UniGest: Text Entry Using Three Degrees of Motion

Steven J. Castellucci
Department of Computer Science
and Engineering
York University
4700 Keele St.
Toronto, Ontario M3J 1P3 Canada
stevenc@cse.yorku.ca

I. Scott MacKenzie
Department of Computer Science
and Engineering
York University
4700 Keele St.
Toronto, Ontario M3J 1P3 Canada
mack@cse.yorku.ca

Abstract

This paper introduces *UniGest*, a technique that provides pointer input and text entry in a single device without occupying the display. It uses a Nintendo *Wii* motion-sensing remote to capture gestures that are mapped to character input. A gesture alphabet is proposed, with each gesture composed of at most two primitive motions. A web-based user study measured movement times for primitive motions. Results range from 296 to 481 ms, implying an upper-bound *UniGest* text entry performance prediction of 27.9 wpm.

ACM Classification Keywords

H5.2 [Information interfaces and presentation]: User Interfaces – Input devices and strategies.

Keywords

Text entry, motion-sensing, unistrokes, game controller

Introduction

Pointer-based onscreen keyboards allow navigation and text entry using a single device (e.g., a mouse or a stylus). However, onscreen keyboards take up valuable screen real estate. Furthermore, keyboards are not convenient in presentation or entertainment situations, which often lack a desktop. This paper introduces *UniGest*, a motion-based technique that facilitates both pointer input and text entry without occupying the display.

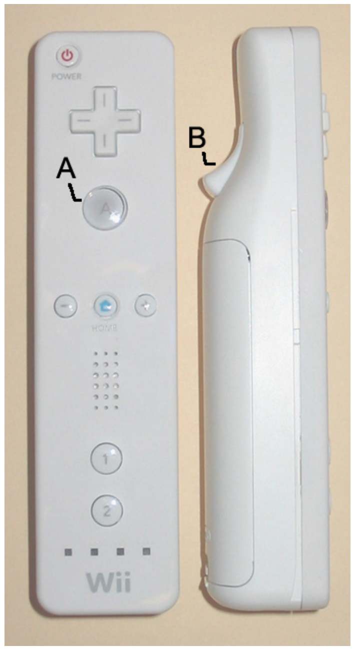

Figure 1. The *Wii Remote* (a.k.a. *Wiimote*), the commercial device used to implement *UniGest*. The large A-button is visible on the top of the left *Wiimote*. The B-button is the trigger on the underside, visible on the right *Wiimote*.

UniGest uses the game controller for the *Wii* console by Nintendo (www.nintendo.com). The *Wii Remote* (aka *Wiimote*) appears in Figure 1. Unlike conventional game controllers, it incorporates a motion-sensing accelerometer. To provide precise pointing input, the *Wiimote* uses a sensor bar placed above or below the display. The sensor bar provides points-of-reference via a series of infrared light-emitting diodes (LEDs).

After reviewing related work, we describe *UniGest*. To predict *UniGest*'s text entry performance, an empirical study was conducted to gather movement time values. Details and results follow. This paper then concludes and outlines future work.

Related Work

Unistrokes

First introduced at the ACM *SIGCHI* conference in 1993, *Unistrokes* is a gesture alphabet for stylus-based text entry [1]. Its gesture alphabet appears in Figure 2.

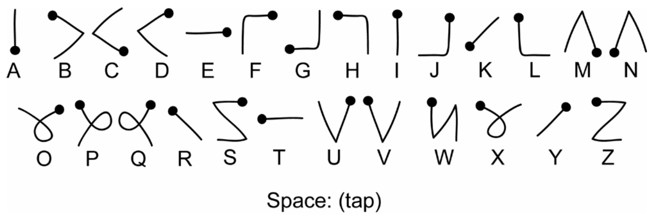

Figure 2. The *Unistrokes* alphabet. Dots indicate starting positions for each gesture.

Unistroke gestures bare little resemblance to their associated Roman letters. However, each letter is assigned a short stroke, with frequent letters (e.g., E, A, T, I, R) associated with a straight line. The single-stroke idea allows easy entry without attending to the writing area. Furthermore, the alphabet's strokes are well distinguished in "sloppiness space" [1], allowing accurate recognition of inaccurate (i.e., sloppy) input. By employing motion sensing, *UniGest* aims to utilize *Unistrokes'* robustness, while removing the necessity of a touch screen or digitizing tablet.

Text Entry with Game Controllers

Wilson and Agrawala [5] devised a technique that mapped the left and right joysticks of a conventional game controller to the left and right half of an onscreen QWERTY keyboard. Users employ the appropriate joystick to highlight the intended letter, and then press the left or right trigger button to select it. Wilson and Agrawala hoped to exploit users' familiarity with the QWERTY layout [5]. Their user study yielded text entry performance of 7.1 wpm (words-per-minute) with novice users.

Sandnes and Aubert [3] also mapped the QWERTY layout to a dual joystick controller. However, their technique does not display an onscreen keyboard. Instead, users imagine that the left and right joysticks represent positions between the D and F keys and J and K keys, respectively. Users move the appropriate joystick in the direction of the intended letter. A letter is entered by releasing the joystick, allowing it to spring back to the rest position. A language model resolves the ambiguity of having multiple letters assigned to each joystick position. In their study, participants achieved a text entry rate of 6.8 wpm [3].

Koltringer et al.'s *TwoStick* [2] employs a non-Qwerty layout. Users are presented with a nine-by-nine onscreen grid divided into zones. Each unit in the grid represents (at most) one character (some are empty). To input a character, one joystick selects a zone, while the other selects a character within that zone. The user enters the character by returning the character selection joystick to its rest position. A user study yielded text entry rates of 4.3 wpm initially, increasing to 14.9 wpm after five hours of practice [2].

UniGest

Because of space restrictions, this paper is limited to *UniGest*'s text entry feature. A user enters a character by holding the B-button and performing its corresponding gesture. Releasing the B-button terminates the gesture. The current *UniGest* alphabet appears in Figure 3. Each gesture is composed of primitive motions: up, down, left, right, up-left, up-right, down-left, down-right, roll-left, and roll-right. The gesture set was designed with the following goals:

- Composition involving, at most, two primitive motions, with frequent letters (i.e., E, T, A, O, and I) requiring only one such motion
- Separation within sloppiness space
- Similarity to lower case Roman letters (where possible)

For text entry evaluation, only the twenty-six lowercase letters of the English alphabet, and SPACE and BACKSPACE characters, are currently defined. However, composing gestures with a maximum of two of the ten primitive motions allows up to 100 gestures. Furthermore, capitalization can be triggered by pressing a dedicated key on the *Wiimote*, allowing even more gestures.

Method

A web-based empirical study gathered movement time values for each primitive motion. These were used to calculate the anticipated entry time for each letter of alphabet, plus the SPACE character. A text entry performance prediction (in wpm) was calculated by using a language model with word frequencies. A similar technique was used to predict text entry performance on mobile phones [4].

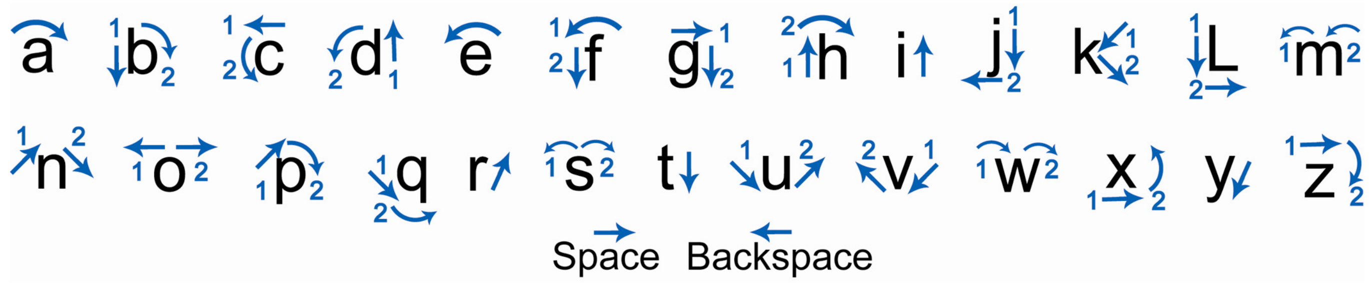

Figure 3. The *UniGest* alphabet. Motions are recognized when holding the B-button on the *Wiimote*. When a character requires two motions, they are numbered in order. For example, to enter 'b', users hold the B-button, move the *Wiimote* downward, tilt it right, and release the B-button.

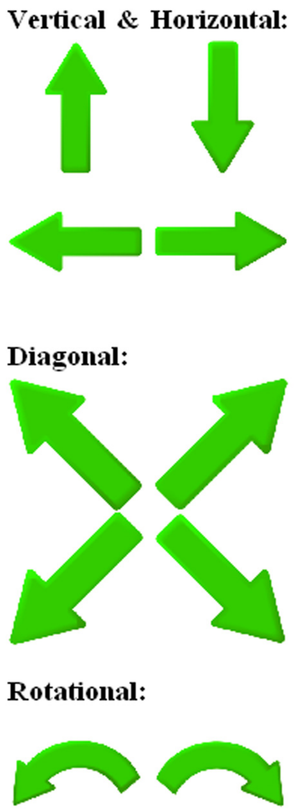

Figure 4. A portion of the user study's instruction page, depicting the prompts for motions.

Participants
Eighteen people participated in a week-long study. They were recruited by posting links in gaming and electronics message boards (a.k.a. forums). Although we were hoping to enlist some female participants, all respondents were male. Ages ranged from 18 to 42 with an average of 23.5 (*SD* = 5.9). Two participants were left-handed, and five performed the study while standing, as determined through a questionnaire (discussed later). Three used the first version of the *Wii*'s optional Internet Channel (9.10), while the remaining fifteen had upgraded to the latest one (9.30). The *Wii*'s Internet Channel is not preinstalled on the console – it must be explicitly downloaded. Furthermore, as of July 2007, the Internet Channel is no longer available as a free download. This might have contributed to the lack of additional participation in the study.

An advantage of conducting a web-based study is the ability to reach an international audience. Fourteen participants were located in North America, two in Europe, one in Asia, and one in South America. Participants were also classified according to *Wii* usage as Novice (2; less than five hours per week), Intermediate (8; five to ten hours per week), or Expert (8; more than ten hours per week).

Apparatus
Participants accessed the study web page using a *Wii* equipped with the Internet Channel. They were allowed to sit or stand during the study, and use either version of the Internet Channel (9.10 or 9.30). They were allowed to rest between trials, as desired.

Design
The independent variable, Motion, had ten levels, one for each primitive motion. The study employed a repeated-measures factor, Block, with levels 1 to 10. Each block contained 10 trials, one for each gesture. The order of the motions was governed by a 10-by-10 balanced Latin Square. The dependent variable was Movement Time in milliseconds. A JavaScript *Wiimote* API allowed gathering of *Wiimote* positional data using the *Wii*'s Internet Channel.

Procedure
Upon following an HTML link to the study (www.cse.yorku.ca/~stevenc/wiistudy/), each participant was presented with a consent form and instructions for the study. The instructions presented images demonstrating the motions (Figure 4). A description accompanied each image. A practice block of trials was given, followed by ten blocks of experimental trials.

Because of API restrictions, the A-button (not the B-button as desired) was used to indicate motion input during the study. Figure 5 illustrates the data-gathering screen. An image appeared in the center of the screen to prompt for a motion. Each participant held the A-button, performed the motion, and released the A-button. During each A-button event, the timestamp and *Wiimote* position were saved. Upon releasing the A-button, the *Wiimote*'s start and end positions were compared to ensure that the required motion occurred. If so, the duration of the motion was saved, and the next motion's image was presented. A progress bar (positioned at the top of the screen) was updated at the completion of each block. At the conclusion of the study, each participant completed a questionnaire to

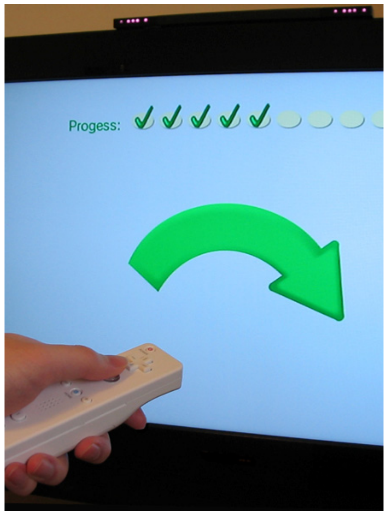

Figure 5. The study's data gathering page. An individual is rolling the *Wiimote* to the right, as prompted on-screen. The progress bar at the top of the screen indicates that he has completed approximately half of the study. Also visible is the sensor bar (on top of the television), whose infrared LEDs are invisible to the un-aided eye.

gather demographic and positional data (i.e., sitting or standing during the study).

Results and Discussion

Movement Times

Table 1 lists the movement times for each motion. With ten blocks for eighteen participants, each value is the mean of 180 trials. The 22 ms difference between Roll-left and Roll-right movement times was not statistically significant ($F_{1,17}$ = 3.37, p > .05). However, the discrepancy between linear and rotational movement time was ($F_{1,17}$ = 14.28, p < .005). The average movement time per block decreased from 345 ms in Block 1 to 326 ms in Block 10, approximately five minutes later (see Table 2). The main effect of Block on Movement Time was significant ($F_{9,153}$ = 2.61, p < .01).

Motion	*Mean* (ms)	*SD* (ms)
Up	310	149
Down	312	149
Left	297	125
Right	296	136
Up-left	321	159
Up-right	316	152
Down-left	310	161
Down-right	326	176
Roll-left	459	233
Roll-right	481	248

Table 1. Movement times by primitive motions

Allowing participants to use their own *Wii* consoles in their own environments hinders internal validity; however, it greatly benefits external validity. Furthermore, analysis of recorded random variables revealed that handedness ($F_{1,16}$ = 3.11, p > .05), position during the study ($F_{1,16}$ = 3.11, p > .05), previous *Wii* experience ($F_{2,15}$ = 0.56, ns), and Internet Channel version ($F_{1,16}$ = 3.11, p > .05), did not have a significant effect on Movement Time.

Block	*Mean* (ms)	*SD* (ms)
1	345	159
2	337	164
3	346	154
4	339	147
5	359	140
6	347	170
7	355	165
8	337	153
9	338	165
10	326	157

Table 2. Movement times by block

Text Entry Rate Prediction

Given the measured times for primitive motions and gesture-to-letter mappings, it is straight-forward to compute a predicted text entry rate for a language such as English given a word-frequency list. Using Silfverberg et al.'s method [4], the prediction for *UniGest* is 27.9 wpm. This figure is a theoretical maximum and does not account for the time to recall a desired character's gesture, the duration between gestures, or the time to return one's hand to a rest position after a character's entry. Still, even if actual text entry speed is only a quarter of this (i.e., 7.0 wpm), *UniGest* is comparable to the text entry rates reported by Wilson and Agrawala [5] (7.1 wpm), Sandnes and Aubert [3] (6.8 wpm), and Koltringer et al. [2] (4.3 to 14.9 wpm).

Conclusion and Future Work

UniGest is a gesture-based technique for pointer input and text entry. Using the *Wiimote* as an input device, movement times ranged from 296 to 481 ms for primitive motions. Using these movement times, a text entry model predicts an upper-bound *UniGest* performance of 27.9 wpm.

As future work, we may revise the *UniGest* alphabet. Given the measured movement times, performance prediction of prospective alphabets can be derived and compared easily. We also plan to conduct a detailed empirical study to evaluate actual text entry performance using *UniGest*, and to evaluate pointer-based throughput using the *Wiimote*.

Acknowledgements

We would like to thank Rob Teather and Vicky McArthur for their help with the web-based study. This research was funded by the Natural Sciences and Engineering Research Council of Canada.

References

[1] Goldberg, D. and Richardson, C. Touch-typing with a stylus. *Proc. Interact 1993* and *Proc. CHI 1993*, ACM Press, 1993, 80-87.

[2] Koltringer, T., Isokoski, P. and Grechenig, T. TwoStick: Writing with a game controller. *Proc. Graphics Interface 2007*, ACM, 2007, 103-110.

[3] Sandnes, F. E. and Aubert, A. Bimanual text entry using game controllers: Relying on users' spatial familiarity with QWERTY. *Interacting with Computers* 19 (2). 140-150.

[4] Silfverberg, M., MacKenzie, I. S. and Korhonen, P., Predicting text entry speed on mobile phones. *Proc. CHI 2000*, ACM Press, 2000, 9-16.

[5] Wilson, A. D. and Agrawala, M. Text entry using a dual joystick game controller. *Proc. CHI 2006*, ACM, 2006, 475-478.