TiltType: Accelerometer-Supported Text Entry for Very Small Devices

Kurt Partridge, Saurav Chatterjee, Vibha Sazawal, Gaetano Borriello[†]
Computer Science and Engineering
University of Washington
Box 352350
Seattle, WA 98195-2350 USA
+1-206-685-9432
{kepart|sauravc|vibha|gaetano}@cs.washington.edu

Roy Want
Intel Research
2200 Mission College Blvd.
Santa Clara, CA 95052-8119
+1-408-765-9204
roy.want@intel.com

ABSTRACT

TiltType is a novel text entry technique for mobile devices. To enter a character, the user tilts the device and presses one or more buttons. The character chosen depends on the button pressed, the direction of tilt, and the angle of tilt. TiltType consumes minimal power and requires little board space, making it appropriate for wristwatch-sized devices. But because controlled tilting of one's forearm is fatiguing, a wristwatch using this technique must be easily removable from its wriststrap. Applications include two-way paging, text entry for watch computers, web browsing, numeric entry for calculator watches, and existing applications for PDAs.

KEYWORDS: Input/output devices, interaction techniques, wearable computing, mobile devices, text entry, accelerometer applications, wristwatch computers.

INTRODUCTION

Among text-entry devices for the desktop, the keyboard has no rival. But among mobile devices, it is far from dominant. Mobile devices are small, and users type much more slowly on small keyboards than they do on full-sized ones (see [12] for one study). Some mobile devices do use a physical keyboard or a soft keyboard, but most PDAs also recognize handwriting, and most cellphones use the phone's twelve-key keypad for text entry.

Still smaller devices, such as wristwatches and small one-way pagers, use a different text entry technique called navigate/select. Navigate/select is nothing like a keyboard. The user first scrolls through a sequence of characters to find the desired one and then selects it. Navigate/select tends to be slow and tiresome because the user must constantly monitor the display during the navigation step. The widespread use of navigate/select among very small devices is unfortunate, because better text entry would benefit several applications including instant messaging, calendaring, web browsing, and general purpose computing (e.g. on the Linux Wristwatch).

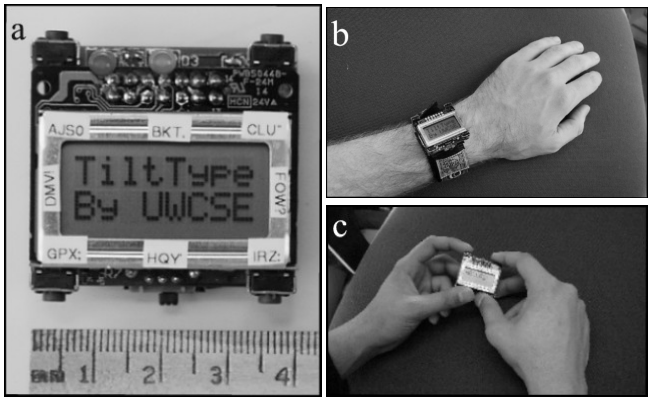

Figure 1: TiltType prototype for text entry. (a) Labels indicate characters that can be entered by tilting in that direction. (b) TiltType is small enough to be worn on the wrist. (c) Two hands are necessary for text entry.

We have constructed a watch-sized text entry device called TiltType (see Figure 1). TiltType uniquely combines two kinds of input: tilting and button pressing. Tilt is particularly suited to small devices (although not limited to them) because tilt sensors can be compact. Button pressing provides natural haptic and audio feedback, which tilting lacks. Used together, tilting and button pressing enable a greater expressiveness than either technique alone.

THE TILTTYPE USER INTERFACE

Our prototype requires two hands to operate. Strictly speaking, one-handed text entry is possible while the device is strapped on a wrist. But this approach is fatiguing because tilting the device requires tilting the whole forearm. Therefore, watches that use TiltType must be easily removable

[†]Gaetano Borriello is jointly affiliated with Intel Research, Seattle, and the University of Washington.

from the wrist. Our prototype uses Velcro, although a finished product would probably use a more rugged locking mechanism.

To enter a character, the user either tilts the device in one of the eight compass directions (North, Northeast, East, etc.) or keeps it level. When the user presses a button, the device tentatively displays the character corresponding to that tilt direction and button. If the user changes the tilt direction while holding the button, the displayed character changes to the character assigned to the new tilt direction. Releasing the button commits the displayed character to the screen.

Using nine tilting directions and three of TiltType's four buttons allows twenty-seven character entries. The English alphabet just fits, with one entry left for a space character.

In addition to the normal tilt angles, the device responds to "extreme" tilt angles that are close to vertical. Extreme tilt angles make more positions available for symbols of other languages, and allow "0" to be entered using the same buttons that the other numerals use.

The fourth button is used for backspace and other special features. Because backspace is a common operation, we made it easiest to perform, by a press and release, regardless of tilt angle. By using this fourth button as a shift key, the user accesses other character sets such as numbers and punctuation, and invokes other features such as capitalizing the next character and clearing the screen.

Figure 2 shows the mapping of the most common characters. The remaining printable ASCII characters can be entered by other tilt/button combinations. In the hope of shortening learning time, we placed the letters in alphabetic order. The Future Work section below describes our ideas for constructing an alternative mapping designed for speed.

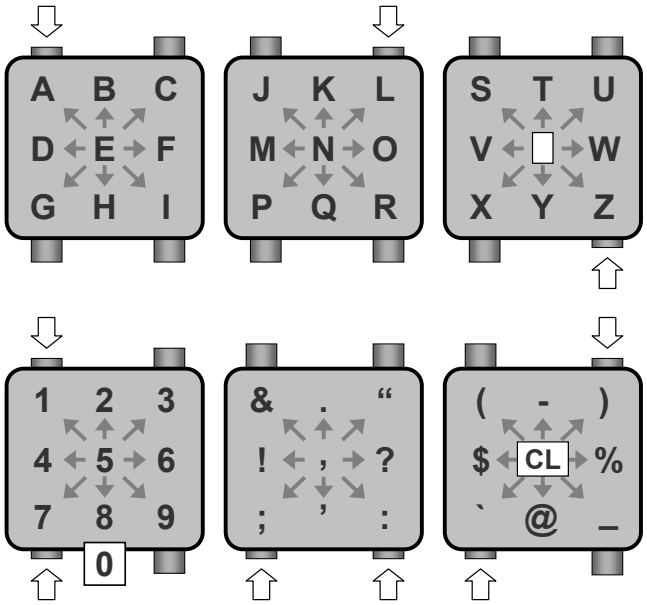

Figure 2: Mapping of positions to the most common characters. Special characters include the space character, zero, and caps lock. Zero is selected by an extreme downward tilt (see text).

THE MECHANICS OF TILTTYPING

Figure 3 shows how TiltType maps tilt into characters. In Figure 3a, the user enters a "C" by pressing the upper left button and tilting the device forward and right for a total tilt of about 30 degrees. Assume for the moment that all acceleration is due to gravity. Gravity acts as a constant downward force ending on the surface of a sphere surrounding the origin. Projecting the gravity vector into TiltType's plane gives the point whose X and Y coordinates are reported by the accelerometer. The radial distance to the point is proportional to the sine of the tilt angle. The set of valid points forms a disc, with points on the perimeter corresponding to a full tilt of 90 degrees.

Figure 3b shows the role of fixed thresholds in mapping the coordinate space to character cells. We chose Cartesian thresholds instead of radial thresholds to simplify the computation. This approach distorts the mapping somewhat. Only about 15 degrees of tilt is needed to select a side character such as "F", but over 21 (or $15\sqrt{2}$) degrees is necessary to select a corner character such as "C". The extreme positions need at least 60 degrees of tilt in a side direction. The Cartesian thresholds make extreme positions unreachable if a corner is tilted too far down.

Fixed thresholding is not the only way to map acceleration into characters. We tried out another approach in which the tilt at the time of each button press marked the center of the character grid. Thresholds were set relative to this point. But this method proved confusing. As users searched for characters, they expected that leveling the device would select the character in the center position. They also expected that repeated presses of the same button at the same tilt angle would enter the same character, whether it was in the center position or not. For these reasons, we implemented fixed thresholds. To accommodate users that prefer a slight offset to all tilts, we added a recalibration operation that permanently changes

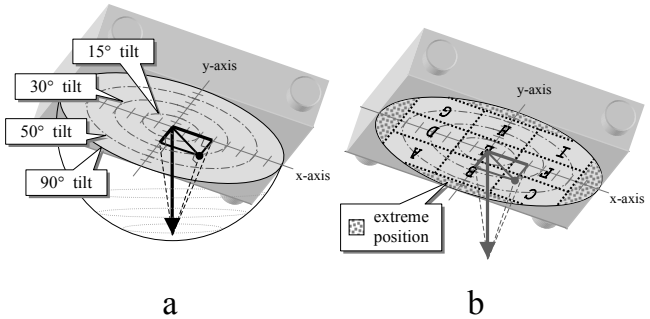

Figure 3: Characters are chosen by projecting the acceleration vector into the plane of the device and comparing the projection's coordinates to fixed thresholds.

the origin's position.

Earlier we assumed that gravity was the only force. In reality, additional acceleration can be externally imposed, such as when the user is inside a moving vehicle. We used TiltType on a bus and found that the error rate did increase, but text entry was not impossible.

Additional acceleration also occurs when the user translates or rotates the device. Rapid translational acceleration can significantly affect the measurements. Movements over a large distance during normal use are unusual, but intense forces over a small distance can occur when the user releases a button. We overcome this problem by buffering the measurements and using the values recorded a few milliseconds before the button release takes place [1].

Rotational acceleration does become noticeable at faster text entry speeds. But this effect is exaggerated in our prototype because the accelerometer is located in a corner of the board to simplify design. An accelerometer at the device's center should be less affected by angular acceleration.

INFORMAL OBSERVATIONS

We have not run controlled experiments, but we did demonstrate TiltType at a public open house. TiltType was well received. Of the fifty or so people who tried it (mostly children between ages eight and sixteen), all but one were able to enter a few characters, and many were able to enter their name. Initial text entry speeds were slow, but this generally did not appear to dampen their interest.

We made several observations in our own use of the device. We tended to tilt the device well beyond the thresholds to avoid ambiguities at the boundary. A few degrees of difference in the thresholds therefore did not seem to affect our performance. We also noticed that few errors arose from accidentally entering extreme characters. The large difference in tilt angle between the extreme and normal characters made this unlikely.

We found that we committed many symmetry errors in which the device was either tilted in the wrong direction, or tilted properly while the wrong button was pressed. With practice these errors seemed to occur less often.

The "tentative character" visual feedback helped us learn the character mapping, but was difficult to see at some tilt angles. Not surprisingly, we found that memorizing the character positions boosted our text entry speeds. Using only kinesthetic and haptic feedback, we could then enter text without having to look at the screen. We still needed to check for errors, but several characters could be checked at the same time when the device was examined at a normal viewing angle.

We did not feel any fatigue from using TiltType. The lack of fatigue may be due to its small mass and the small range of muscle movements. The buttons on the prototype require only one newton of operating force, which makes them more comfortable than conventional watch buttons. As can happen with conventional keyboards, overuse of TiltType might cause Repetitive Stress Injury (RSI), although we do not have the long-term data necessary to speculate on the nature or probability of injury.

IMPLEMENTATION

TiltType measures 4.0 cm x 3.7 cm x 1.4 cm. The display is anchored to a printed circuit board that also houses an ADXL202E MEMS accelerometer, a PIC16F877 microcontroller, a potentiometer for adjusting the screen contrast, four buttons, and a connector for serial communication to a PC or PDA. TiltType is powered by two 3V lithium coin cells. The display, an Optrex DMC-50448N character LCD, was chosen for its small size and simple interface. Despite having only eight columns and two lines, it worked well for our conceptual evaluation. All these parts are available from Digikey except the accelerometer, which can be obtained directly from Analog Devices.

The microprocessor reads the accelerometer about twenty times a second. The samples are not averaged, but capacitors on the accelerometer filter out noise above 100 Hz.

The accelerometer consumes 1.6mW with a 33% duty-cycle. A more sophisticated design could further reduce its awake time for greater power savings.

RELATED WORK

TiltType was inspired by Unigesture [11], another text-entry device that uses an accelerometer. Unigesture's goal is to enable one-handed text-entry on PDA or cellphone-sized devices. Unigesture does not use buttons. Instead, a tilting gesture is mapped to a group of characters, and a sequence of gestures is mapped to a specific word by T9-style dictionary lookup. Unigesture was inspired by non-text-entry accelerometer-based user interfaces developed by Rekimoto [10], Harrison et al. [3], and Levin and Yarin [7].

Bitman [6] is a toy that also uses tilt for text-entry. However, it uses a simpler, navigate/select interaction technique. One tilt dimension is used to scroll linearly through the alphabet; the other is used to switch between upper and lower case.

Other kinds of wearable keyboards have been developed for mobile users. The Twiddler is popular in the wearable computing community. It uses one-handed chording combinations of twelve buttons to enter characters. However, it is about the size of a stapler, which is too large for many users. L3's WristPC keyboard provides a full QWERTY keyboard on a forearm, but is larger still.

Some systems [2, 4] allow "virtually typing." The user makes typing motions onto any surface, and the system determines what keys would have been pressed had there been a keyboard present. These systems are bulkier than watches and do not include their own displays.

Some PDA touch-screen techniques, such as Graffiti™ and Quikwriting [9], could also work on a watch. Some adjustments would be necessary, as there is little room to carry even a shrunken stylus, and using one's finger would block the display. An interface that combined a touch-sensitive display with an accelerometer and buttons would be more expressive than TiltType, and might enable new interaction techniques.

In another PDA input technique, Dasher [13], users choose a path through a scrolling field of characters whose target sizes are adjusted according to the probability of that character being entered next. It would be interesting to see if an accelerometer could be used as a control input for Dasher.

Other forms of text entry could be adapted for very small devices. Morse code is feasible, although its learning curve intimidates many users. A few devices (such as the Samsung Watch Phone) use speech recognition, although speech is sensitive to noise and non-private. Speech recognition also requires a relatively large amount of processing power, which takes its toll in either battery size or battery lifetime.

As with other techniques, TiltType could benefit from macros and word completion, such as developed by POBox [8].

FUTURE WORK

Important metrics in the evaluation of any text-entry system are typing speed, error rates, and learning times. We plan to conduct user studies to measure these quantities. Data from these studies will also be useful in constructing a model of timings for moving between various positions. For example, pressing different buttons while in the same tilt position is probably faster than pressing buttons while changing tilt positions. Changing tilt angle along one axis may be faster than changing it along two. Using this model, a more optimal keyboard mapping can be designed that maps more common English letter sequences onto more rapidly typed TiltType positions.

A trade-off exists between the number of buttons and the number of tilt positions. We chose four buttons because many watches have four buttons, and because this design fit the number of characters in the English language. But using more buttons and fewer tilt angles might lead to faster text entry or lower error rates. We also chose two concentric zones, the normal positions and the extreme positions, but more concentric zones may also be possible. A general study modeled after those carried out for marking menus [5] would help clarify these tradeoffs.

ACKNOWLEDGMENTS

We thank our paper shepherd Ken Hinckley and the anonymous reviewers for their helpful and detailed suggestions. We also thank James Landay, Ken Fishkin, Anthony LaMarca, and David Koizumi for their ideas and comments.

REFERENCES

1. W. Buxton, R. Hill, and P. Rowley. Issues and techniques in touch-sensitive tablet input. *Computer Graphics*, 19(3):215–224, 1985.

2. M. Fukumoto and Y. Tonomura. Body coupled FingeRing: Wireless wearable keyboard. In *Conference Proceedings on Human Factors in Computing Systems*, pages 147–154. ACM Press, 1997.

3. 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. In *Conference proceedings on Human Factors in computing systems*, pages 17–24. ACM Press/Addison-Wesley Publishing Co., 1998.

4. B. Howard and S. Howard. Lightglove: Wrist-worn virtual typing and pointing. In *Proc. 5th IEEE International Symposium on Wearable Computers*, pages 172–173, 2001.

5. G. Kurtenbach, A. Sellen, and W. Buxton. An emprical evaluation of some articulatory and cognitive aspects of 'marking menus'. *Journal of Human Computer Interaction*, 8(1), 1993.

6. R. Kuwakubo. Bitman. http://www.vector-scan.com.

7. G. Levin and P. Yarin. Bringing sketching tools to keychain computers with an acceleration-based interface. In *Proc. of ACM SIGCHI 99*, 1998.

8. T. Masui. POBox: An efficient text input method for handheld and ubiquitous computers. In *Proceedings of the International Symposium on Handheld and Ubiquitous Computing (HUC'99)*, pages 289–300, 1999.

9. K. Perlin. Quikwriting: continuous stylus-based text entry. In *Proceedings of the 11th annual ACM Symposium on User Interface Software and Technology*, pages 215–216. ACM Press, 1998.

10. J. Rekimoto. Tilting operations for small screen interfaces. In *Proceedings of the 9th annual ACM UIST*, pages 167–168. ACM Press, 1996.

11. V. Sazawal, R. Want, and G. Borriello. The unigesture approach: One-handed text entry for small devices. In *MobileHCI*. to appear, 2002.

12. A. Sears, D. Revis, J. Swatski, R. Crittenden, and B. Shneiderman. Investigating touchscreen typing: The effect of keyboard size on typing speed. *Behavior & Information Technology*, 12(1):17–22, 1993.

13. D. J. Ward, A. F. Blackwell, and D. J. C. MacKay. Dasher—a data entry interface using continuous gestures and language models. In *Proceedings of the 13th annual ACM UIST*, pages 129–137. ACM Press, 2000.