ForceBoard: Subtle Text Entry Leveraging Pressure

Mingyuan Zhong[1] **Chun Yu**[123†] **Qian Wang**[12] **Xuhai Xu**[1] **Yuanchun Shi**[123]
[1] Department of Computer Science and Technology
[2] Key Laboratory of Pervasive Computing, Ministry of Education
[3] Global Innovation eXchange Institute, Tsinghua University, Beijing, 100084, China
{zhongmy14, q-wang14, xxh14}@mails.tsinghua.edu.cn, {chunyu, shiyc}@tsinghua.edu.cn

ABSTRACT
We present ForceBoard, a pressure-based input technique that enables text entry by subtle finger motion. To enter text, users apply pressure to control a multi-letter-wide sliding cursor on a one-dimensional keyboard with alphabetical ordering, and confirm the selection with a quick release. We examined the error model of pressure control for successive and error-tolerant input, which was incorporated into a Bayesian algorithm to infer user input. A user study showed that, after a 10-minute training, the average text entry rate reached 4.2 WPM (Words Per Minute) for character-level input, and 11.0 WPM for word-level input. Users reported that ForceBoard was easy to learn and interesting to use. These results demonstrated the feasibility of applying pressure as the main channel for text entry. We conclude by discussing the limitation, as well as the potential of ForceBoard to support interaction with constraints from form factor, social concern and physical environments.

Author Keywords
Pressure input; text entry; one-dimensional input.

ACM Classification Keywords
H5.2. Information interfaces and presentation: User interfaces — Input devices and strategies.

INTRODUCTION
Nowadays, touchscreen-based software keyboards are the main facility for text entry on mobile and wearable devices. However, there are circumstances when this method is inadequate due to various restrictions, such as technical restriction (e.g., typing on a wet touchscreen), spatial restriction (e.g., typing on a smartwatch or a smart wristband), social restriction (e.g., typing a short message during a meeting), and physical restriction (e.g., putting hands in a pocket).

In this paper, we present ForceBoard, a pressure-based input technique that enables text entry by subtle motion. As shown in Figure 1, the display interface of ForceBoard contains a one-dimensional keyboard with alphabetical ordering and a multi-letter-wide sliding cursor on it. To select a letter, users apply pressure to move the cursor across the keyboard and perform a quick release (reducing pressure quickly but not lift off the finger) to confirm the selection. We deem ForceBoard as a potential solution for the aforementioned restrictive interaction scenarios, considering its space-saving feature and subtle movement of the finger.

Figure 1. The experiment interface in the pilot study. Only the alphabetical layout and a 5-letter-wide cursor are presented. The orange window is the sliding cursor to select letters, whose position is depended on the pressure level.

To understand users' ability of pressure control in the context of text entry, we carried out an independent experiment to obtain the error model of pressure control for successive and error-tolerant input, which was then incorporated into the statistical decoding algorithm to interpret user input. Meanwhile, tactile feedback was employed to facilitate pressure control and to reduce the requirement for visual attention. We also design ForceBoard to support the input of individual letters, numbers, and punctuations.

To evaluate the performance of ForceBoard, we conducted a lab study with twelve participants, where both character-level and word-level input performance were tested. Results showed that after 10-minutes practicing with ForceBoard, users could type 4.2 WPM (words per minute) for character-level input (error rate: 1.1%), and 11.0 WPM for word-level input (error rate: 0.5%). The obtained input speed (11.0 WPM) was significantly faster than prior one-dimensional text entry techniques [3, 29]. Subjective user feedback revealed that ForceBoard was easy to learn and interesting to use. In conclusion, our research for the first time demonstrates the feasibility of applying pressure as the main channel for text entry, which is useful for subtle interaction when one is in interaction restrictive settings.

† denotes the corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org
CHI 2018, April 21–26, 2018, Montreal, QC, Canada
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5620-6/18/04…$15.00
https://doi.org/10.1145/3173574.3174102

RELATED WORK

We summarize relevant work from three aspects: human ability of pressure control, pressure as an input modality, text entry on constraint interfaces.

Ability of Pressure Control

Previous works have studied human's ability to control pressure. Mizobuchi et al. [11] found that users had the ability to accurately distinguish 6 levels of pressure intensity using a stylus. Wilson et al. [24] reported that, with adequate feedback, users could even distinguish pressure intensity up to 10 levels. Wang et al. [22] investigated the maintenance accuracy when applying pressure on a sensor with a probe ranging from 0.5N to 5N. The results showed that the absolute error linearly increased as the pressure increased, while the relative error was greatest when pressure was 0.5N. Ramos et al. [12] reported that it was difficult to control pressure accurately when pressure intensity was small. Yin et al. [27] quantified the pressure used for tasks interaction. They reported that normal tasks such as drawing and writing had a pressure sized from 0.82N to 3.16N, while the resting pressure was between 0.78 N and 1.58 N. The change of error rate was found from 4.9% for one layer of pressure to 35% for six layers of pressure. These results pointed out the challenge of using pressure to select a letter (1/26) on a keyboard.

Researchers have found that visual feedback was important to ensure the high accuracy and low variance to select a pressure target, that is, to apply pressure of a required amount [12, 20, 22, 24, 27]. Besides, there are two methods to select a pressure widget on a pressure-based interface: Dwell and Quick-Release. Results showed that Quick-Release provided faster response speed while Dwell demonstrated higher response accuracy [1, 12, 24]. Stewart et al. [20] examined different transfer functions mapping pressure values to input values, including linear [12], quadratic [2], and fish-eye [18], and found linear function outperformed the other functions. In our solution, we adopt the Quick-Release and linear transfer function to build the ForceBoard.

Pressure & Text

Various pressure-based interaction techniques have been proposed in literature, such as menu selection [2, 12, 25], navigation [19, 26], zooming in/out [13, 14], scrolling [9, 15], multiple controls [7], and so on. We direct users to [9, 26] for an overview of them.

Pressure has also been utilized in text entry, but merely as an auxiliary channel. Brewster et al. [1] proposed a novel way to allow users to specify the case of letters with pressure on soft keyboards, by applying soft pressure to indicate lower case and a hard pressure to indicate upper case. PressureText [10] allows users to select a letter on ambiguous keyboard with three levels of pressure. On the other hand, ForceType [23] enables users to explicitly control the uncertainty of touch via touch pressure. This enables users to control over the relative influence of the language model in the statistical decoding.

Touch-based Text Entry

Touch-based text entry on smart devices have traditionally adopted a QWERTY or QWERTY-like layout, with a tap or gesture-based interaction scheme on a touch surface. Reyal et al. [16] studied the performance of Smart Touch Keyboard (STK) and Smart Gesture Keyboard (SGK) using Google Keyboard in the wild, where text entry rates reached 31.1 wpm for the STK and 39.1 wpm for the SGK. WatchWriter [6] implements touch and gesture typing on a smart watch, reaching 22 wpm for the STK and 24 wpm for the SGK. 1Line keyboard [3] condenses the QWERTY keyboard into a line with eight ambiguous keys, and reaches 30 wpm on tablet computers. Compared with ForceBoard, touch-based QWERTY keyboards are 2~3 times faster, but also require larger area for input and movement of fingers. Therefore, we recognize ForceBoard as an alternative solution when the input surface or allowed finger movement is limited.

Text Entry on Constraint Interface

Recently, text entry techniques on constraint user interfaces are emerging. Rotext [21] allows users to input on a circular keyboard with a rotation device. Text entry rate of 12.6 WPM was recorded at the beginning usage and could increase to 21 WPM with extensive training. SwipeZone [3] allows users to input characters on the smart glass with gestures, and text entry rate was 8.73 WPM. SWIM [27] employs the tilting of the device to move the swiping cursor on the QWERTY keyboard. It reaches 15 WPM text entry rate. 1D Handwriting [29] allows users to handwrite on the side touchpad of smart glasses, by projecting 2D strokes into 1D space. The input speed was 4.67 WPM for character-level input and 9.72 WPM for word-level input.

ForceBoard shares a similar visual design with a scanning keyboard [16], which is specially designed for augmentative and alternative communication (AAC). A scanning keyboard also has a linear keyboard layout and a moving cursor. But the input method is completely different: A cursor jumps periodically across letters; Users signal the selection when the desired key is selected. In contrast, users of ForceBoard actively apply pressure to control the position of the cursor to select a character.

PILOT STUDY

The pilot study aimed to examine feasibility of pressure-based text entry and help decide a number of design choices, i.e., the keyboard layout, the selection method and the appropriate width of cursor that should be used.

Participants and Apparatus

We recruited eight participants from the local university campus. All participants had at least two years of experience with smart phones. Three of them owned a pressure-sensitive phone (iPhone 6S), but none of them have experience in using the pressure-based interaction.

We implemented the experimental system on an iPhone 6s device, which had a pressure-sensitive touchscreen (size: 4.7”; resolution: 1334×750). The range of pressure sensitive was reported to be 0–0.38 N (0–385 grams of force), with a resolution of 0.01–0.03 N (1–3 grams of force). The system reported pressure data according to the pressure applied on the touchscreen.

Design and Procedure

The experiment was designed with three independent factors:

1. keyboard layout: alphabetical A-Z, QWERTY, and ENBUD [21]
2. Cursor width: 1, 3, 5, 7, and 9 letters
3. Selection methods: Dwell and Quick Release. Users first move the cursor to the desired position. Then, users maintain pressure for 300ms in Dwell mode. In Quick Release mode, users release pressure immediately instead.

Figure 1 shows one example of the keyboard design. We asked participants to move the cursor on each character and then release finger. Each of the conditions of the independent factors were tried and compared with each other. We observed user behaviors throughout the study, and gathered user reports after the study.

Results

All participants reported preference in using the alphabetical layout, and considered it to be the easiest to search for letters. We observed that users were not familiar with QWERTY or ENBUD in one-dimension, and users spent a much longer time in searching letters before input.

We found it was almost impossible for users to input with 1-letter-wide (1/26) cursor. 3-letter-wide (1/8) cursor also imposed a significant load on participants, which was reported to be uncomfortable and difficult to control. These results were consistent with past research [11]. Users were more satisfied with cursor whose width is no less than 5 letters. However, the wider a cursor is, the more ambiguous it becomes. To determine the appropriate width of the cursor, we conducted a simulation using a language model containing the 10,000 most used words (with frequency for each word) from ANC (American National Corpus). Results showed that with the 9-letter-wide cursor (1/3), up to 14.5% of words would not show up in the top 5 candidates. In comparison, this number for the 7-letter-wide cursor (1/4) and 5-letter-wide (1/5) cursor was 3.6% and 0.5% respectively. Therefore, we deemed these two cursors as more promising.

In consistent with previous research [2,12,24], users preferred Quick Release as it was much faster than Dwell, especially when the cursor was wide enough. Although Dwell was more accurate for input, it required more effort to maintain the pressure stability. In particular, we found that for inputting a serial of letters, users subjectively felt it was faster and tended to not lift off their finger between letters. We refer it as *on-contact Quick Release* in this paper, which was employed in ForceBoard.

Based on the findings found in the pilot study, instead of designing a technique based on accurate selection of individual letters that caused significant delay of the input speed, we inclined to adopt the multi-letter-wide cursor, and take advantage of language models to resolve the ambiguity [5]. This decision placed higher priority on word input rather than individual character input. To support this decision, we assumed inputting words from a predefined lexicon represents the majority of daily text entry tasks. Similarly, many smart keyboards also leverage statistical decoding to auto-correct users' input errors [5].

STUDY 1: PRESSURE CONTROL FOR TEXT ENTRY

Before we proceed to the design of ForceBoard, we conduct an independent study to obtain the error model of pressure control, which is an essential component of the statistical decoding algorithm to interpret user input. Although error model of pressure input has been studied by the past research, their objective emphasized only on accurate selection [11,24]. There is no research thus far take consideration of continuous and fast input, with a reasonable tolerance of inaccurate control.

Apparatus and Participants

We used the same apparatus as in the pilot study and recruited another fourteen participants (seven males and seven females, aged between 18 and 33, mean = 23.0). All participants had at least two years of experience with smart phones, and none of them had experience in using pressure-based input technique before.

Figure 2. The Wizard of Oz keyboard with cursor width as 7, used to collect users' typing data

The experimental system was modified from the one used in the pilot study. In this system, we displayed input tasks and input results. Participants held the phone with their dominant hand, and made pressure inputs with the thumb of the same hand. The system recorded all input events, including pressure and the time-stamp. Before having a concrete keyboard technique, we used a Wizard of Oz keyboard, which displayed a sequence of asterisks instead of letters to prevent users adjusting their input behavior according to the result displayed. This allowed us to identify the natural behavior of pressure control in text entry.

Design and Procedure

The within-subject factor was *cursor-width* (5 and 7 letters). The cursor-width is measured in terms of the width of one letter on the keyboard (about 1.8 mm). For each width, 78 arbitrary letter sequences were randomly generated, with each containing 3 letters that was easy to be memorized. In sum, each participant entered 234 letters where each letter in the English alphabet (A-Z) repeated 9 times.

We asked participants to memorize each sequence, and then select each letter in sequence with on-contact Quick release. In other words, users applied pressure on the screen to move the cursor to the desired letters, select the letters by releasing pressure but not letting go of the finger off the screen, and repeated the procedure to select multiple letters. We told the participants to select letters promptly, without the need to fixate on specific letter.

Results

We investigate the pressure distribution of the user input around the target, as well as the input time and learnability of pressure input.

Error Model of Pressure Control

We define *offset* as the distance between the cursor location at Quick Release (the highest position of pressure) and the intended target center. Overshooting the target position results in a positive *offset*. The unit of *offset* is also the width of one letter on the keyboard. For each letter, data fallen outside three times the standard deviation were removed as outliers, which accounted for 1.5% of the data. Error model of pressure control was obtained based on the distribution of *offset*. Figure 3 shows the mean *offset* and the standard deviation of *offset* for each letter.

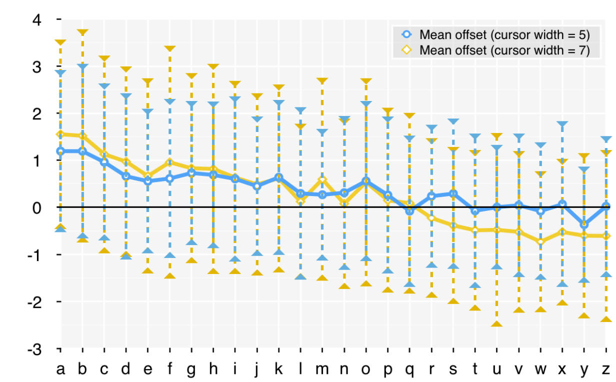

Figure 3. Mean *offset* of each letter, with standard deviations shown as error bars. One unit of deviation corresponds to 1.8 mm on the device.

The mean *offset* decreased linearly as the position on the keyboard increased (*cursor-width* = 5: R^2 = 0.81; *cursor-width* = 7: R^2 = 0.93), showing a decreasing tendency of users to overshoot as the expected pressure increased. RM-ANOVA showed a decreasing trend of the mean *offset* as pressure increased (*cursor-width* = 5: $F_{25,325} = 6.35$, p<.001; *cursor-width* = 7: $F_{25,325} = 11.85$, p < .001). These findings were found consistent with the previous research [12]. However, we did not observe significant variation in the standard deviation of *offset* as force increased (*cursor-width* = 5: $F_{25,325} = 1.35$, p=.123; *cursor-width* = 7: $F_{25,325} = 1.01$, p = .456).

Figure 4 shows the overall distribution of the *offset* by merging all data for the 7-letter-wide cursor together. This was done by offsetting the mean of distribution for each letter to zero. As shown in Figure 4, the overall distribution is slightly right-skewed, with a skewness of 0.67, where its standard deviation is 1.87 letters.

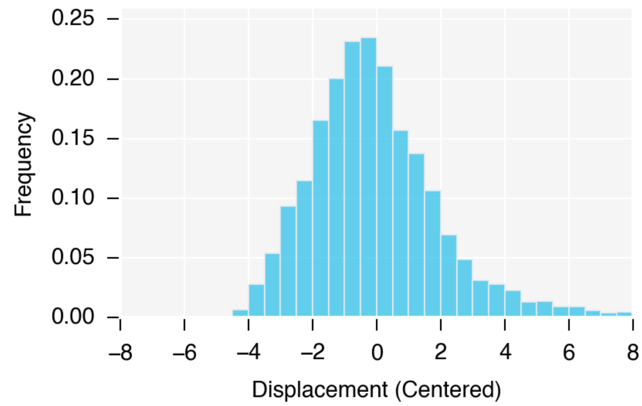

Figure 4. Distribution of centered *offset* of all letters for the 7-letter-wide cursor

Occasionally, users overshot or undershot the target. The target missing rate was 7.7% for the 5-letter-wide cursor, and 5.8% for the 7-letter-wide cursor. It was observed that when users noticed the overshoot, they tended to release pressure to make the cursor return to the intended letter. Therefore, the maximum pressure point was not always the intended position. This observation suggested the need to allow users to re-position the cursor if overshooting occurs.

Input times

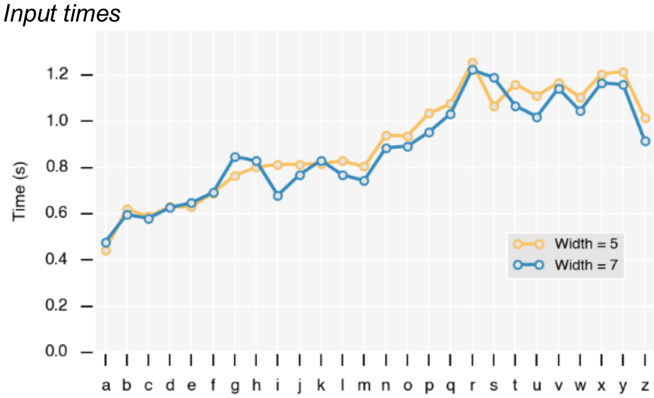

Figure 5. Mean input time for each letter for the 5-letter-wide cursor and the 7-letter-wide cursor

Figure 5 shows the mean input time for each letter A-Z. RM-ANOVA showed significant effect of *letter position* on input time ($F_{25,325} = 7.22$, p < .001), which indicated that input time is increased when the required pressure is increased. The effect of *cursor width* on input time was not significant ($F_{1,13}$ = 1.30, p = .276). The average time for five-letter-wide cursor and seven-letter-wide cursor were 0.91 second (SD = 0.58) and 0.86 second (SD = 0.52), respectively.

DESIGN AND IMPLEMENTATION OF FORCEBOARD

Based on the findings of the pilot study and Study 1, we derived our final design and implementation of ForceBoard.

Interaction Design

The one-dimensional keyboard

We divided the one-dimensional keyboard into three regions, as shown in Figure 6. The left region is for resetting of the cursor, which has a length of 5 letters. The middle region is for characters, including a space bar, letters in alphabetical ordering, and punctuation marks. The right region consisted of a delete button.

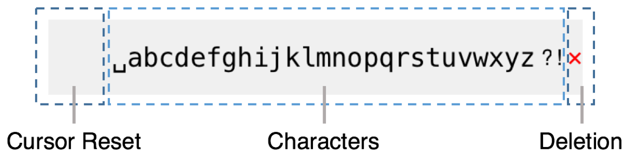

Figure 6. One-dimensional keyboard layout for ForceBoard.

ForceBoard employs two cursors: 1) a sliding cursor that always reflects the pressure level applied; and 2) a selection cursor (the orange underline), which indicates the letters that are recognized by the system.

As the pressure increases, the selection cursor follows closely with the sliding cursor. When the pressure is quickly released to the reset region, the letters indicated by the selection cursor will become the input. However, if the user releases the pressure and dwells at a lower position for over 300 ms, the selection cursor will be re-positioned to align with the sliding cursor. Then a quick release will result in the inputting of letters covered by the current selection cursor.

Figure 7. Behavior of the sliding cursor (the box) and the selection cursor (the underline) during the selection process

Input letters, words, numbers and punctuations

To input a letter, a user first applies pressure to move the cursor over the intended letter, and then performs a quick release. Once the finger is lifted off the screen, the system displays candidate letters in descending order of probability according to the error model derived from Study 1. The first candidate is chosen by default. To select a different candidate, users can tap on the screen to select the next candidate, and long press to select the previous one. We found multiple taps were a faster and more accurate method to select a target than pressure-based selection.

To input a word, user needs to repeatedly select letters of that word without lifting the finger off the screen. A list of words is updated in real-time as the user makes the inputs. Once the finger is lifted off the screen, the system leverages statistical decoding to interpret intended word and list a number of candidate words, from which users can select the intended word by taps. The same method can be used to select candidates from word list. For single-letter words such as “a” and “I”, an additional word “a␣” and “I␣” is displayed alongside their respective letter, indicating a choice between a word and a letter. A space is automatically added when entering the next word. Figure 8 shows an example of the pressure applied in order to input a word.

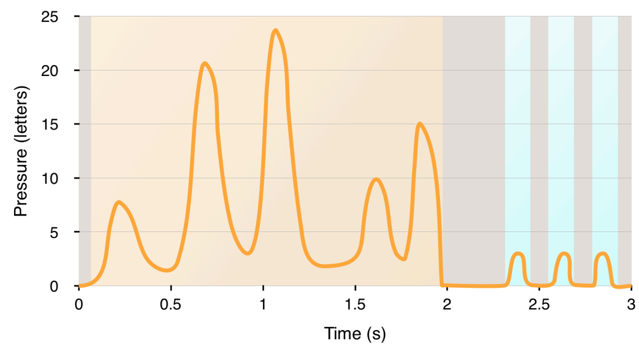

Figure 8. The pressure a user made while inputting the word “force”. The five peaks in the light brown region each represents a pressure input for each letter in the word. Note that the user’s finger was continuously in contact with the surface (pressure greater than zero). The user then released the finger, and tapped three times (blue region) to select the word from candidates.

To make a deletion, user makes a fast movement of the cursor to the right edge of the keyboard. This triggers a deletion with haptic feedback. If user triggers a deletion in the midst of entering a pressure sequence, a pressure inputted letter is deleted. However, if a deletion is made after a sequence is confirmed, the whole word is deleted.

To input a number or a punctuation, user moves the cursor to the right, and dwells on the number and punctuation region. A list of numbers and punctuations will be shown as candidates.

Word Prediction

We leverage statistical decoding to interpret user input. The algorithm has two essential components, which are the error model we derived from Study 1, and a unigram language model. The basic idea is to compute the posterior probability of all words in a predefined language model given the observed user input, and rank them according to their probability. For the language model, we used the top 10,000-word in the ANC, as described in the Pilot Study. The details of the algorithm are as follows.

Suppose user input a sequence of pressure ($I = p_i \dots p_n$). The posterior probability of a candidate word w ($w = l_1 l_2 \dots l_n$) given input I is $P(w|I)$. According to Bayes Rule, we have

$$P(w|I) = \frac{P(w) \cdot P(I|\, w)}{P(I)}$$

where $P(I)$ is constant for all candidates and thus can be ignored. $P(w)$ is the word frequency specified in the language model. Further, we suppose the pressure applied for each letter was independent. Thus, we have

$$P(I|w) = \prod_{i=1}^{n} P(p_i|l_i)$$

where $P(p_i|l_i)$ represents the distribution of pressure for letter l_i, which is derived from the error model we obtained from Study 1.

To enable auto-completion, we modified the equation as

$$P(w, I) = P(w) \cdot \prod_{i=1}^{n} P(p_i|l_i) \cdot \sigma^{m-n}$$

where σ ($0 < \sigma < 1$, effective when $m > n$) is a heuristic penalty term for avoiding long but frequent words dominating the candidate list. The predicted words are then sorted in descending order of their probability, and presented to the user.

In addition, for OOV words, users can input them by selecting individual letters in sequence, in which case the prediction from the language model will not be calculated.

STUDY 2: PERFORMANCE EVALUATION

In Study 2, we aimed to evaluate the performance of ForceBoard. We tested its performance for both character-level input and word-level input.

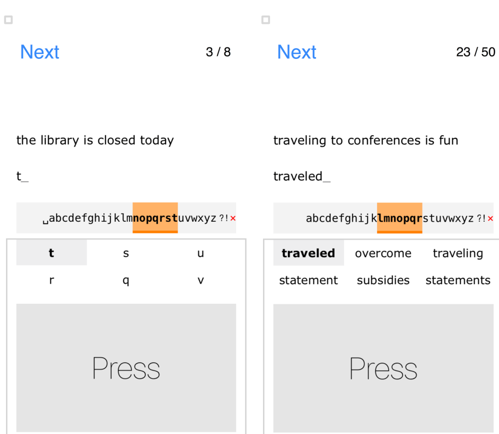

Figure 9. The experiment interface for character-level input (left) and word-level input (right). A 7-letter-wide cursor was used in this experiment.

Participants and Experiment Setup

Twelve participants (six males and six females, ages 18–27, mean = 21.9) were recruited from the local university campus. Four of them were regular users of devices equipped with pressure sensors, but none of them used the technology frequently.

The experiment setting was similar to Study 1. In Study 2, we implemented the interaction design and the decoding algorithm as described before, and rendered a region for displaying the phrase to be entered (Figure 9).

Design

Participants were asked to type several given phrases. The experiment was designed into two sessions: character-level input in one session and word-level input in the other session.

Character-level Input

In the character-level session, participants were asked to finish the input of eight phrases letter by letter. The phrases were divided into four blocks, with two phrases in each block. Participants could have a one-minute break between each block. In this session, word-level text entry was disabled.

Word-level Input

In this session, the word-level text entry was turned on and participants need to type in 40 phrases. These phrases were also divided into four blocks, with 10 in each block.

Note that the phrases were randomly sampled from MacKenzie and Soukoreff's phrase set [6] that were averagely consisted of 25 letters or five words. Hence in both sessions participants had similar times of input letter or word selection.

Procedure

Before the experiment, participants were briefed about ForceBoard and the experiment objectives. Then participants spent two minutes on familiarizing with pressure input and the user interface. Before each session, participants received instructions regarding the relevant session, and input two phrases as a warm-up. During the experiment, participants were required to input "as quickly and accurately as possible". After they finish each session, participants were asked to finish a questionnaire asking about their experience during typing.

Results

Error rates

The average uncorrected error rates were 1.1% (SD = 4.4%) for character-level input and 0.47% (SD = 2.5%) for word-level input. There was no significant effect of *block* on error rate for both conditions (character-level: $F_{3,33} = 1.61$, $p = .21$, word-level: $F_{3,33} = 0.76$, $p = .52$).

The average corrected error rates were found 2.0% (SD = 5.2%) for character-level input, and 1.8% (SD = 4.8%) for word-level input. The effect of block on corrected error rate was insignificant for word-level input ($F_{3,33} = 0.72$, $p = .55$), but significant for character-level input ($F_{3,33} = 3.67$, $p = .023$). For character-level input, corrected error rate was 5.8% in the first block, and declined to 0.20% in the last block. This indicated users could learn to input more accurately with practice.

Text entry speed

Figure 10 shows the text entry rate of ForceBoard for input. The calculation for text entry speeds included the time users spent on correcting errors. The average speed was 4.24 WPM for character-level input, and 11.04 WPM for word-level input. The mean text entry rate speeds achieved 4.42 WPM for the last two phrases for character-level input, and 12.80 WPM in word-level input. RM-ANOVA showed a significant main effect of *block* on the text entry rates for word-level input ($F_{3,33}$ = 9.51, p = .0001), which demonstrated the learning effect. However, no significant effect of *block* was observed for character-level input ($F_{3,33}$ = 1.55, p = .22). The best performer achieved 6.6 WPM for character-level input and 17.5 WPM for word-level input.

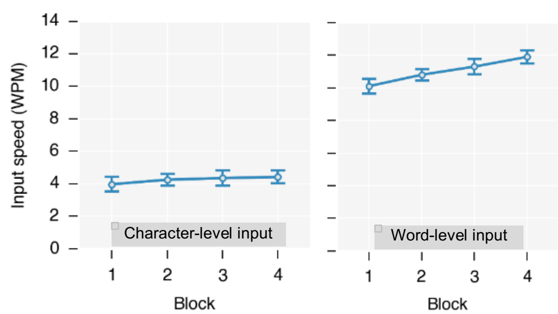

Figure 10. Average input speeds across 4 blocks. Error bars represent 95% confidence intervals.

Time breakdown of word-level input

We analyzed the time composition per key to provide insights into how users learn to interact with ForceBoard. To achieve this, we divided the input time for each character into three components: 1) *pressing time* (the time of pressure being applied from nearly zero to maximum); 2), *release time* (the time of pressure being released from maximum to nearly zero); and 3) *reset time* (the elapsed time between the release of the last letter and the application of the next letter).

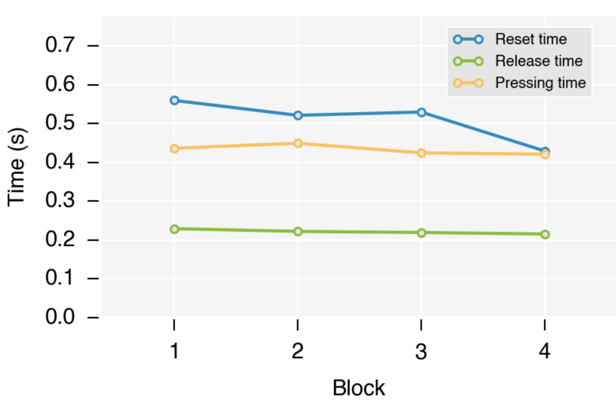

Figure 11. Word-level input time breakdown over blocks

The mean values of *pressing time, release time* and *reset time* were 0.43 second (SD = 0.31), 0.22 second (SD = 0.18) and 0.51 second (SD = 0.86) respectively. There were no significant effects of the block found on *pressing time* and *release time*. However, we observed a significant effect of *block* on *reset time* ($F_{3,33}$ = 5.76, p = .003). The *reset time* generally decreased with *block* (Figure 11), with the last block at 23% faster (mean = 0.43 s, SD = 0.68 s) than the first block (mean = 0.56 s, SD = 0.95 s). This suggested that users were getting better to seek keys across the keyboard after more practices.

User Feedback

Figure 12 shows the results of questionnaire. The questions were all answered on a 1-5 Likert Scale (1 not at all – 5 a lot). The results showed that the mental demand and frustration experienced while using ForceBoard were low (mental demand: M = 2.90, SD = 1.04; frustration: M = 2.54, SD = 1.03). Five out of 12 participants explicitly commented that ForceBoard was easy to learn, and could be mastered in just a few minutes. Users rated the physical demand to be low (M = 2.73, SD = 1.19), suggesting that ForceBoard could be used without causing significant physical stress. Seven out of 12 users also reported that ForceBoard was interesting to use, and that entering text with only subtle movements of the finger was convenient. In overall, participants liked the interaction style of ForceBoard, and rated an average score of 4.0 (SD = 0.45).

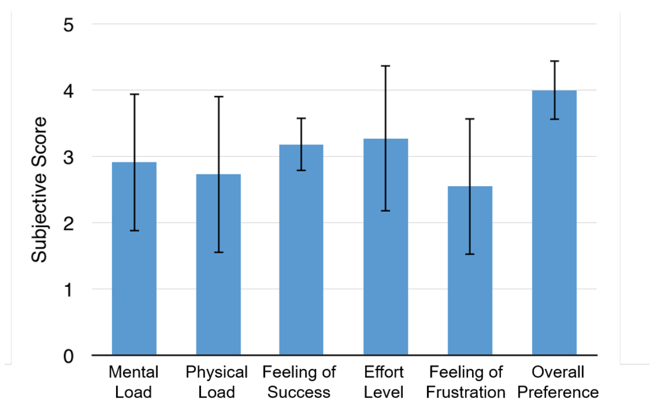

Figure 12. Users' subjective feedback of ForceBoard. Error bars represent standard deviation.

APPLICATIONS

To our knowledge, ForceBoard is the first in literature to leverage pressure control as a main input channel for text entry. ForceBoard presents a unique text entry method that requires only isometric pressure control of a finger. Our exploration successfully demonstrates the feasibility and performance of this novel typing method. Although the present research was conducted on an iPhone device, ForceBoard can also have several other application scenarios. We will discuss them below.

Limited Form Factor of Device

The most noteworthy feature of ForceBoard is that it requires only a single point for input. Therefore, it is especially valuable for smart devices of limited form factors. These include not only popular wearable devices such as smartwatches and smart wristbands, but also emerging and future smart devices such as smart rings and pens. Meanwhile, the software interface of ForceBoard is also very compact. It has only a one-dimensional A-Z keyboard and a multi-letter width cursor. A calculation shows that the 7-letter-wide cursor would be more than 5 mm wide on an Apple watch. This means that ForceBoard significantly saves on screen real estate, and can be deployed on devices with a very small screen.

Subtle Movement of Fingers

Using ForceBoard, finger movement is small due to the isometric pressure control. Thus, ForceBoard can be useful when subtle movement of the finger is preferred, for example, for users who are physically constrained (e.g., keeping hands in a pocket), users who does not want to be noticed (e.g., replying a short message in a meeting), and users who want to keep the input content secret (e.g., entering passcode on an ATM machine or a POS machine).

Inaccessibility with Capacitive Touchscreens

Currently, capacitive touchscreens are common on smart devices, such as mobile phones and smartwatches. However, they might function improperly in certain circumstances. For example, users wearing a glove cannot touch the screen; when the screen is dampened or contaminated (e.g., during cooking, painting, gardening), it will become insensitive or erroneous; capacitive sensing screens cannot work for underwater devices, such as underwater cameras. For all these cases, ForceBoard provides a potential solution, which requires only a pressure sensor.

Compatibility with A Separate Display

Unlike touch keyboard, ForceBoard is instinctively suitable for input-display separate settings (e.g., AR/VR helmets, smart glasses and distant display devices). For instance, during an immersive VR experience, users can easily input text with ForceBoard by applying pressure on the handheld controllers. Although in our experiments, we did not separate the input and display surfaces on iPhone, we argue that since users perform subtle finger movement on ForceBoard, looking down on the finger will provide little information. Therefore, it will not likely suffer from performance degradation with GUI shown on a separate display.

LIMITATIONS AND FUTURE WORK

Although effective, the text entry rate of ForceBoard is much slower than touch-based keyboards [3, 6, 16]. Therefore, we deem the practical value of ForceBoard should be limited to the restrictive interaction scenarios which we discuss above. Moreover, typing on ForceBoard requires continuous visual attention on the cursor in order to form close-loop feedback of pressure control. This is contrast to touch-based keyboard typing where users do not necessarily need to look at each key carefully but leverage muscle memory to input. This means more cognitive load is needed when typing on Forceboard. However, considering in suitable interaction scenarios of ForceBoard, users are more likely to perform short typing, this issue should not be serious.

In addition, there are several limitations of this work, which we also see as opportunities for future work.

First, we observed the learning effect of the ForceBoard through a short-term study with four blocks and 40 phrases in total. A longitudinal study may provide more comprehensive results on learning, physical fatigue, and mental stress, etc., and is worth exploring in the future.

Second, previous work found that isometric rate control has a better performance than isometric position control [30]. In this research, we employed a position control method for the cursor movement in the ForceBoard. It remains as an interesting future question on how users' performance and preference will change with the rate control techniques.

Third, the word prediction algorithm can be upgraded in the future. For example, we can incorporate more sophisticated language models in the algorithm, which might provide more accurate candidates compared to the unigram language model.

CONCLUSION

In this paper, we present ForceBoard, which allows users to type text with pressure with subtle motion movement of the finger. This pushes the limits of motion amplitude required for text entry. We conducted a series of user studies to determine the design strategies, including keyboard layout, width of cursor, feedback design, interaction logic and so on. In particular, we examined users' ability to control pressure in a fast and inaccurate fashion. The empirical results and established error model of pressure control complemented prior research on modeling a person's ability of accurate pressure control. Based on the results, we adapted a widely used statistical decoding algorithm to interpret pressure-based text input. The results showed that after ten minutes of training, users could input 11 words per minute with ForceBoard. Meanwhile, subjective user feedback revealed that ForceBoard was easy to learn and interesting to use. We conclude by discussing potential applications of ForceBoard as well as its limitations.

ACKNOWLEDGEMENTS

This work is supported by the National Key Research and Development Plan under Grant No. 2016YFB1001200, the Natural Science Foundation of China under Grant No. 61672314 and No. 61572276, Tsinghua University Research Funding No. 20151080408, and also by Beijing Key Lab of Networked Multimedia.

REFERENCES

1. Stephen A. Brewster and Michael Hughes. 2009. Pressure-based text entry for mobile devices. In*Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services* (MobileHCI '09). ACM, New York, NY,

USA, Article 9, 4 pages. http://dx.doi.org/10.1145/1613858.1613870

2. Jared Cechanowicz, Pourang Irani, and Sriram Subramanian. 2007. Augmenting the mouse with pressure sensitive input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '07). ACM, New York, NY, USA, 1385-1394. http://dx.doi.org/10.1145/1240624.1240835
3. Frank Chun Yat Li, Richard T. Guy, Koji Yatani, and Khai N. Truong. 2011. The 1line keyboard: a QWERTY layout in a single line. In *Proceedings of the 24th annual ACM symposium on User interface software and technology* (UIST '11). ACM, New York, NY, USA, 461-470. http://dx.doi.org/10.1145/2047196.2047257
4. Tovi Grossman, Xiang Anthony Chen, and George Fitzmaurice. 2015. Typing on Glasses: Adapting Text Entry to Smart Eyewear. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (MobileHCI '15). ACM, New York, NY, USA, 144-152. http://dx.doi.org/10.1145/2785830.2785867
5. Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. 2002. Language modeling for soft keyboards. In *Proceedings of the 7th international conference on Intelligent user interfaces* (IUI '02). ACM, New York, NY, USA, 194-195. http://dx.doi.org/10.1145/502716.502753
6. Mitchell Gordon, Tom Ouyang, and Shumin Zhai. 2016. WatchWriter: Tap and Gesture Typing on a Smartwatch Miniature Keyboard with Statistical Decoding. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (CHI '16). ACM, New York, NY, USA, 3817-3821. http://dx.doi.org/10.1145/2858036.2858242
7. Staas de Jong, Dünya Kirkali, Hanna Schraffenberger, Jeroen Jillissen, Alwin de Rooij, and Arnout Terpstra. 2010. One-press control: a tactile input method for pressure-sensitive computer keyboards. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '10). ACM, New York, NY, USA, 4261-4266. http://dx.doi.org/10.1145/1753846.1754136
8. I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '03). ACM, New York, NY, USA, 754-755. http://dx.doi.org/10.1145/765891.765971
9. Dinesh Mandalapu and Sriram Subramanian. 2011. Exploring pressure as an alternative to multi-touch based interaction. In *Proceedings of the 3rd International Conference on Human Computer Interaction* (IndiaHCI '11). ACM, New York, NY, USA, 88-92. http://dx.doi.org/10.1145/2407796.2407810
10. David C. McCallum, Edward Mak, Pourang Irani, and Sriram Subramanian. 2009. PressureText: pressure input for mobile phone text entry. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '09). ACM, New York, NY, USA, 4519-4524. http://dx.doi.org/10.1145/1520340.1520693
11. Sachi Mizobuchi, Shinya Terasaki, Turo Keski-Jaskari, Jari Nousiainen, Matti Ryynanen, and Miika Silfverberg. 2005. Making an impression: force-controlled pen input for handheld devices. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '05). ACM, New York, NY, USA, 1661-1664. http://dx.doi.org/10.1145/1056808.1056991
12. Gonzalo Ramos, Matthew Boulos, and Ravin Balakrishnan. 2004. Pressure widgets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '04). ACM, New York, NY, USA, 487-494. http://dx.doi.org/10.1145/985692.985754
13. Gonzalo Ramos and Ravin Balakrishnan. 2005. Zliding: fluid zooming and sliding for high precision parameter manipulation. In *Proceedings of the 18th annual ACM symposium on User interface software and technology* (UIST '05). ACM, New York, NY, USA, 143-152. http://dx.doi.org/10.1145/1095034.1095059
14. Jun Rekimoto and Carsten Schwesig. 2006. PreSenseII: bi-directional touch and pressure sensing interactions with tactile feedback. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '06). ACM, New York, NY, USA, 1253-1258. http://dx.doi.org/10.1145/1125451.1125685
15. Christian Rendl, Patrick Greindl, Kathrin Probst, Martin Behrens, and Michael Haller. 2014. Presstures: exploring pressure-sensitive multi-touch gestures on trackpads. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '14). ACM, New York, NY, USA, 431-434. http://dx.doi.org/10.1145/2556288.2557146
16. Shyam Reyal, Shumin Zhai, and Per Ola Kristensson. 2015. Performance and User Experience of Touchscreen and Gesture Keyboards in a Lab Setting and in the Wild. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15). ACM, New York, NY, USA, 679-688. http://dx.doi.org/10.1145/2702123.2702597
17. Roark, Brian, Melanie Fried-Oken, and Chris Gibbons. "Huffman and linear scanning methods with statistical

language models." Augmentative and Alternative Communication 31.1 (2015): 37-50.

18. Kang Shi, Pourang Irani, Sean Gustafson, and Sriram Subramanian. 2008. PressureFish: a method to improve control of discrete pressure-based input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '08). ACM, New York, NY, USA, 1295-1298. http://dx.doi.org/10.1145/1357054.1357256
19. Daniel Spelmezan, Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2013. Side pressure for bidirectional navigation on small devices. In *Proceedings of the 15th international conference on Human-computer interaction with mobile devices and services* (MobileHCI '13). ACM, New York, NY, USA, 11-20. http://dx.doi.org/10.1145/2493190.2493199
20. Craig Stewart, Michael Rohs, Sven Kratz, and Georg Essl. 2010. Characteristics of pressure-based input for mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '10). ACM, New York, NY, USA, 801-810. http://dx.doi.org/10.1145/1753326.1753444
21. William S. Walmsley, W. Xavier Snelgrove, and Khai N. Truong. 2014. Disambiguation of imprecise input with one-dimensional rotational text entry. *ACM Trans. Comput.-Hum. Interact.* 21, 1, Article 4 (February 2014), 40 pages. http://dx.doi.org/10.1145/2542544
22. Dangxiao Wang, Jian Jiao, Gaofeng Yang, and Yuru Zhang. 2016. Force Maintenance Accuracy Using a Tool: Effects of Magnitude and Feedback." *IEEE Transactions on Haptics*:1-1.
23. Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. 2014. Uncertain text entry on mobile devices. In P*roceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '14). ACM, New York, NY, USA, 2307-2316. http://dx.doi.org/10.1145/2556288.2557412
24. Graham Wilson, Craig Stewart, and Stephen A. Brewster. 2010. Pressure-based menu selection for mobile devices. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services* (MobileHCI '10). ACM, New York, NY, USA, 181-190. http://dx.doi.org/10.1145/1851600.1851631
25. Graham Wilson, David Hannah, Stephen Brewster, and Martin Halvey. 2012. Investigating one-handed multi-digit pressure input for mobile devices. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '12). ACM, New York, NY, USA, 1727-1732. http://dx.doi.org/10.1145/2212776.2223700
26. Graham Wilson, Stephen Brewster, and Martin Halvey. 2013. Towards utilising one-handed multi-digit pressure input. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '13). ACM, New York, NY, USA, 1317-1322. http://dx.doi.org/10.1145/2468356.2468591
27. Hui-Shyong Yeo, Xiao-Shen Phang, Steven J. Castellucci, Per Ola Kristensson, and Aaron Quigley. 2017. Investigating Tilt-based Gesture Keyboard Entry for Single-Handed Text Entry on Large Devices. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (CHI '17). ACM, New York, NY, USA, 4194-4202. http://dx.doi.org/10.1145/3025453.3025520
28. Jibin Yin, Xiangshi Ren, and Shumin Zhai. 2010. Pen pressure control in trajectory-based interaction. *Behav. Inf. Technol.* 29, 2 (March 2010), 137-148. http://dx.doi.org/10.1080/01449290902904733
29. Chun Yu, Ke Sun, Mingyuan Zhong, Xincheng Li, Peijun Zhao, and Yuanchun Shi. 2016. One-Dimensional Handwriting: Inputting Letters and Words on Smart Glasses. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (CHI '16). ACM, New York, NY, USA, 71-82. http://dx.doi.org/10.1145/2858036.2858542
30. Zhai, S. Investigation of feel for 6DOF inputs: isometric and elastic rate control for manipulation in 3D environments. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* Sage CA: Los Angeles, CA: SAGE Publications. Vol. 37, No. 4. 323-327