-tap [20] and faster 3D rotations than touch input [5].

There are a few examples of studies directly comparing touch and tilt game input [2, 6, 14]. Using a shooter game played on an *iPod*

*Touch,* Browne and Anand [2] compared virtual control touch input to tilt input. Participants both preferred and performed better with tilt input, and survived longer in the game. However, this is likely because the interaction did not support multi-touch, i.e., participants could not move and shoot simultaneously. Conversely, tilt input allowed multiple actions at once. Hence their results are likely biased in favour of tilt input, and a better touch input implementation may yield different results.

Medryk and MacKenzie [14] compared tilt and touch input using *Bit Trip Beat* by Gaijin Games. They measured participant game scores, hit accuracy, and level completion time, and report that tilt input offered worse performance than touch input. Our current work improves upon their study in several ways. These largely stem from the fact that the authors used a commercial game rather than developing a custom game (see Section 3 for a detailed discussion of this tradeoff). The game offered limited flexibility in exploring the design space of touch and tilt control options – essentially, the authors could only compare control schemes implemented by the game's developers. The game's touch input used a direct vertical mapping of the touch point to the paddle position. The game did not include virtual controls such as those used by Browne and Anand [2] or in our study. Unlike *Bit Trip Beat*, our study employs position-control mappings with a virtual control, which we refer to as a "touch strip".

Hynninen [6] conducted a study using three first-person shooter games on an *iPod Touch*. One of these games included Activision's *Call of Duty: World at War: Zombies* which offered both tilt and touch input (using virtual controls). Hynninen reports that tilt input was inferior to the virtual joysticks used in this game. However, this research has low internal validity, since like most commercial games implementation details are unavailable.

## 2.2 Virtual/Soft Controls

Although similar to physical controls, virtual controls often perform much worse in practice. This is most likely due to the absence of tactile feedback [21, 22].

Chu and Wong [3] report that players almost unanimously preferred physical controls over virtual controls. Tilt input, on the other hand, may leverage proprioception which can help compensate for missing tactile feedback [15]. Similarly, Lee and Zhai [10] report that audio and visual feedback help compensate for the absence of tactile feedback. Their study [10] on text entry indicates that soft keys are efficient and only perform marginally (and not significantly) worse than physical keys. This seems largely dependent on the feedback mechanisms used.

Other researchers report that rather than attempting to emulate physical gamepads, alternative UI arrangements can offer better touch input performance. Oshita and Ishikawa [17] developed a touch input UI for game-play. Instead of simulating the physical gamepad, numerous virtual buttons were used. Each virtual button operated like a macro issuing sequences of gamepad button presses. They found that while gamepad entry speed was faster, their UI was less error-prone. However, their input method used numerous virtual buttons, and thus occludes a large portion of the screen. Tilt input does not present this problem.

# 3. COMMERCIAL VS. CUSTOM GAMES IN HCI RESEARCH

An important issue in evaluating game UIs is deciding which game to use [7, 13]. Broadly speaking, the choice is between a professionally-developed commercial game and a custom game. We examine this choice in detail, as it is not made lightly.

The key advantage in using commercial games is high external validity: Results generalize better to real-world situations. This is unsurprising since commercial products are real games. However, conducting experimental research using commercial games generally suffers from low internal validity: Observations are not reliably attributable to the test conditions. There are two reasons.

First, source code for commercial games is proprietary and unavailable to researchers. Without the ability to inject code to manage an experiment, measurement and data collection are compromised. This occurs since logging user actions, timestamps, etc., is relegated to an external process [6, 12-14, 22]. In some cases, game scores [12, 14] or level completion time [12] can be externally viewed and logged as dependent variables. This crude form of data collection is cumbersome, prone to logging errors, and lacks precision. In other cases, experimenters are left manually counting and recording events in real-time using a log sheet [22] or video playback [7, 13].

Second, when using commercial games, it is sometimes necessary to evaluate conditions across different games or platforms because they are not available in a single game or platform (e.g., touch on game A vs. tilt on game B). In such cases, it is impossible to reliably distinguish whether the observed differences were due to the test conditions (e.g., touch vs. tilt) or uncontrolled factors inherent to the games or platforms (e.g., processor or display differences between devices) [7, 22]. Such differences across platforms are confounding variables, further compromising internal validity.

In contrast, the advantage of custom games is the greater control one has over experiment design, data collection, game-play, etc. Internal validity is high since test conditions are implemented using the same apparatus and data collection is embedded in the game. The difficulty, of course, is that the process involves implementing a complete game, which is extremely time consuming. Custom-developed games also typically lack the impressive visuals and complex environments of commercial games [7, 13] and are thus unlikely to engage players to the same extent. Consequently, while offering greater internal validity, external validity may be lower with a custom game. However, provided the custom game creates a realistic and engaging environment, external validity can be maintained.

A possible middle ground is to run custom software and game software simultaneously using a platform emulator [21]. Data collection is added to an emulator to gather low-level metrics such as where and when the participant touches on-screen buttons. This option may not provide the variety of input possibilities with either commercial or custom games since input (touch, tilt, or otherwise) is limited to the emulated platform controls.

Based on the relative merits of both approaches, we favour the custom-game option for experimental evaluations. We argue that this allows researchers to "distill" a game to just the elements under study, be they control options, display characteristics, or gameplay elements. Consequently, and unlike other work [6, 12, 14, 22], we developed a game as our experimental framework. Based on the arguments above, this offers a higher degree of experimental control, and may yield more reliable experimental results than previous work using commercial games.

# 4. METHODOLOGY

## 4.1 Participants

Twelve participants were recruited for the experiment. Their ages ranged from 19 to 34 years (*mean* = 24.7, *SD* = 4.7). Nine were male. All had some prior mobile gaming experience. Eight reported playing mobile games (with touch input) several times per month or more often. Tilt input experience was more limited, with three participants indicating they never play tilt-based games. The rest reported playing tilt input games occasionally.

## 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 5. 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 with the Android SDK. The software offered both touch and tilt input, each with position-control and velocity-control.

At the beginning of each level, the ball movement started at the right side of the screen, centered along the edge. Its initial motion was downward to the left – hence it (almost) immediately hit the left wall, which would impart some randomness to its subsequent motion. Consequently, its motion was always unpredictable from the start of the trial.

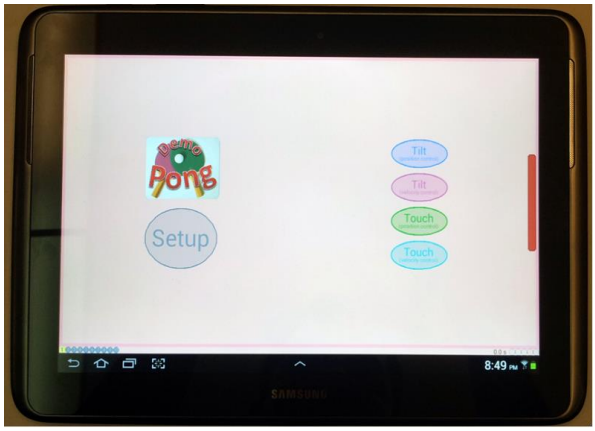



**Figure 5. Samsung *Galaxy Note 10.1* tablet running the experimental software. The software startup screen is shown, including the four control option buttons, and a setup button used for setting experimental parameters.**

### *4.2.1 Touch Input Conditions*

Touch input used a virtual control that appeared at the bottom left of the screen, see Figure 6. This "touch-strip" simulated virtual joysticks or directional pads used in some mobile games. Since Pong only allows paddle movement in the up and down directions, we used a 1-dimensional touch strip rather than a multi-directional virtual joystick – left/right motions were unnecessary. The touch strip was positioned to be easily accessible by the user's left thumb. This was intended to leverage participant familiarity with physical game controllers, which almost universally have the directional controls on the left side for easy access by the left thumb.

In touch + position-control, touching the touch strip moved the paddle position to the corresponding location. For example, touching the middle of the touch strip set the paddle in the middle of the screen (on the right). Similarly, touching the top of the touch strip moved the paddle to the top of the screen.

Touch + velocity-control was similar to isometric joysticks on modern gamepads. Touching the touch strip increased paddle velocity in the specified direction (relative to the centre). This used a linear interpolation between velocities of 0 cm/s (at the centre of the strip) to full speed (25 cm/s) in the direction specified.

### *4.2.2 Tilt Input Conditions*

For tilt + position-control, a neutral tilt angle of 35º about the "roll" axis positioned the paddle in the centre of the screen. The tilt range was ±25º. The extremes of 10º and 60º positioned the paddle at the top and bottom of the screen, respectively. Other positions mapped linearly between these extremes.

For tilt + velocity-control, 35º was again treated as the center or neutral position. This condition otherwise behaved similar to touch + velocity-control. Tilting away from centre linearly mapped the paddle velocity up to extremes of 25 cm/s at 10º/60º, using the same directions as tilt + position-control. The speed was chosen through pilot testing to determine a suitably fast velocity control option that felt comparable to position control.

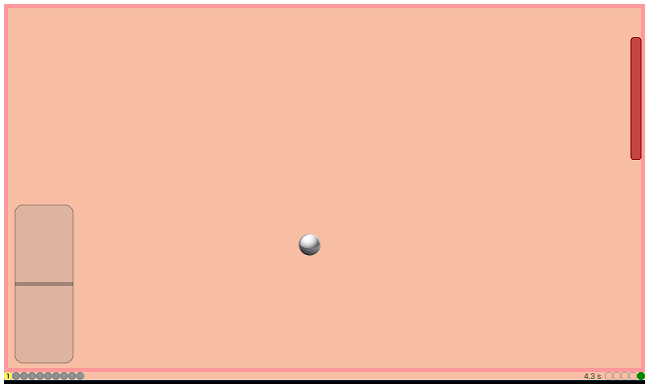


**Figure 6. The experimental software, showing the touch condition. The touch strip is shown at the bottom left, and appeared visually identical in both touch-control conditions. The touch strip was not shown in the tilt-control conditions.**

## 4.3 Procedure

The experimenter first explained the experiment's purpose and participants gave informed consent. The experimenter then explained each condition, allowing participants to practice each for two 5-ball rounds. Participants then performed each condition seated comfortably in a quiet setting. The experiment took approximately 2 hours in total to complete (i.e., four conditions at approximately 30 minutes each). Due to the length of the experiment, testing occurred over two sessions. Each participant took at least a 1-hour break between the two sessions. This was intended to avoid excessive fatigue and boredom on the part of the participants, which could influence the results.

Participants completed 10 game sequences. Each sequence consisted of up to 15 rounds in each condition. Each round was one "level" of the game, consisting of 5 cycles of ball movement – each of which could either hit or miss the paddle. At the end of a sequence, the player's performance was assessed as follows:

- 0 misses – proceed to the next level
- 1-2 misses – repeat level
- 3-5 misses – go back one level

Game difficulty increased in three ways as levels progressed:

- The ball velocity increased. At level 10, the ball velocity was 150% of the velocity at level 1.
- The paddle size decreased. At level 10, the paddle size was 50% of the size at level 1.
- A random offset was applied to the bounce angle at the wall opposite the paddle. The offset increased linearly from 0.1 $x$ at level 1 to $x$ at level 10, where $x$ = 45° × `nextRandom()`. The offset was applied relative to minimum and maximum bounce angles of -45° and +45°, respectively.

Participants were instructed to play the game to the best of their ability. Faster experiment completion was used as an inducement; if participants reached level 10 (the highest difficulty) and completed it without missing the ball, the sequence ended early, rather than requiring all 15 rounds. Hence, a sequence could be as short as 10 rounds, or as long as 15.

## 4.4 Design

The experiment used a 2 × 2 × 10 within-subjects design. The independent variables and levels were

| | |
|---|---|
| Input method: | touch input, tilt input |
| Order of control: | position-control, velocity-control |
| Sequence number: | 1, 2, 3, … 10 |

The four combinations of input method and order of control were counterbalanced with participants divided into four groups according to a balanced Latin square. Thus, "group" was a nominal between-subjects factor.

Sequence number increased sequentially for each condition. As noted above, within each sequence there were 10 to 15 rounds. Each round consisted of 5 trials (ball hits/misses). Hence the design produced between 24,000 and 36,000 trials, depending on participant performance. The actual number of recorded trials was 35,935.

The most common dependent variables in user studies relate to participant speed and accuracy. For speed, we analyzed the reciprocal measure, time: the time for participants to reach various game-levels. We also treated game-level achieved as a dependent variable. For accuracy, we analyzed participants' tendency to hit or miss the ball.

# 5. RESULTS AND DISCUSSION

Results for each dependent variable are presented below. Unless stated otherwise, all statistical analyses used repeated measures ANOVA. Furthermore, all ANOVA tests for group were not statistically significant, indicating that counterbalancing had the desired effect of cancelling potential effects due to the order of testing conditions.

## 5.1 Level Achieved

Figure 7 shows the game-level achieved in each sequence of game-play for each control condition. Each point in the chart is the mean of the top game-level achieved for the twelve participants for the specific condition and sequence. At later sequences, the results coalesce around levels 4-5 for the velocity-control conditions and levels 7-8 for the position-control conditions. Note the clear learning progression in the position-control lines. For velocity-control, there was little learning evident over the course of the ten sequences of game-play. In short, while controlling the paddle using velocity-control tilt or touch, participants struggled!

Since participants may have achieved their highest game-levels in different sequences, the distinction between conditions is even more dramatic if examined overall (vs. by sequence). Figure 8 shows the mean highest game-level achieved by test condition (over all sequences).

Participants achieved a mean high game-level of 8.6 for tilt + position-control. This was more than 2× the high of 4.2 for tilt + velocity-control. For touch input the means of the highest game-levels achieved were 5.7 (position-control) and 4.6 (velocity-control). The main effects were statistically significant both for input method ($F_{1,8}$ = 18.75, $p$ < .005) and for order of control ($F_{1,8}$ = 90.75, $p$ < .0001). In the case of input method, touch input was superior. Additionally, the input method × order of control interaction effect was statistically significant ($F_{1,8}$ = 28.57, $p$ < .0001). As seen in the Figure 8, the difference between position-control and velocity-control for tilt input was much greater than for touch input.

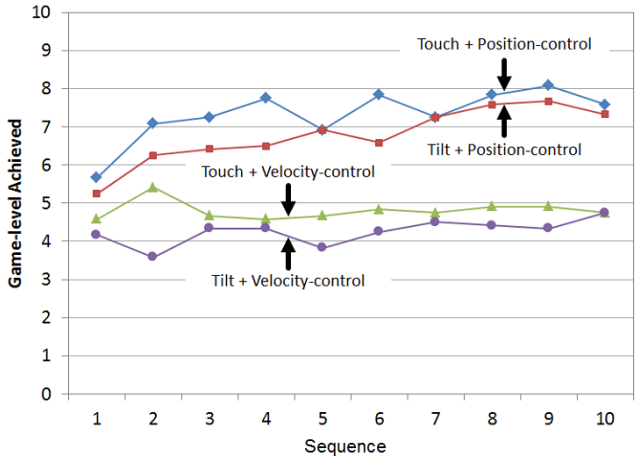



**Figure 7. Average highest game level achieved for each test condition and sequence.**

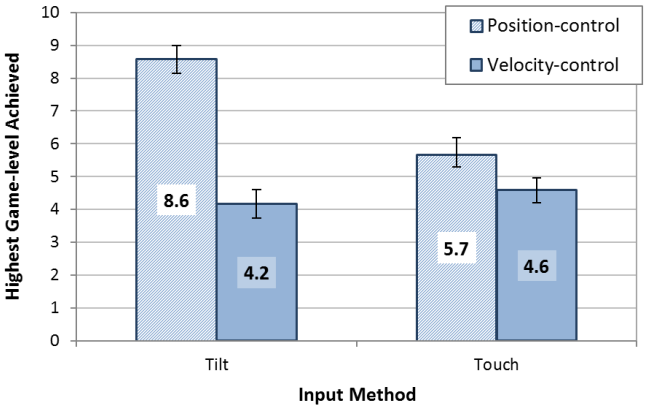



**Figure 8. Participants' highest game-level by input method and order of control. Error bars show ±1 *SE*.**

## 5.2 Miss Percentage

To gauge accuracy, it was logged if the ball hit or missed the paddle for each trial. "Miss percentage" is the number of misses divided by the total number of trials, expressed as a percentage. The grand mean for miss percentage was 20.1%. There was only a modest reduction over the ten sequences for all test conditions

(not shown). This value is expected, since 1 miss in 5 trials was the threshold for advancing or not advancing to the next game level.

Figure 9 shows miss percentage by input method and order of control. The results again favoured position-control over velocity-control, with means of 17.2% (tilt) and 15.8% (touch) for position-control and 24.5% (tilt) and 22.4% (touch) for velocity-control.

Combining the means for touch and tilt, the overall miss percentage during position-control game-play was 16.6%. At 23.5%, the overall mean miss percentage during velocity-control game-play was a substantial 42% higher than position-control. The effects were statistically significant for order of control ($F_{1,8}$ = 114.9, $p < .0001$) but not for input method ($F_{1,8}$ = 4.71, $p > .05$). The input method × order of control interaction effect was not statistically significant ($F_{1,8}$ = 0.56, ns).

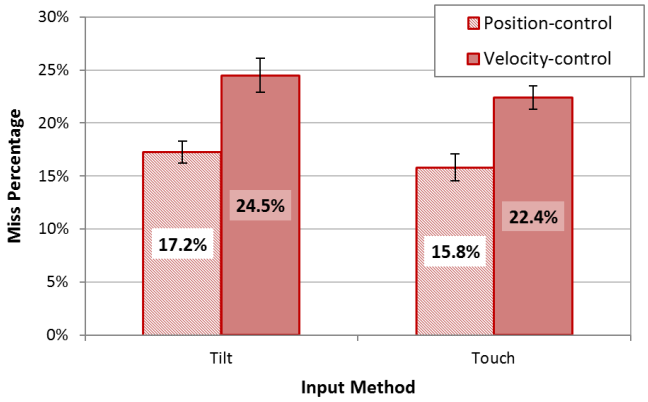



**Figure 9. Miss percentage by input method and order of control. Error bars show ±1 *SE*. Lower scores are better.**

## 5.3 Time to Reach Level

We also examined how long it took participants to reach various game-levels as a function of input method and order of control. Bear in mind that the pace of game-play was set by the design of the game and the experiment. As determined by the ball velocity, a round of five trials took approximately 13 seconds in level 1. Fifteen rounds (1 sequence) took a little over 3 minutes. The time per round decreased during testing, since the ball velocity increased at higher game levels. Overall, participants took 25 to 30 minutes for the 10 sequences of testing for each condition.

Although the pace of game-play was set by design, the level achieved depended on the skill of the participants and the ease with which they could play the game given the current input method (tilt vs. touch) and order of control (position vs. velocity). Using level 5 as an example, Figure 10 shows the mean time to reach the level vs. testing condition.

For the position-control conditions, participants took a little over 3 minutes, on average, to reach level 5. Since each of the 10 sequences of testing took a little over 3 minutes, level 5 was attained in the 1st sequence for some participants, perhaps by the 2nd sequence for others.

For the velocity-control conditions, participants took longer to reach level 5 – about 5 or 6 minutes. Note in Figure 10 the large error bars for the velocity-control conditions. Something unusual is suggested. Indeed, the use of level 5 for this analysis was deliberate: Many participants were unable to reach higher levels of game-play with the velocity-control conditions. Not only was the variability greater in the time to reach level 5, some data points are missing in Figure 10. Two participants failed to reach level 5 for tilt + velocity-control game-play. One participant failed to reach level 5 for touch + velocity-control game-play.

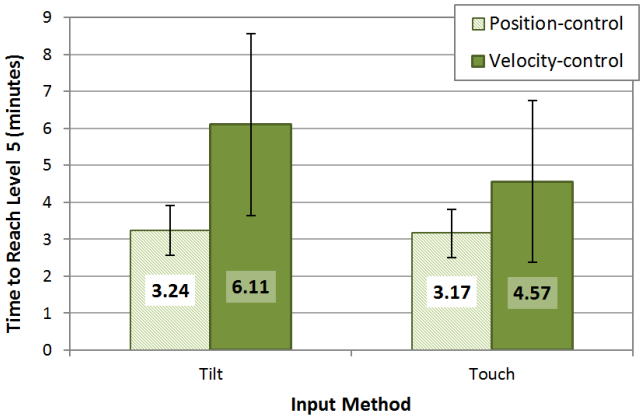



**Figure 10. Time to reach level 5 by input method and order of control. Error bars show ±1 *SE*.**

Since the total testing time was 25-30 minutes per participant per condition, it is worth examining in greater detail what transpired over the course of testing, particularly for participants who languished at the lower levels of game-play. For this analysis, the two touch conditions are chosen for discussion. Figure 11 shows the time to reach the various game levels for each of the twelve participants during game-play using touch input.

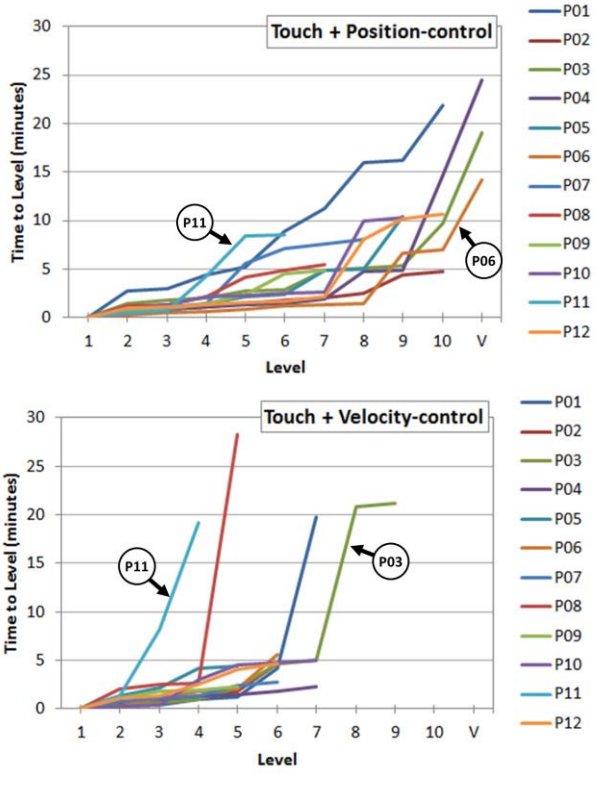



**Figure 11. Time to level for each of 12 participants. (a) touch + position-control game-play. (b) touch + velocity-control game-play. See text for discussion.**

The stories are very different for position-control and velocity-control. Figure 11a depicts touch + position-control. The result is unsurprising. Participants advanced quickly through the game, reaching the mid-levels after one or two rounds (approximately 3-6 minutes). Participants were suitably challenged thereafter. Six participants reached level 10. Of these, 3 achieved Victory ("V" in figure), completing level 10 with zero misses.

The best-performing participant was P06, who reached level 10 in about 7 minutes and Victory in about 14 minutes. The worst-performing participant was P11, who reached level 6 in about 8 minutes but was unable to reach a higher level in the remaining 15 minutes or so of testing. The other participants were between these two extremes.

The situation for touch + velocity-control was quite different. See Figure 11b. Not one of the 12 participants reached level 10. The best-performing participant was P03, whose highest level was 9, reached in about 21 minutes. But, most of the other participants only reached level 5 or 6. Participant P05 only reached level 4, and only after about 18 minutes of game-play.

## 5.4 Subjective Results and Observations

Participants clearly struggled with the velocity-control conditions. This was especially true with tilt + velocity-control, as it was likely the most unusual condition. Participants verbally expressed frustration to the experimenter. One participant commented that this condition was "best when the ball was approaching the top or bottom of the screen". This comment likely referred to the relative ease with which one could move the paddle to the screen edges in this condition. In contrast, keeping the paddle stationary in the screen centre was difficult.

Participants almost unanimously preferred touch + position-control. Only one participant preferred tilt + position-control. This participant mentioned that they found the touchscreen difficult to use. Specifically, the participant noted "it was difficult to know how hard to press – if I didn't press hard enough, my input didn't register, but if I pressed too hard the friction was too high".

Upon finishing the experiment, participants completed a short survey. The survey questions were based, in part, on those recommended in ISO 9241-9 for evaluating computer pointing devices, which was previously employed in evaluating tilt input interfaces [11, 19]. The questions related to the operational smoothness, mental and physical effort required, accuracy, general comfort, and overall ranking. A summary of the responses is shown in Figure 12. In all cases, higher scores are more favourable.

Survey responses were analyzed using the Friedman non-parametric test. Statistical values are also found in Figure 12. The only two questions that did not yield a significant difference between the conditions were "General comfort" and "Physical effort". Overall, the tilt + velocity-control condition received a significantly worse rating than the other conditions. Apparently, the form factor of the tablet did not dramatically influence comfort, however.

# 6. OVERALL DISCUSSION

Clearly, there is a dramatic performance difference between position-control and velocity-control, with the advantage favouring position-control for both tilt and touch input. As discussed above, previous work has suggested that it is important to match the property sensed to the order of control [9, 23]. Given that both input methods employ position sensing of an input

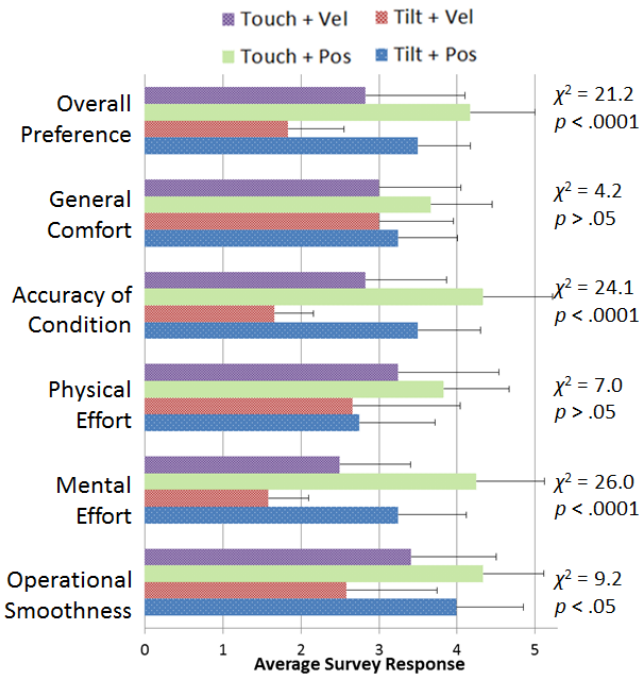



**Figure 12. Average response scores from the participant survey by question. Statistical values shown to the right. Higher scores are better. Errors bars show +1 *SD*.**

property, these results are not entirely surprising. That position-control tended to perform better is further supported by other previous work [16, 19].

What is surprising is the comparatively small difference between touch and tilt input for most dependent variables recorded. As discussed earlier, previous work has not definitively answered which control method offers better performance [2, 6, 14, 17]. In terms of game-level achieved and miss percentage, our results suggest little performance difference in the two methods. However, touch was superior to tilt for highest game-level achieved.

It is possible that previous results on touch vs. tilt control (for games) have been somewhat mixed due to this relatively small margin of difference between these control styles. As discussed earlier, a principle advantage of having developed a custom game for our experiment is the ability to have greater control over studies using commercial games. We thus have greater confidence in our results than had we used a commercial game with both tilt and touch options.

The quick feedback may partially explain the difference between position- and velocity-control. With position-control, feedback is more immediate – the participant simply touches or tilts the device, and the paddle moves instantaneously to the desired location. In contrast, feedback is delayed with velocity-control. Rather than immediately positioning the paddle, this condition increases the paddle speed in the desired direction. Participants must plan their motions carefully while watching the movement of the paddle. This is likely reflected in the significantly worse ranking for "mental effort required" for the velocity-control modes, especially tilt-velocity (see Figure 12).

One participant commented that the touch strip should be positioned on the right side of the screen, as they were right-handed. The design intent in putting the touch strip on the left was

consistency with classic gamepads. Over the past thirty years, game console controllers have almost universally included the directional controls on the left side of the device for use with the left thumb. Modern console gamepads offer some flexibility, as they typically include two analog sticks (one for each thumb). The primary advantage of touchscreens is flexibility, so it would be possible to dynamically position the virtual controls. Some games already do this, in fact. Nevertheless, we expect that switching which side of the screen the directional control appears on would have little impact on our results.

One final consideration relates to the actual size of the touch strip used for the touch input modes. The touch strip was designed to be easily accessible with the left thumb. In practice, however, some participants (particularly female participants with smaller hands) mentioned that they felt the touch strip was slightly too large for their thumb's range of motion. This may further help explain the relative *absence* of difference between touch and tilt input – perhaps with a more accommodating touch strip size, touch input may have offered better performance over all.

# 7. RECOMMENDATIONS FOR GAME DEVELOPERS

Based on our results, we now present suggestions for game developers considering tilt and touch input options.

### *7.1.1 Offer both touch and tilt input*

As mentioned earlier, some games allow players a choice of control option. We argue that this is a good design decision as our results suggest little difference between these control styles. Moreover, as indicated in Figure 12, tilt + position-control came in a close second in overall preference to the touch + position-control condition. Consequently, we suggest letting players choose which control option they prefer. This can be helpful, for example, if (as indicated by some participants) they find their thumb becoming irritated from extended use of the touchscreen. Alternatively, if tilt input becomes fatiguing or straining on the wrist, participants could switch to touch input instead.

### *7.1.2 Consider position-control mappings*

Order of control had a much stronger impact on our results than input method. Regardless of input method, position-control globally offered superior performance than velocity-control. Consequently, we suggest that developers consider position-control mappings where possible and appropriate.

Note that our results may not apply to scrolling games. In these games, usually velocity-control mappings are used. With tilt input or indirect touch input like that used in our study, it is impossible to uniquely specify an off-screen position directly with the input method. Hence in these cases, it is likely that position-control is simply infeasible.

Conversely, velocity-control is still used in games that do *not* use unbounded scrolling. Marble maze games are a good example; all gameplay occurs on a single screen, while ball position is determined according to a tilt + velocity mapping. We suspect that position-control may offer better performance in these games, even though the experience is less realistic. This is a point for further study, however.

### *7.1.3 One size may not fit all*

Based on our observations, participants with smaller hands had difficulty effectively using virtual controls designed for larger hands. Consequently, designers should consider including options to rescale these virtual controls. This could be in the form of a slider to allow continuous scaling, and may further influence the relative distances between virtual buttons (should the game use any).

### *7.1.4 Allow changing control positions*

Based on participant feedback, it may be beneficial to consider changing the position of virtual controls to accommodate user preference or handedness. Some games already support this – for example, *Grabatron* positions the virtual joystick at the position of the first touch point of the screen. Many games, however, simply position controls in a default layout, similar to console gamepads. We also made the assumption that this would prove to be a natural control layout, but providing an option to switch this is worthwhile.

# 8. CONCLUSIONS

We conducted an experiment comparing position-control and velocity-control mappings across two input methods, tilt and touch input. The objective of this work was to better understand control styles in mobile games. Results of our study indicate that, at least for non-scrolling games, order of control matters more than input method. Position-control consistently outperformed velocity-control, regardless if it was implemented with touch or tilt input. Our hope is that developers can use these results to make informed decisions around control options in their games.

## 8.1 Future Work

A clear avenue for future work is to investigate how these results extend to a game with a scrolling viewpoint. As mentioned earlier, it is likely that velocity-control mappings make more sense in these cases. Novel control methods would be required to employ position-control in such games. Such options might include combining an additional information source (e.g., multitouch or pressure) to expand or amplify the range of movement while using a position-control mapping.

Another option for future work is to further compare these control styles to physical controls [18]. Physical controls are comparatively rare on mobile devices today, but are ubiquitous on dedicated game devices (e.g., handhelds such as the Nintendo *3DS*). Previous work [21, 22] has demonstrated that physical controls tend to perform better than touch-based controls for games. However, this work used commercial games, and again suffers from the relative lack of control in the design of the experiments. We plan to address this using custom games supporting both touch and physical controls.

## 8.2 Acknowledgments

Thanks to all participants for taking part in the study. This work was supported by NSERC.

# 9. REFERENCES

[1] Balakrishnan, R. and MacKenzie, I. S. 1997. Performance differences in the fingers, wrist, and forearm in computer input control, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, USA, March 22-27, 1997). CHI '97. ACM, New York, NY, 303-310. DOI= http://dx.doi.org/10.1145/258549.258764

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

[3] Chu, K. and Wong, C. Y. 2011. Mobile input devices for gaming experience, In *Proceedings of the IEEE International Conference on User Science and Engineering* (Selangor, Malaysia, Nov. 29 - Dec. 1, 2011). IEEE, New York, NY, 83-88. DOI= http://dx.doi.org/10.1109/iUSEr.2011.6150542

[4] Gilbertson, P., Coulton, P., Chehimi, F., and Vajk, T. 2008. Using "tilt" as an interface to control "no-button" 3-D mobile games, *Computers in Entertainment*, 6, 38, (Oct. 2008), 1-13.

[5] Henrysson, A., Marshall, J., and Billinghurst, M. 2007. Experiments in 3D interaction for mobile phone AR, In *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia* (Perth, Australia, Dec. 1-4 2007). ACM, New York, NY, 187 - 194. DOI= http://dx.doi.org/10.1145/1321261.1321295

[6] Hynninen, T. 2012. 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.

[7] Järvelä, S., Ekman, I., Kivikangas, J. M., and Ravaja, N. 2012. Digital games as experiment stimulus, In *Proceedings of DiGRA Nordic 2012* (Tampere, Finland, June 6-8 2012). Digital Games Research Association.

[8] Kantowitz, B. H. and Elvers, G. C. 1988. Fitts' Law with an isometric controller: effects of order of control and control display gain, *Journal of Motor Behavior*, 20, (Sept. 1988), 53-66.

[9] Kim, W. S., Tendick, F., Ellis, S. R., and Stark, L. W. 1987. A comparison of position and rate control for telemanipulations with consideration of manipulator system dynamics, *IEEE Journal of Robotics and Automation*, 3, (Oct. 1987), 426-436. DOI= http://dx.doi.org/10.1109/JRA.1987.1087117

[10] Lee, S. and Zhai, S. 2009. The performance of touch screen soft buttons, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems* (Boston, US, April 4-9). CHI 2009. ACM, New York, NY. 309-318. DOI= http://dx.doi.org/10.1145/1518701.1518750

[11] MacKenzie, I. S. and Teather, R. J. 2012. FittsTilt: The application of Fitts' law to tilt-based interaction, In *Proceedings of the 7th Nordic Conference on Human-Computer Interaction* (Copenhagen, Denmark, Oct. 14-17, 2012). NordiCHI 2012. ACM, New York, NY, 568-577. DOI= http://dx.doi.org/10.1145/2399016.2399103.

[12] McMahan, R. P., Alon, A. J. D., Lazem, S., Beaton, R. J., Machaj, D., Schaefer, M., Silva, M. G., Leal, A., Hagan, R., and Bowman, D. A. 2010. Evaluating natural interaction techniques in video games, In *Proceedings of the IEEE Symposium on 3D User Interfaces* (Waltham, USA, March 20-21, 2010). 3DUI 2010. IEEE, New York, NY, 11-14. DOI= http://dx.doi.org/10.1109/3DUI.2010.5444727

[13] McMahan, R. P., Ragan, E. D., Leal, A., Beaton, R. J., and Bowman, D. A. 2011. Considerations for the use of commercial video games in controlled experiments, *Entertainment Computing*, 2, (Jan. 2011), 3-9. DOI= 10.1016/j.entcom.2011.03.002

[14] Medryk, S. and MacKenzie, I. S. 2013. A comparison of accelerometer and touch-based input for mobile gaming, *International Conference on Multimedia and Human-Computer Interaction*, (Toronto, Canada, July 17-18, 2013). International ASET, Ottawa, Canada, 117.1-117.8.

[15] Mine, M. R., Frederick P. Brooks, J., and Sequin, C. H. 1997. Moving objects in space: exploiting proprioception in virtual-environment interaction, In *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques* (Los Angeles, USA, Aug. 3-8, 1997). SIGGRAPH '97. ACM, New York, NY, 19-26. DOI= http://dx.doi.org/10.1145/258734.258747

[16] Oakley, I. and O'Modhrain, S. 2005. Tilt to scroll: Evaluating a motion based vibrotactile mobile interface, In *Proceedings of the Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (Pisa, Italy, March 18-20, 2005). IEEE, New York, NY, 40-49. DOI= http://dx.doi.org/10.1109/WHC.2005.138

[17] Oshita, M. and Ishikawa, H. 2012. Gamepad vs. touchscreen: a comparison of action selection interfaces in computer games, *Proceedings of the Workshop at SIGGRAPH Asia* (Singapore, Nov. 28-Dec. 1, 2012). ACM, New York, NY, 27-31. DOI= http://dx.doi.org/10.1145/2425296.2425301

[18] Silfverberg, M., MacKenzie, I. S., and Kauppinen, T. 2001. An isometric joystick as a pointing device for handheld information terminals, *Proceedings of Graphics Interface* (Ottawa, Canada, June 7-9, 2001). CIPS, Toronto, Canada, 119-126.

[19] Teather, R. J. and MacKenzie, I. S. 2014. Position vs. velocity control for tilt-based interaction, *Proceedings of Graphics Interface* (Montreal, Canada, May 7-9, 2014). CIPS, Toronto, Canada, 51-58.

[20] Wigdor, D. and Balakrishnan, R. 2003. TiltText: Using tilt for text input to mobile phones, *Proceedings of the ACM Symposium on User Interface Software and Technology* (Vancouver, Canada, Nov. 2-5, 2003). UIST 2003, ACM, New York, NY, 81-90. DOI= http://dx.doi.org/10.1145/964696.964705

[21] Zaman, L. and MacKenzie, I. S. 2013. Evaluation of nano-stick, foam buttons, and other input methods for gameplay on touchscreen phones, *Proceedings of the International Conference on Multimedia and Human-Computer Interaction* (Toronto, Canada, July 17-18, 2013). International ASET, Ottawa, Canada, 69.1-69.8.

[22] Zaman, L., Natapov, D., and Teather, R. J. 2010. Touchscreens vs. traditional controllers in handheld gaming, *Proceedings of the International Academic Conference on the Future of Game Design and Technology* (Vancouver, Canada, May 6-7, 2010). FuturePlay 2010. ACM, New York, NY, 183-190. DOI= http://dx.doi.org/10.1145/1920778.1920804

[23] Zhai, S., User performance in relation to 3D input device design. 1998, *ACM Siggraph Computer Graphics*, 32, (Oct. 1998), 50-54.

[24] Zhai, S., Milgram, P., and Buxton, W. 1996. The influence of muscle groups on performance of multiple degree-of-freedom input, *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, Canada, April 13-18, 1996). CHI '96, ACM, New York, NY, 308-315. DOI= http://dx.doi.org/10.1145/238386.238534