DriftBoard: A Panning-Based Text Entry Technique for Ultra-Small Touchscreens

Tomoki Shibata[1], Daniel Afergan[1,2], Danielle Kong[1], Beste F. Yuksel[1,3], I. Scott MacKenzie[4], Robert J.K. Jacob[1]

[1]Tufts University
Medford, MA, USA
{tshibata, dkong02, jacob}@cs.tufts.edu

[2]Google Inc.
Mountain View, CA, USA
afergan@google.com

[3]University of San Francisco
San Francisco, CA
byuksel@usfca.edu

[4]York University
Toronto, ON, Canada
mack@cse.yorku.ca

ABSTRACT

Emerging ultra-small wearables like smartwatches pose a design challenge for touch-based text entry. This is due to the "fat-finger problem," wherein users struggle to select elements much smaller than their fingers. To address this challenge, we developed *DriftBoard*, a panning-based text entry technique where the user types by positioning a movable qwerty keyboard on an interactive area with respect to a fixed cursor point. In this paper, we describe the design and implementation of DriftBoard and report results of a user study on a watch-size touchscreen. The study compared DriftBoard to two ultra-small keyboards, ZoomBoard (tapping-based) and Swipeboard (swiping-based). DriftBoard performed comparably (no significant difference) to ZoomBoard in the major metrics of text entry speed and error rate, and outperformed Swipeboard, which suggests that panning-based typing is a promising input method for text entry on ultra-small touchscreens.

Author Keywords

Soft keyboard; text entry; ultra-small touchscreens; fat-finger problem; fixed cursor and movable keyboard; DriftBoard

ACM Classification Keywords

H.5.2 User Interfaces: Input devices and strategies, Interaction styles

INTRODUCTION

User input on ultra-small wearables is more limited than output. For example, smartwatches offer instant access to notification messages, but responsive input actions pose a challenge. Notably, text entry with a conventional software keyboard on a smartwatch suffers due to the fat-finger problem [24], where finger width exacerbates precise selection of tiny character keys. This limits human communication using wearable technology such as a smartwatch.

Several approaches that manipulate the keyboard display or introduce new gesture sequences have shown promise against this challenge. For instance, ZoomBoard [20] employs a tapping-based approach where tap actions zoom in to the keyboard, thus making the desired key large enough to touch. Swipeboard [5] employs a swiping-based approach where each character is associated with a unique sequence of swipes. Therefore, the user does not need to hit an exact location on the screen.

ZoomBoard and Swipeboard compensate for the small form factor by requiring users to make multiple taps per character or learn new gesture sequences. This may discourage users from adopting these innovative technologies. Further, other kinds of touch-based interactions are available as potential input methods for text entry on ultra-small devices.

Our goal in this research is twofold: (1) design an alternative form of touch-based input which overcomes the fat-finger problem, and (2) validate its potential for text entry on ultra-small touchscreens.

To accomplish this, we developed *DriftBoard*, a panning-based text entry technique using a fixed cursor point and a movable qwerty keyboard. To type with DriftBoard, the user performs a panning action: dragging on the interactive area to position the movable qwerty keyboard toward the fixed cursor point, and then releasing the finger from the touchscreen to enter the character currently underneath the fixed cursor.

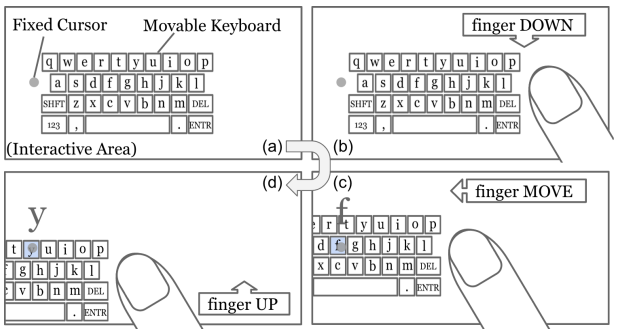

Figure 1. DriftBoard consists of three key components: (a) a fixed cursor, a movable keyboard, and an interactive area. To type, the user (b) places a finger down on the interactive area, (c) moves the finger to place the desired key on the movable keyboard under the fixed cursor, and then (d) releases the finger up from the interactive area. In the figures, "y" is entered. Note that the keyboard can extend outside the interactive area and that key highlighting and enlargement are provided for the character underneath the fixed cursor.

UIST 2016, October 16–19, 2016, Tokyo, Japan.

ACM 978-1-4503-4189-9/16/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2984511.2984591

We also report results of a user study with 10 participants performing text copy tasks utilizing DriftBoard (panning-based) on a smartwatch size touchscreen (28 $\times$ 14 mm). Two points of comparison were ZoomBoard [20] (tapping-based) and Swipeboard [5] (swiping-based), replicated on a screen of the same size.

We measured the major performance metrics of text entry speed and error rate, along with efficiency defined with keystrokes per character (KSPC) [25]. Additionally, task workloads were assessed with NASA-TLX [10].

RELATED WORK

There is considerable work in HCI on the design of on-screen visual keyboards. This work includes visual keyboard optimization, for example, OPTI [19], Metropolis [32] and ATOMIK [33], and gestural input methods such as Dasher [29] and SHARK2 [13] (shape writing).

Additionally, text entry research faces new challenges as devices become smaller than full-sized physical keyboards. There is research addressing this issue; for example, (i) Drag-and-Type [14] includes a cursor manipulated by a finger on a miniature qwerty soft keyboard, (ii) one-key keyboard [12] reduces the number of keys on a screen, and (iii) H4-Writer [18] employs a Huffman code generation algorithm to map minimum key sequences to characters, reducing the number of selectable elements on a screen to four items.

There also exists several established text entry techniques for portable use. For example, Twiddler [16] is a chording keyboard with a 3 $\times$ 4 array of physical keys, in which one or two keys are pressed simultaneously to enter a character. With EdgeWrite [30], the user draws a unique pattern by traveling through four corners of a square with a stylus, and each pattern maps to a character of input. Another method, QuikWriting [22], requires the user to make continuous stylus movements among groups of characters arranged in regions around the perimeter of a square.

Text Entry on Ultra-Small Devices

Text entry is also of interest in wearable computing. In this context, form factors are often far smaller than those in mobile computing. Not limited to text entry, Skinput [9] uses the skin for input. Other techniques leveraging additional sensing for input are Back-of-device [2], TiltType [21], DualKey [8], NanoStylus [31] and so on. Further, WatchWriter [6] demonstrates the shape writing approach within ultra-small scale.

Next we present recent research addressing the fat-finger problem on text entry using ultra-small touchscreen devices.

Zooming and Tapping Approach

Zooming into a keyboard is one approach to mitigating the fat-finger problem. Character keys are enlarged until the size is comfortable for tapping. One example, ZoomBoard [20], enlarges a qwerty keyboard at one level per tap action. SplitBoard [11] splits a qwerty keyboard into flickable pieces, so that each character key can occupy more space on the screen. The Virtual Sliding Qwerty [4] utilizes a qwerty keyboard larger than the screen with the user sliding the keyboard to make the desired key appear. One common trait of these methods is the need for multiple discrete actions to type a character.

Swipe Gesturing Approach

Typing with swipe gestures has the desirable property of not requiring the user to pinpoint an exact location on the screen. Thus, the fat-finger problem is averted. An example is Swipeboard [5], which employs a qwerty layout navigable by a two-stroke swipe gesture to enter a character. This eye-free technique is applicable to smart eyewear [7]. Nevertheless, the downside of swipe gesture methods is that users must learn the gesture patterns required for input.

Pointer Shifting Approach

Users gain more control on selecting character keys when allowed to refine their finger position before committing to a selection. For example, ZShift [15] applies a "Shift pointing technique" [28] with a qwerty keyboard. The user's touch point creates a callout showing the occluded area under the finger. The callout includes a one-level zoom of the touched area, highlighting the selected character. There are two disadvantages: Refining a finger location takes longer than an immediate direct tap, and the callout consumes and occludes pixels beyond the keyboard area.

Other Approaches

Reducing the number of keys is also applicable for wearable devices as users can tap relatively larger elements compared to tiny character keys. However, departing from the familiar qwerty layout increases learning time and may hinder adoption. Utilizing an external device for input is also applicable; however, this may nullify the benefits of wearability.

Lastly, Table 1 shows a summary of research on ultra-small keyboards using touch-based interactions with no external equipment. The table and also the survey [1] provide insight into the range of potential text entry metrics on ultra-small touchscreens.

DRIFTBOARD

In this section, we introduce *DriftBoard*, a panning-based text entry technique combining a fixed cursor point and a movable qwerty keyboard. In mobile computing contexts, dragging a finger on a touchscreen to navigate a pointer has been explored to resolve the fat-finger problem [28, 23]. Further, "Fix and Slide" [26], a caret navigation technique, introduces dragging to navigate a movable background to locate a caret, which is fixed by a preceding tap, at a desired in-text position.

Our work focuses on the domain of text entry on ultra-small touchscreens. DriftBoard combines three key components: a fixed cursor, a movable keyboard, and an interactive area, plus panning with automatic character selection on finger release. The underlying idea is that the user performs panning on the interactive area to position the movable keyboard at the proper location, wherein the fixed cursor acquires the desired key.

Figure 1 illustrates the interaction on DriftBoard with a panning action. Figure 2 demonstrates a user typing with DriftBoard on an Android smartwatch with a round display.

Technique	Approach	Execution	Size (mm) Keyboard (Interactive area)	Phrase set	Performance Speed(wpm) (Error(%))
ZoomBoard (Oney et al., 2013) [20]	Zoom	Two step tap	16.5 × 6.1 (-)	MacKenzie & Soukoreff [17]	9.3 (-)
Swipeboard (Chen et al., 2014) [5]	Gesture	Two step swipe	12 × 12 (-)	Own word set	19.58 (13.30)[1]
ZShift (Leiva et al., 2015) [15]	Pointer shift	Finger drag and release	S) 16.0 × 6.5 (18 × 18) M) 21.3 × 8.6 (24 × 24) L) 28.4 × 11.4 (32 × 32)	MacKenzie & Soukoreff [17]	S) 5.4 (1.3) M) 7.2 (1.3) L) 9.1 (0.9)
SplitBoard (Hong et al., 2015) [11]	Zoom	Flick and tap	- (29.3 × 29.3)[2]	MacKenzie & Soukoreff [17]	A) 15.2 (0.58)[3] B) 16.3 (0.4)[3]
Virtual Sliding QWERTY (Cha et al., 2015) [4]	Zoom	Slide and tap	40.0 × 21.0 to 80.0 × 41.0 (29 × 22) × 4 different "gain"[4]	302 phrases from MacKenzie & Soukoreff [17]	Best) 13.2 (-) Ave.) 11.9 (-) Worst) 10.1 (-)
WatchWriter (Gordon et al., 2016) [6]	Gesture	Shape writing	- (1.3-inch circular)	MacKenzie & Soukoreff [17]	Mean) 24.0 (3.7)

[1]"Hard error rate" [5]; [2]The size includes spaces for a presented phase and a text to be entered.; [3]Average uncorrected error rate; [4]"speed of navigation" [4]

Table 1. A summary of published ultra-small keyboard research.

A panning action requires three primitive finger events: down, move, and up. Within one panning action, the finger-down event initiates panning as the user's finger meets the interactive area. A series of finger-move events continuously translates the location of the movable keyboard. A finger-up event, in which the user removes their finger from contact with the interactive area, completes the panning action and selects the character acquired by the fixed cursor. If no character key is acquired by the fixed cursor on finger-up, no character is entered.

We believe our technique addresses the fat-finger problem for two reasons: (1) Unlike tapping but like swiping, panning does not need to be initiated at an exact pixel location and the finger does not even need to be on the keyboard area. Plus, the position of the fixed cursor is designed to avoid a finger going over it. (See our Design Choices below.) Thus, it resolves the finger occlusion problem. (2) Character selection occurs only at the time of finger release, which allows a user to refine the finger location before releasing. Thus, it resolves the finger precision problem.

On the other hand, DriftBoard is still distinct from ZShift [15] in terms of the movement strategies for the target and acquisition point. Whereas the finger pointer moves in ZShift, the keyboard moves in DriftBoard. Furthermore, DriftBoard and Visual Sliding Qwerty keyboard (VSQ) [4] differ in their use of finger events. VSQ employs a slide and tap; DriftBoard uses panning with automatic character selection on finger-up (if there is a character at the cursor location).

Design Choices

Considering the performance costs and benefits of the techniques discussed in the previous section, we made the following design choices when implementing DriftBoard:

- We focused only on the touch-based interactions available on touchscreens, so users can type with no external equipment.
- We used a qwerty layout so users leverage existing skills.
- We adopted a panning method that allows users to enter one character with one input action. (Note: "clutching" is required only rarely to type a character.)
- We set the size of the keyboard so each key is reachable from any other key with one panning action.
- We placed the fixed cursor on the left of the interactive area to avoid finger occlusion around the fixed cursor for right-handed users. (Note: The cursor location is configurable and can be placed on the right for left-handed users.)
- We added visual feedback (key highlighting and enlargement) for the character underneath the fixed cursor. However, this feedback does not overlap with the text the user is composing. This allows users to follow their typing progress on the same screen (see Procedure).

Figure 2. DriftBoard on a Motorola Moto 360 smartwatch with a round display. Left: The fixed cursor is on top of "r". Right: The cursor remains fixed, but acquires "z" due to keyboard movement. A finger-up at this time enters "z". Note that the size of the movable keyboard is not necessarily the same as the device's display. Note also that DriftBoard is applicable to non-rectangular shapes.

Preliminary Study

To investigate the feasibility of panning-based input for text entry, we conducted a preliminary user study with 13 participants (9 females), all right-handed native English speakers with a mean age of 21.

The DriftBoard prototype was installed on an Android Nexus 7 (2013) touchscreen device. The measured sizes (mm) of the interface elements were 28 × 14 (interactive area); 19 × 8 (keyboard); and 1.7 × 1.8 (key).

The participants transcribed 5 phrases per block and 15 blocks total. The phrases were lowercase and randomly selected from MacKenzie and Soukoreff's phrase set [17]. They were instructed to use the index finger of the right hand and to complete the task as quickly and accurately as possible. The participants watched the experimenter enter several characters on DriftBoard, and then started their trials without any training.

The preliminary user study demonstrated an initial average text entry speed of 6.54 words per minute (wpm), with an error rate of 3.80%. After about one hour, users reached 10.07 wpm and an error rate of 2.98%. Based on these promising results, we conducted a more comprehensive user study, the main experiment in this paper, as detailed in the next section.

METHOD

The purpose of the main user study was to evaluate the potential of our panning-based text input method for ultra-small touchscreens. The study compared DriftBoard (panning-based) with two alternative techniques for ultra-small touchscreens: ZoomBoard [20] (tapping-based) and Swipeboard [5] (swiping-based). All three methods were implemented on an interactive area of the same size, 28 × 14 mm.

Recall that ZoomBoard maps two consecutive tap actions on a qwerty layout to one character, where the first tap invokes a one-level zoom on the keyboard and the second tap selects the character the finger landed on. Swipeboard maps two consecutive swipe actions to a character, where the first swipe determines one of nine character groups and the second swipe selects a character within the group.

Participants

We recruited 10 participants (5 female) of ages between 19 and 33 (mean 22.8, SD 4.2) by posting flyers on a university campus. The participants did not participate in the preliminary study. All were right-handed, native English speakers or bilingual from birth and have normal or corrected-to-normal vision. All were paid 10 U.S. dollars per hour for participating, in addition to a performance bonus of 5 U.S. dollars (as a form of motivation). Testing proceeded over two sessions of 90-120 minutes each (one per day) and was conducted in a quiet office or small conference room.

Apparatus

The three input techniques were implemented as Android applications. We used a Nexus 7 (2013) tablet for accurate logging and covered the unused surface area to offer a watch-size feeling. Touch actions outside of the interactive area were ignored. To increase internal validity of this study, the device was placed on a table consistently across all conditions while participants performed the task.

Figure 3 shows the physical size of the interactive area. Figure 4 shows the implementation of each input technique. The measured sizes are given in Table 2. The interactive area of 28 × 14 mm was fixed for all input techniques and was roughly half the 1.6-inch display diagonal. This size fits within the constraints of the smallest smartwatch face available at the present time.

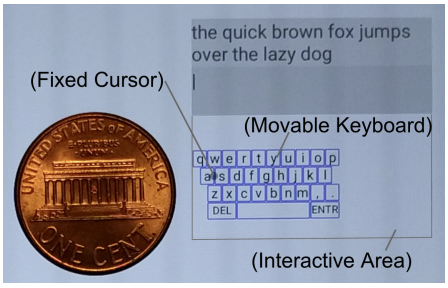

Figure 3. DriftBoard prototype on an Android touchscreen device with a U.S. one cent coin as a reference. In the user study, participants entered the phrase appearing on top, with characters appearing below the phrase.

Technique	Interactive area		Keyboard		Key	
	w	h	w	h	w	h
DriftBoard	28	14	18	8	1.8	2
ZoomBoard	28	14	23	9.5	2	2
(zoomed)					6	6
Swipeboard	28	14	21.5	12	7.5*	3.5*
(2nd layer)					7	8

*3-character block

Table 2. The measured size in (mm).

Some additional implementation details (as replication notes) are now given. Our main objective was to study and compare the core methods for text entry: panning, tapping, and swiping.

- ZoomBoard: Swipe to type a Space character and swipe to delete [20] were not implemented. Instead, the Space, Delete, and Enter keys were visually presented. The zoom-in scaling factor was 3× (approximately) [20].
- Swipeboard: Enter was assigned a double swipe-down action and was not visually presented on the screen. This is similar to Space and Delete in the original implementation [5].
- ZoomBoard and Swipeboard: "2-second timeout" [5] was implemented.
- DriftBoard: The location of the movable keyboard was reset to the position in Figure 3 when a new phrase appeared. This ensures that the very first character of any character keys is reachable with one panning action.

Procedure

The experiment consisted of two sessions (one per day) for each participant with 3 input techniques per session, 5 blocks per input technique and 5 phrases per block. All phrases (lowercase only, no punctuation) were randomly picked from MacKenzie and Soukoreff's phrase set [17], a representative set of English phrases. Participants were instructed (1) to use the index finger of their right hand, (2) to memorize each phrase before beginning (although the presented phrase did not disappear), (3) to correct errors only if they realized immediately that an error occurred, and (4) to proceed as quickly

and accurately as possible. In the first session, at the beginning of each input technique, participants watched the experimenter enter several characters with verbal instructions. Then, testing began without practice. In the second session, prior to testing each technique, participants were asked to enter "hello world" and to delete a couple of characters to refresh their memory.

Informed consent was obtained from each participant at the beginning of each session. After that, the participants completed a brief demographic questionnaire in the first session. The participants also completed the NASA-TLX questionnaire after each input technique in both sessions.

The participants were allowed to take a short break and allowed to wipe the device screen between phrases and between blocks. The experimenter wiped the device screen after each block to reduce finger residue on the screen.

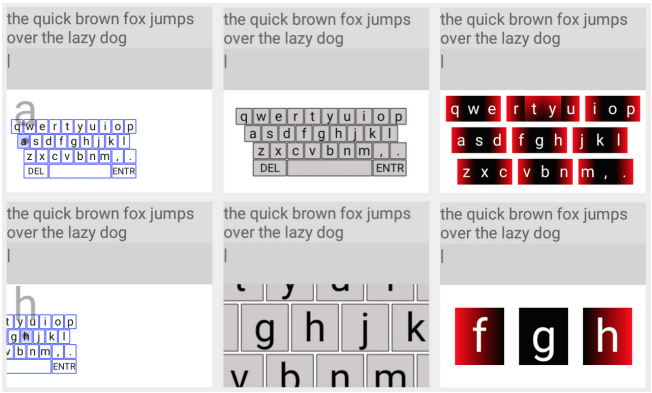

Figure 4. Input techniques tested. Left-Top: The fixed cursor of DriftBoard is acquiring "a", and Left-Bottom is acquiring "h". Center-Top: The initial state of ZoomBoard, and Center-Bottom a zoomed state by a tap near "h". Right-Top: The initial state of Swipeboard showing all character blocks, and Right-Bottom the state showing the "fgh" block.

Design

We used a within-subjects design with two independent variables: input technique (3) and block (10). We used block as an independent variable to capture participants' improvement with practice. The dependent variables were text entry speed (words per minute), error rate (%) and efficiency (%) calculated by keystrokes per character (KSPC). In addition, the responses to NASA-TLX by input technique served as a dependent variable.

Text entry speed was calculated by "dividing the length of the transcribed text by the entry time (in seconds), then multiplying by sixty (seconds in a minute) and dividing by five (the accepted word length, including spaces)" [3]. Entry time was measured from the first to last finger event in the interactive area, except for inputting the Enter key, which was used to complete entry of a phrase.

Error rate was calculated using the minimum string distance (MSD) method [25]. This measure reflects the number of character errors given the total number of characters entered.

Efficiency (using KSPC) was calculated as the minimum keystrokes required for the presented phrase divided by the actual keystrokes and then multiplied by 100.

In order to offset learning effects across the input techniques, each of six possible orders of the three input techniques was assigned to one or two participants (of different gender). The order for the first and second sessions was the same for each participant.

RESULTS

We now report analyses on 1,455 phrases out of the 1,500 total collected (i.e., 10 participants $\times$ 2 sessions $\times$ 3 techniques $\times$ 5 blocks $\times$ 5 phrases). We excluded 7 erroneous phrases, due to hardware issues, and 38 outliers, which were phrases with an error rate greater than 50% or efficiency less than 50%. All outliers were found in the Swipeboard condition and, evidently, were due to a participant accidentally inputting the Enter key. We believe classifying these as outliers and removing them makes our comparison fair across the three input techniques.

Figure 5 shows averaged performance of the text copy task, where the first block of the second session was denoted as the 6th block. In summary, the fastest observed entry speed for each technique was as follows: DriftBoard: 9.74 wpm with error rate 0.60% and efficiency 93.6% at the 10th block; ZoomBoard: 10.0 wpm with error 1.35% and efficiency 92.2% at 7th block; and Swipeboard: 8.52 wpm with error 2.46% and efficiency 91.1% at 8th block. The ratio of the entry speeds was DriftBoard : ZoomBoard : Swipeboard = 100 : 103 : 87.5. We next report detailed results for each dependent variable.

Text Entry Speed

The grand mean for entry speed was 8.34 wpm. ZoomBoard at 9.23 wpm was the fastest, followed by DriftBoard of 8.77 wpm and then Swipeboard of 7.11 wpm. However, a modest difference existed between DriftBoard (9.74 wpm) and ZoomBoard (9.94 wpm) by the time the tenth block was reached, whereas differences involving Swipeboard (8.14 wpm) were still observed. Using an ANOVA, the main effect of input method was statistically significant ($F_{2,18} = 30.127, p < .0001$). Post hoc comparisons using Fisher LSD on input method revealed no significant difference between ZoomBoard and DriftBoard, and significant differences between Swipeboard and the other two input methods.

Not surprisingly, the ANOVA for entry speed found a significant effect of block ($F_{9,81} = 36.617, p < .0001$). This confirms that there is a pronounced effect of learning. See Figure 5. The mean entry speed for the first block was 6.41 wpm and for the last block, 9.27 wpm. Lastly, there was also a significant interaction effect ($F_{18,162} = 1.746, p < .05$).

Error Rate

The grand mean for error rate was 1.74%. DriftBoard at 1.13% was the lowest, followed by ZoomBoard at 1.56% and then by Swipeboard at 2.51%. At the tenth block, DriftBoard was still the most accurate with a mean error rate of 0.60%, followed by ZoomBoard at 1.08% and then Swipeboard at 1.48%. As shown in Figure 5, the error rate of each input technique followed a slightly declining trend and was consistently below 3% for DriftBoard and ZoomBoard and below 4% for Swipeboard. With an ANOVA, the main effect of input method was statistically significant ($F_{2,18} =$

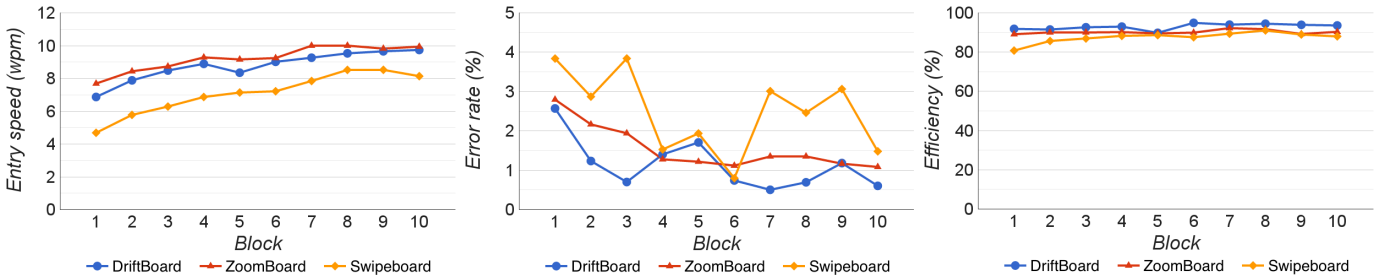

Figure 5. Text entry performance measurements for the three input techniques.

$5.553, p < .05$). Post hoc comparisons using Fisher LSD on input method revealed no significant difference involving ZoomBoard, but a significant difference between Swipeboard and DriftBoard.

On error rate, the ANOVA indicated no significant effects of block ($F_{9,81} = 1.799, p > .05$) nor block by input technique ($F_{18,162} = .760, ns$). This was likely because the participants were instructed to correct errors through the entire session, as detailed in the Procedure section. It should be noted that correcting errors takes a toll on entry speed and efficiency.

Efficiency

Efficiency in terms of KSPC takes error corrections into account. For example, inputting a character and then deleting it produces extra keystrokes. This applies to all three input techniques. In other cases, recall that the both ZoomBoard and Swipeboard require two consecutive actions to select a character. Therefore, where a user made the first action but not the second before the timeout, the effect was to decrease efficiency. For DriftBoard, if a user completed a panning action but the fixed cursor did not acquire a character key, efficiency was also decreased.

The grand mean for efficiency was 90.12%. DriftBoard was more efficient at 92.97% overall, compared to ZoomBoard 90.27% and Swipeboard 87.48%. At the tenth block, we observed 93.59% for DriftBoard, 90.34% for ZoomBoard and 88.00% for Swipeboard. With an ANOVA, the main effect of input method was statistically significant ($F_{2,18} = 6.815, p < .01$). Post hoc comparisons using Fisher LSD on input method revealed no significant difference involving ZoomBoard and a significant difference between Swipeboard and DriftBoard.

Regarding efficiency, the ANOVA indicated no significant effects of block ($F_{9,81} = 1.993, p > .05$) (although $p = .0506$) and of interaction ($F_{18,162} = 1.488, p > .05$). If we consider the block effect ($p = .0506$) to be weakly significant, it could supplement the pronounced effect of learning which was observed in entry speed.

Task Workload

Table 3 shows mean responses of task workloads assessed with NASA-TLX. We report the responses from the second sessions, as we think they best describe participants' total experiences. A Friedman test was used to test for significant differences on the responses to the six NASA-TLX categories. No significant differences were found for mental demand, physical demand, temporal demand and performance. Significant differences were found for effort and frustration, with the responses indicating more frustration for Swipeboard. On the other hand, no significant difference on frustration was reported between DriftBoard and ZoomBoard. This may be due to a nature of gesture methods that require users to learn gesture sequences.

	Technique				
NASA-TLX	**D**	**Z**	**S**	**Friedman *H* (*p*)**	**Post Hoc**
Mental demand	6.8	5.9	9.8	1.72 (.427)	
Physical demand	5.5	4.6	6.8	2.23 (.328)	
Temporal demand	8.8	12.1	11.5	0.72 (.698)	
Performance	15.7	13.8	13.1	3.30 (.192)	
Effort	9.7	10.1	13.1	6.70 (.035)	Z-S
Frustration	6.9	6.6	12.2	7.39 (.025)	D-S, Z-S

Note: D = DriftBoard, Z = ZoomBoard, S = Swipeboard
(Response range: 0-20; Lower is better, except for Performance.)

Table 3. Mean responses of task workload index.

DISCUSSION AND FUTURE WORK

Overall, the user study revealed that DriftBoard and ZoomBoard were analogous (no significant differences) on text entry speed, error rate, and efficiency. This dispels the notion that users must directly tap a key with a finger to use a touchscreen keyboard effectively. Furthermore, the text entry performance of DriftBoard appeared significantly better than that of Swipeboard. In addition, Swipeboard was reported to induce more frustration than the other two methods. Frustration along with performance would be a serious detriment for user acceptance. We therefore posit that DriftBoard and ZoomBoard could provide a faster path to user acceptance than Swipeboard, especially for first-time users.

In our user study, ZoomBoard and Swipeboard did not reach text entry speeds in the 17-20 wpm range, which was reported in the original Swipeboard study [5]. As Gupta et al. [8] also note, we think this was mainly due to a combination of differences in the phrase sets used and differences in the goals of the user studies; the original Swipeboard study used the authors' own reduced word set to mitigate the impact of learning [5]. To further support this point, the original ZoomBoard study, which used the same phrase set we used, reported an entry speed of 9.3 wpm [20], which is in the range of our user study result.

Limitations

There are limitations in our user study that suggest improvements in the future. (1) Evaluating performance for typing

uppercase and symbol characters is necessary for more general text entry. In our user study, we simplified the keyboard shown in Figure 1 by removing the "SHIFT" and "123" keys to increase the internal validity. (2) Evaluating the technique with the device physically worn on a body is necessary, since a user normally wears the device and moves freely in the real world. (3) Performance differences in a broader population should also be considered, as wearable users are not limited to university students, who made up our experimental pool. (4) Evaluating expert performance in a single study is important to properly compare the input techniques of interest. We believe addressing these existing limitations in future work will increase external validity of the introduced technique.

Improving Text Entry Speed

Improving existing ultra-small keyboards is a potential future research area. Using our panning approach, we now propose several alternatives to improve text entry speed for more experienced users. For example, we could employ a special method for entering "space" (the most frequent character) by assigning it to the entire area beyond the keyboard. Further, continuous typing with panning could improve overall text entry speed. For example, DriftBoard combined with shape writing method [13] eliminates finger down-up actions within a word, as required in the current implementation of DriftBoard. Similarly, pressure-sensitive touchscreens could further streamline input by replacing finger up with finger press, so that users could continuously type without releasing their finger from the touchscreen.

Along with different types of finger input, incorporating a language model to enable word predictions and error corrections is another key to improving text entry performance. We believe this would be promising, based on evidence with two techniques which showed significantly improved text entry compared to a miniature qwerty keyboard: WatchWriter [6], a gesture typing method (shape writing) with a statistical decoder built on a language model, and VelociTap [27], with a touch-event-sequence decoder built on a combination of a probabilistic keyboard model and language models. However, developing mechanisms for presenting and responding to word predictions and error corrections on an ultra-small interface presents additional challenges.

Implications on A Fixed Cursor and Movable Keyboard

Our user study focused on comparing methods of finger input. We now call attention to the role of visual elements in DriftBoard, beyond our user study configuration. Recall that we flipped the movement strategies of the acquisition point and targets, compared to the conventional tap input method. Figure 6 illustrates implications of the DriftBoard concept and exposes questions for future work:

(a) ***The fixed target acquisition point*****:** The target acquisition point can be as small as a single pixel, therefore a keyboard even smaller is still feasible. Q: What configuration of keyboard size and location of the fixed cursor would work best?

(b) ***Movable targets*****:** The targets are movable, therefore a keyboard even larger than the overall screen may be feasible and may allow uses to type using a "gain" [4] factor between finger movement and keyboard movement. Q: What configuration of keyboard size and gain control would work best?

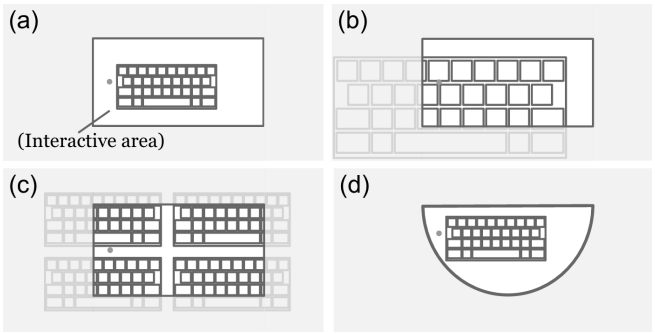

Figure 6. Implications of the DriftBoard concept. (Gray area is not visible to users.)

(c) ***The interactive area*****:** The interactive area can be a small subset of a much larger space of targets, therefore different keyboard configurations can be used for lowercase, uppercase, punctuation, and special symbols. Q: How is performance affected by using duplicated keyboards versus a single mode-switching keyboard?

(d) ***Shape of the interactive area*****:** The interactive area need not be a rectangle, therefore a round interactive area paired with a rectangular keyboard is feasible. Q: What kinds of shape pairing would work best?

CONCLUSION

Text entry remains a significant challenge for the emerging generation of ultra-small wearables because of the incompatibility of tiny user interface elements with normal-sized human fingers. We have introduced a panning-based text entry mechanism, using a fixed cursor and a movable keyboard, which overcomes the fat-finger problem for text entry on ultra-small touchscreens. Our user study demonstrated that the text entry performance of DriftBoard (panning-based) was comparable (no significant difference) to ZoomBoard (tapping-based) and superior to Swipeboard (swiping-based). Finally, we discussed the limitations of our user study and the implications of the fixed cursor and movable keyboard concept for future work. We hope our proposed panning-based typing contributes to improving input mechanisms on current wearables, expands the design space for such devices, and leads to unlocking the full potential of human-wearable communication.

ACKNOWLEDGMENTS

We would like to thank Remco Chang, Ronna ten Brink, Anzu Hakone, Joshua Lee from Tufts University and NSF (IIS-1218170 and IIS-1065154) for their support.

REFERENCES

1. Arif, A. S., and Mazalek, A. A survey of text entry techniques for smartwatches. In *Proc. HCI International '16*, Springer (2016), 255–267.

2. Baudisch, P., and Chu, G. Back-of-device interaction allows creating very small touch devices. In *Proc. CHI '09*, ACM (2009), 1923–1932.

3. Castellucci, S. J., and MacKenzie, I. S. Gathering text entry metrics on Android devices. In *Proc. CHI EA '11*, ACM (2011), 1507–1512.

4. Cha, J.-M., Choi, E., and Lim, J. Virtual sliding qwerty: A new text entry method for smartwatches using Tap-N-Drag. *Applied Ergonomics 51* (2015), 263–272.
5. Chen, X. A., Grossman, T., and Fitzmaurice, G. Swipeboard: A text entry technique for ultra-small interfaces that supports novice to expert transitions. In *Proc. UIST '14*, ACM (2014), 615–620.
6. Gordon, M., Ouyang, T., and Zhai, S. WatchWriter: Tap and gesture typing on a smartwatch miniature keyboard with statistical decoding. In *Proc. CHI '16*, ACM (2016), 3817–3821.
7. Grossman, T., Chen, X. A., and Fitzmaurice, G. Typing on glasses: Adapting text entry to smart eyewear. In *Proc. MobileHCI '15*, ACM (2015), 144–152.
8. Gupta, A., and Balakrishnan, R. DualKey: Miniature screen text entry via finger identification. In *Proc. CHI '16*, ACM (2016), 59–70.
9. Harrison, C., Tan, D., and Morris, D. Skinput: Appropriating the skin as an interactive canvas. *Commun. ACM 54*, 8 (2011), 111–118.
10. Hart, S. G. NASA-task load index (NASA-TLX); 20 years later. In *Proc. the human factors and ergonomics society annual meeting*, vol. 50, Sage Publications (2006), 904–908.
11. Hong, J., Heo, S., Isokoski, P., and Lee, G. SplitBoard: A simple split soft keyboard for wristwatch-sized touch screens. In *Proc. CHI '15*, ACM (2015), 1233–1236.
12. Kim, S., Sohn, M., Pak, J., and Lee, W. One-key keyboard: A very small qwerty keyboard supporting text entry for wearable computing. In *Proc. OZCHI '06*, ACM (2006), 305–308.
13. Kristensson, P.-O., and Zhai, S. SHARK2: A large vocabulary shorthand writing system for pen-based computers. In *Proc. UIST '04*, ACM (2004), 43–52.
14. Kwon, T., Na, S., and h. Park, S. Drag-and-type: A new method for typing with virtual keyboards on small touchscreens. *IEEE Transactions on Consumer Electronics 60*, 1 (2014), 99–106.
15. Leiva, L. A., Sahami, A., Catala, A., Henze, N., and Schmidt, A. Text entry on tiny qwerty soft keyboards. In *Proc. CHI '15*, ACM (2015), 669–678.
16. Lyons, K., Starner, T., Plaisted, D., Fusia, J., Lyons, A., Drew, A., and Looney, E. W. Twiddler typing: One-handed chording text entry for mobile phones. In *Proc. CHI '04*, ACM (2004), 671–678.
17. MacKenzie, I. S., and Soukoreff, R. W. Phrase sets for evaluating text entry techniques. In *Proc. CHI EA '03*, ACM (2003), 754–755.
18. MacKenzie, I. S., Soukoreff, R. W., and Helga, J. 1 thumb, 4 buttons, 20 words per minute: Design and evaluation of H4-Writer. In *Proc. UIST '11*, ACM (2011), 471–480.
19. MacKenzie, I. S., and Zhang, S. X. The design and evaluation of a high-performance soft keyboard. In *Proc. CHI '99*, ACM (1999), 25–31.
20. Oney, S., Harrison, C., Ogan, A., and Wiese, J. ZoomBoard: A diminutive qwerty soft keyboard using iterative zooming for ultra-small devices. In *Proc. CHI '13*, ACM (2013), 2799–2802.
21. Partridge, K., Chatterjee, S., Sazawal, V., Borriello, G., and Want, R. TiltType: Accelerometer-supported text entry for very small devices. In *Proc. UIST '02*, ACM (2002), 201–204.
22. Perlin, K. Quikwriting: Continuous stylus-based text entry. In *Proc. UIST '98*, ACM (1998), 215–216.
23. Roudaut, A., Huot, S., and Lecolinet, E. TapTap and MagStick: Improving one-handed target acquisition on small touch-screens. In *Proc. AVI '08*, ACM (2008), 146–153.
24. Siek, K. A., Rogers, Y., and Connelly, K. H. Fat finger worries: How older and younger users physically interact with PDAs. In *Proc. INTERACT '05*, Springer-Verlag (2005), 267–280.
25. Soukoreff, R. W., and MacKenzie, I. S. Metrics for text entry research: An evaluation of MSD and KSPC, and a new unified error metric. In *Proc. CHI '03*, ACM (2003), 113–120.
26. Suzuki, K., Okabe, K., Sakamoto, R., and Sakamoto, D. Fix and slide: Caret navigation with movable background. In *Proc. UIST '15 Adjunct*, ACM (2015), 79–80.
27. Vertanen, K., Memmi, H., Emge, J., Reyal, S., and Kristensson, P. O. VelociTap: Investigating fast mobile text entry using sentence-based decoding of touchscreen keyboard input. In *Proc. CHI '15*, ACM (2015), 659–668.
28. Vogel, D., and Baudisch, P. Shift: A technique for operating pen-based interfaces using touch. In *Proc. CHI '07*, ACM (2007), 657–666.
29. Ward, D. J., Blackwell, A. F., and MacKay, D. J. C. Dasher - A data entry interface using continuous gestures and language models. In *Proc. UIST '00*, ACM (2000), 129–137.
30. Wobbrock, J. O., Myers, B. A., and Kembel, J. A. EdgeWrite: A stylus-based text entry method designed for high accuracy and stability of motion. In *Proc. UIST '03*, ACM (2003), 61–70.
31. Xia, H., Grossman, T., and Fitzmaurice, G. NanoStylus: Enhancing input on ultra-small displays with a finger-mounted stylus. In *Proc. UIST '15*, ACM (2015), 447–456.
32. Zhai, S., Hunter, M., and Smith, B. A. The Metropolis keyboard - An exploration of quantitative techniques for virtual keyboard design. In *Proc. UIST '00*, ACM (2000), 119–128.
33. Zhai, S., Hunter, M., and Smith, B. A. Performance optimization of virtual keyboards. *Human–Computer Interaction 17*, 2-3 (2002), 229–269.