PalmBoard: Leveraging Implicit Touch Pressure in Statistical Decoding for Indirect Text Entry

Xin Yi[12], **Chen Wang**[12], **Xiaojun Bi**[3], **Yuanchun Shi**[12]
[1]Department of Computer Science and Technology, Tsinghua University, Beijing, 100084, China
[2]Key Laboratory of Pervasive Computing, Ministry of Education
[3]Department of Computer Science, Stony Brook University, Stony Brook, NY, United States
{yixin, shiyc}@mail.tsinghua.edu.cn c-w16@mails.tsinghua.edu.cn xiaojun@cs.stonybrook.edu

ABSTRACT
We investigated how to incorporate implicit touch pressure, finger pressure applied to a touch surface during typing, to improve text entry performance via statistical decoding. We focused on one-handed touch-typing on indirect interface as an example scenario. We first collected typing data on a pressure-sensitive touchpad, and analyzed users' typing behavior such as touch point distribution, key-to-finger mappings, and pressure images. Our investigation revealed distinct pressure patterns for different keys. Based on the findings, we performed a series of simulations to iteratively optimize the statistical decoding algorithm. Our investigation led to a Markov-Bayesian decoder incorporating pressure image data into decoding. It improved the top-1 accuracy from 53% to 74% over a naïve Bayesian decoder. We then implemented PalmBoard, a text entry method that implemented the Markov-Bayesian decoder and effectively supported one-handed touch-typing on indirect interfaces. A user study showed participants achieved an average speed of 32.8 WPM with 0.6% error rate. Expert typists could achieve 40.2 WPM with 30 minutes of practice. Overall, our investigation showed that incorporating implicit touch pressure is effective in improving text entry decoding.

Author Keywords
Touch-typing; text entry; input prediction; touch pressure

CCS Concepts
•**Human-centered computing** → **Text input;**

INTRODUCTION
Despite the pervasiveness of touch input devices (e.g., smartphones and tablets), text entry on touch surfaces has remained challenging, due to the imprecision of finger touch [6] and the lack of tactile feedback [10]. This problem is exacerbated for indirect text entry [23, 44], in which the input and output areas are decoupled and users often type in an eyes-free manner (i.e., focusing the visual attention on the output rather than the input area) [10].

To address the challenges in touch-based text entry, modern text entry methods (e.g., Google Gboard, Microsoft SwiftKey, iOS keyboard) have adopted the statistical decoding algorithm [14]: it infers the probability of a word from the input signals (e.g., a series of touch points) and combines it with a language model for prediction. Although using such an algorithm has substantially improved text entry performance, it is far from satisfactory. One shortcoming is that it exclusively relies on the touch point coordinates for decoding [14], ignoring other touch information. Many existing touch sensors can provide extra touch information such as finger pressure during touch interaction. Would using such information improve the text entry decoding performance? If so, how can a statistical decoder leverage it?

In this paper, we have investigated how to incorporate the implicit touch pressure, the finger pressure applied by users in typing, into statistical decoding. We focused on one-handed touch-typing on indirect interface as an example scenario. It features both eyes-free and multi-finger typing, and is highly desirable when only one hand is available. Our investigation showed that incorporating implicit touch pressure significantly improved the text entry performance.

We first carried out a user study to examine users' one-handed touch-typing behavior on a pressure-sensitive touchpad. By analyzing the pressure image data, we discovered distinct patterns in contact areas and levels of pressure when hitting different keys. Based on the collected data, we then optimized the classical statistical decoding algorithm through iterative simulations. Our investigation led to a Markov-Bayesian decoder which incorporated pressure image data into statistical decoding. Compared with a naïve Bayesian decoder, it improved the top-1 accuracy from 53% to 74%. Inspired by the simulation results, we implemented the Markov-Bayesian decoder in PalmBoard, a one-handed, indirect text entry method on a pressure-sensitive touchpad. Our second user study showed that PalmBoard achieved significantly higher input speed and lower error rate over two baselines: naïve Bayesian decoders with rectangle and general keyboard models. PalmBoard had an average input speed of 32.8 WPM with 0.6% error rate. Expert typists could achieve 40.2 WPM with 30 minutes of practice.

CHI '20, April 25–30, 2020, Honolulu, HI, USA.

ACM ISBN 978-1-4503-6708-0/20/04 ...$15.00.
http://dx.doi.org/10.1145/3313831.3376442

Our contributions are three-fold: 1) We examined the typing behavior of one-handed, indirect touch-typing and discovered patterns in implicit touch pressure; 2) We proposed a Markov-Bayesian decoder which incorporated touch pressure data into statistical decoding; 3) We have implemented PalmBoard, a text entry method that implemented the Markov-Bayesian decoder and effectively supported one-handed touch-typing on indirect typing interfaces.

RELATED WORK

Statistical Decoding in Text Entry

Most of today's smart keyboards are based on the statistical decoding algorithm first proposed by Goodman et al. [14], which takes the 2D coordinates of the touches as input. Assuming $I = I_i I_2 ... I_n$ to be the sequence of n input touch points, the algorithm looks for the word W in a predefined dictionary with the highest $P(W|I)$ value, calculated as:

$$P(W|I) = \frac{P(I|W) \times P(W)}{P(I)} \propto P(I|W) \times P(W) \quad (1)$$

where *touch model* $P(I|W)$ models the spatial features of touch points, *language model* $P(W)$ can be estimated based on n-gram word frequencies or machine learning models (e.g. LSTM). In practice, researchers usually assume that there are no insertion or deletion errors, and consecutive touch points are independent, thus:

$$P(I|W) = P(I_1 I_2 ... I_n | W_1 W_2 ... W_n) = \prod_{i=1}^{n} P(I_i|W_i) \quad (2)$$

$P(I_i|W_i)$ is usually calculated using bivariate Gaussian distributions (e.g. [4]). Although simple, this algorithm has achieved impressive performance in a number of text entry scenarios (e.g., [15, 22, 39, 41, 43]).

Researchers have tried to improve the classical Bayesian algorithm by using sentence-level decoding [30, 31, 32] or incorporating other input signals (e.g., accelerometer [11], hand posture [12] and fingering [40]). Researchers have also proposed a *relative* keyboard model rather than an *absolute* one to account for the dependency between consecutive touch points [23, 28]. In this way, the calculation could be modified to

$$P(I_1 I_2 ... I_n | W_1 W_2 ... W_n) = P(I_1|W_1) \prod_{i=2}^{n} P(I_i | I_{i-1}, W_i, W_{i-1}) \quad (3)$$

To our knowledge, none of these works have investigated the effectiveness of these models in one-handed touch-tying, nor have explored incorporating the model with pressure image data. As we will show, combining touch image data with Markov-Bayesian model could effectively improve the input prediction performance. Therefore, our simulation and evaluation results could serve as a complement to existing works.

Text Entry Leveraging Muscle Memory

Touch-typing with two hands on a physical keyboard is the most common and efficient way for text entry, average skilled typists could type 50–80 WPM on physical keyboards [3]. In recent years, many researchers have tried to transfer the muscle memory of touch-tying to different interfaces. Findlater et al. [10] examined users' two-handed typing behavior on a touch-sensitive tabletop, and found that users could yield spatially consistent key press distribution. TOAST [28] and BlindType [23] leveraged a Markov-Bayesian decoder to support two-handed touch-typing on touchable tabletops and one-thumb typing on smartphones in indirect typing scenarios respectively. Users were able to achieve an input speed of 41.4 WPM and 17–23 WPM respectively. TiTAN [37] supported ten-finger typing in mid-air. However, it relied on multiple cursors to select keys and did not provide input prediction, participants only reached 13.6 WPM. ATK [40] proposed an augmented Bayesian model to enable mid-air touch-typing without aiming. Users achieved 29.2 WPM with practice. GlanceType [22] enabled two-thumbs typing on split keyboards without visually aiming at the target keys. Users achieved 27 WPM in evaluation.

The existing works have investigated text entry leveraging muscle memory in various scenarios. In comparison, we are the first to investigate one-handed touch-typing, which only leverage the muscle memory of a single hand, but with all five fingers. As a result, our technique could fill the need in specific scenarios (e.g., when the other hand is occupied), and our findings on the typing behavior serve as a complement to researches of touch-typing.

Pressure-Based Text Entry Techniques

Pressure-sensitive interfaces has showed great potential in facilitating various touchscreen interactions [25, 35]. However, incorporating pressure data into text entry has less been explored. To our knowledge, existing works all focused on using pressure as an explicit channel for interaction (e.g., switching case of the input character [7], select letters from ambiguous keyboard [19, 26, 42], control the uncertainly of each tap [34]). In comparison, we are the first to model touch pressure as an important component of the text entry behavior model, and to explore incorporating implicit touch pressure to improve text entry statistical decoding performance.

STUDY 1: EXAMINING ONE-HANDED TOUCH-TYPING BEHAVIOR

Although researches have showed that users could transfer the muscle memory of two-handed touch-typing to different interfaces, whether they could transfer it to one-handed touch-typing is not obvious. Therefore, we carried out a user study to examine users' one-handed touch-typing behavior. This input scenario is valuable when 1) physical keyboards are not available but touch input is supported (e.g., tabletop); or 2) one-handed text input is preferred or required because the other hand is occupied by other tasks or disabled (situationally or permanently). According to our pilot study, even if we told the participants that they were allowed to rest their fingers and palms onto the pad, all of them preferred not to rest the fingers. Therefore, we focused on examining whether resting their palms would affect the typing performance and preferences. To minimize the potential effect of different input prediction algorithms, we only provided asterisk feedback during typing, which was similar to existing works [4, 10, 28]. The results are thus indicative of users' typing performance under ideal condition (i.e., with a "perfect" algorithm).

Participants

We recruited 12 right-handed participants from the campus (9 male, 3 female) with a mean age of 22 (SD = 2.08). All participants used touchscreen devices at a daily basis, but none of them tried one-hand touch-typing before. Each participant was compensated $10.

Apparatus and Platform

We used a Sensel Morph touchpad to sense the users' input (see Figure 1a). The touch area is consisted of an array of pressure sensors, each could sense 0–50 N of pressure with 32,000 levels. The resolution of the pressure sensors is 185×105, resulting in a PPI of 19.5. The device API reports the pressure image at about 130Hz. Meanwhile, we used the $13''$ screen of a MacBook Pro laptop to display the experiment platform.

To ensure that our results can be generalized to larger touch surface with different aspect ratio, we verified that the size of the touchpad (24.0×13.8 cm) was sufficiently big for all subjects to comfortably rest palms and type: in a pilot study, subjects performed eyes-free one-handed touch-typing on an iPad (25.1×17.4 cm) with a painted boundary of the touchpad (24.0 x 13.8 cm). No touch point fell outside the boundaries.

(a) (b)

Figure 1: (a) The experiment setup and (b) platform.

Figure 1b showed the experiment platform, which was developed using C++. It showed the progress of the experiment, the target phrase, and a keyboard layout for the user to refer to. During text entry, the platform showed asterisk feedback according to the user's input. We implemented a simple touch detection algorithm based on the pressure image data, such that the unintentional touches from the palm could be rejected, and the number of asterisks matched with the touch points.

Experiment Design and Procedure

We used a single-factor, within-subjects design. The only factor was *Posture* with two levels: *Hover* (not allowed to rest the palm, as with using most touchscreen devices) and *Placed* (allowed to rest the palm). Participants completed two sessions of text entry tasks, each corresponding to one posture. Each session was divided into 4 blocks. In each block, participants were asked to type 13 phrases randomly selected from the Mackenzie and Soukoreff phrase set [24] plus 2 pangrams ("the quick brown fox jumps over the lazy dog" and "the five boxing wizards jump quickly"). The order of different sessions were counterbalanced. During the experiment, participants were seated in front of a desk, and typed on the touchpad while looking at the experiment platform on the laptop. They were instructed to type "as naturally as possible" without correcting the input, and were allowed to use all the five fingers of the dominant hand to perform the input. A one-minute break was enforced between different blocks. After the experiment, questionnaires and interviews were carried out to gather subjective feedback. The experiment was videotaped for labelling data.

RESULTS

We collected 39,928 touch points across all participants. For each key, we discarded touch points that lied more than three times of standard deviations away from the collected centroid in either x or y dimension (327 taps, 0.8% of the data). We used RM-ANOVA for statistical analysis, and reported significant results when $p < .05$. For subjective ratings, we used non-parametric tests such as Friedman test.

Text Entry Speed

Text entry speed was measured using WPM [1], which was calculated using:

$$WPM = \frac{|S|-1}{T} \times 60 \times \frac{1}{5} \quad (4)$$

where $|S|$ is the number of characters in the final transcribed string, and T is the elapsed time in seconds from first to last tap in a phrase. Figure 2 showed the text entry speed for each posture. The average speed in *Hover* and *Placed* was 34.5 WPM (SD = 11.3) and 34.1 WPM (SD = 10.3) respectively, which was not significantly different ($F_{1,11} = 0.06, p = .81$). *Block* yielded a significant effect on input speed in *Placed* ($F_{3,33} = 24.2, p < .0001$), but not in *Hover* ($F_{3,33} = 0.12, p = .94$). In *Placed* condition, input speed increased monotonically with block, suggesting that participants could get used to resting the palms with practice, which was consistent with existing work [20].

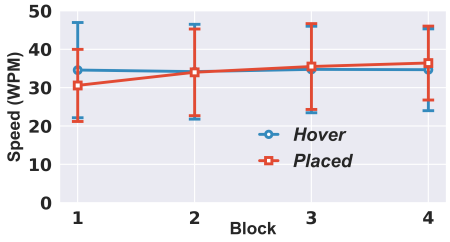


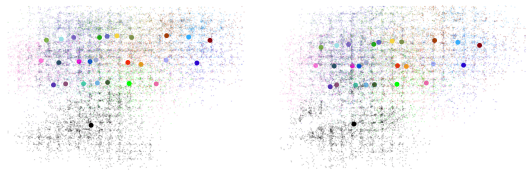
Figure 2: Text entry speed in different conditions, error bar indicated one standard deviation.

Touch Point Distribution

Spread of Size

As users performed typing at different positions on the touchpad, we calculated the center of the keyboard for each user as the middle point between the observed centroid of 'F' and 'J', and aligned touch points from different users according to the center point [10, 28]. Figure 3 showed the collected touch points from all users after alignment. A remarkably overlap among different keys indicated that one-handed touch-typing yield very noisy input signal no matter the palm was lifted or not.

We analyzed the spread of size for touch points of each key, which reflects the touch precision of users. Table 1 showed the



(a) (b)

Figure 3: Collective touch points from all participants in (a) *Hover* and (b) *Placed* conditions, centroid are highlighted.

standard deviations of the collected touch points in both x and y dimensions. Consistent with existing work [28], SD_x was always greater than SD_y, implying that users where more able to control the vertical location of a touch point than horizontal location. Besides, individual spreads were always smaller than collective spreads, suggesting different system offsets among participants. As expected, both SD_x and SD_y was lower in *Placed* compared to in *Hover*, presumably due to a more stable typing posture.

Posture	Hover		Placed	
Axis	X	Y	X	Y
Alphabetical Keys (collective)	10.9 (1.1)	6.8 (0.5)	10.8 (1.7)	6.6 (0.6)
Space Key (collective)	12.1	10.4	14.2	8.9
Alphabetical Keys (individual)	9.1 (2.2)	6.0 (1.4)	8.8 (2.1)	5.7 (2.0)
Space Key (individual)	7.5 (1.3)	7.3 (1.8)	6.9 (1.9)	5.9 (1.9)

Table 1: Spread of size in both dimensions (mm), standard deviations showed in parenthesis, note that SD is not calculable for the second row.

Emergent Keyboard Layout

We analyzed the emergent keyboard layout based on the distribution of touch points, which revealed the perceived keyboard layout of the participants. Figure 3 showed the centroid calculated on collective touch points. In both conditions, the emergent keyboard layout followed well with QWERTY layout. Comparing with the results on size of spread, this implied that users could perform one-handed touch-typing on touchpads with the help of muscle memory, but the input was very imprecise due to lack of tactile and visual feedback.

A major challenge of one-handed touch-typing was hitting keys on the left half of the keyboard. To explore this, we calculated $SR = |QT|/|YP|$ where $|QT|$ denoted the horizontal distance between the centroid of 'Q' and 'T', and $|YP|$ likewise. A smaller SR indicated that the user tended to condense the left half of the keyboard. $SR = 1$ indicated a "perfect" division between the two halves of the keyboard that exactly matched the QWERTY layout.

The mean SR in *Hover* and *Placed* was 0.66 (SD =0.14) and 0.65 (SD = 0.15), with the range being $[0.42, 0.97]$ and $[0.38, 0.88]$ respectively. Linear regression between hand size (measured by the length of middle finger [40]) and SR did not yield observable correlation ($R^2 < 0.25$). Although *Posture* did not exhibited a main effect on SR ($F_{1,11} = 0.02, p = .88$), *Placed* seemed to yielded a slightly lower SR than *Hover*. Surprisingly, for both conditions, SR was lower than 1 for all the participants, suggesting that users always perceived the right half of the keyboard to be wider than the left half (see Figure 4 for illustration). Note that in both conditions, the range of SR was very large, indicating different perceptions of keyboard layout among participants.

(a) (b)

Figure 4: Centroid of touch points from two participants for illustration. (a) $SR = 0.38$, keys on the left half were observably condensed; (b) $SR = 0.97$, fitted well with QWERTY layout.

Key-to-Finger Mapping

Fiet et al. [8] found that in two-handed touch-typing, there existed 6 kinds of key-to-finger mapping strategies regarding the right hand, suggesting significant individual difference. In this study, we manually labelled and analyzed the key-to-finger mapping according to the experiment video. Table 2 showed the number of participants that used different numbers of fingers. 4/12 participants used four fingers both in *Hover* and *Placed*, and 5 participants changed the number of used fingers in different conditions. Interestingly, for individual users, the number of used fingers in *Placed* was always greater than that in *Hover*. This suggested resting the palm onto the touchpad could encourage the users to type with more fingers as in touch-typing. In *Placed*, 4/12 participants used all five fingers to perform the typing.

Participant	Hover	Placed	Participant	Hover	Placed
P1	1	4	*P2*	4	4
P3	4	4	*P4*	2	2
P5	5	5	*P6*	1	2
P7	4	4	*P8*	4	4
P9	4	5	*P10*	5	5
P11	4	5	*P12*	3	4

Table 2: Number of participants that used different number of fingers.

Table 3 showed the average number of keys accounted by each finger. As expected, index finger was used most frequently, followed by middle finger. Ring and little finger was least used. According to our observation, nearly all participants used the index finger to hit keys on the left half of the keyboard. However, for keys on the right half of the keyboard, participants yielded distinct strategies. Although 11/12 participants only used thumb for space key, one participant unintentionally used the thumb to hit alphabetical keys 'z', 'c' and 'v' (17/133 times for the three keys), causing the mean value to be 1.3.

	Thumb	Index	Middle	Ring	Little
Hover	1.3 (1.2)	24.7 (1.9)	11.4 (8.4)	5.8 (5.3)	0.3 (0.9)
Placed	1.3 (0.9)	23.4 (1.8)	12.6 (7.9)	7 (5.0)	0.6 (1.0)

Table 3: Average number of keys accounted by each finger, standard deviations showed in parenthesis.

Features from Pressure Data

We also analyzed the data from the pressure image, which gave us an opportunity to look more into the typing behavior. In this part, we merged data from *Hover* and *Placed* in order to develop a model that could work even when the users switch between the conditions during typing.

Tapping Procedure

We defined the start and end point of a touch to be the instant that the level of pressure reached zero, and calculated the duration of a tap (T) as the elapse between the start and end point. For each data frame at time t, we defined contact size (S_t) as the area with pressure greater than zero, and $Pre_{max,t}$ as the max level of pressure in that frame.

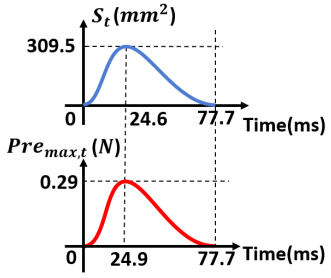


Figure 5: Illustration of S_t and $Pre_{max,t}$ during the procedure of a tap.

Figure 5 illustrated the change of S_t and $Pre_{max,t}$ during a tap. The average duration of a tap was 77.7 ms (SD = 23.1). During a tap, both S_t and $Pre_{max,t}$ first increased with time, then decreased to zero, and the two metrics yield significant correlation with each other. On average, the max S_t and $Pre_{max,t}$ (denoted as $\hat{S}_t$ and $\hat{Pre}_{max,t}$) was reached at 24.6 ms (SD = 16.8) and 24.9 ms (SD = 16.4) respectively, indicating that the press phase was about half shorter than the release phrase, which was similar to [40]. The max S_t and $Pre_{max,t}$ was 309.5 mm^2 (SD = 57.8) and 0.29 N (SD = 0.19) respectively.

Difference Between Keys

We further analyzed whether the above metrics varied among different fingers or keys, Table 4 showed the results. *Finger* yielded a significant effect only on $\hat{S}_t$ (T: $F_{4,12} = 1.74, p = .20$, $\hat{S}_t$: $F_{4,12} = 5.11, p < .05$, $\hat{Pre}_{max,t}$: $F_{4,12} = 1.65, p = .23$). In comparison, *Row* yielded a significant effect on all the three metrics (T: $F_{3,33} = 10.0, p < .0001$, $\hat{S}_t$: $F_{3,33} = 6.57, p < .01$, $\hat{Pre}_{max,t}$: $F_{3,33} = 24.9, p < .0001$). Generally, when hitting keys on lower rows, the taps would yield longer duration, smaller contact size and lighter pressure level. This suggested the possibility of distinguishing taps for different rows based on the pressure features.

	Thumb	Index	Middle	Ring	Little
T	88.8 (25.3)	74.1 (21.2)	78.8 (22.7)	82.7 (23.4)	88.6 (24.7)
$\hat{S}_t$	276.5 (64.8)	318.1 (54.7)	311.9 (53.8)	305.5 (50.0)	232.2 (55.8)
$\hat{Pre}_{max,t}$	0.17 (0.09)	0.32 (0.20)	0.30 (0.17)	0.29 (0.17)	0.14 (0.09)

(a)

	Top	Middle	Bottom	Space
T	74.2 (21.6)	75.0 (21.4)	83.2 (24.5)	87.1 (25.1)
$\hat{S}_t$	321.9 (52.8)	312.8 (51.8)	294.5 (53.6)	285.0 (71.0)
$\hat{Pre}_{max,t}$	0.31 (0.19)	0.32 (0.20)	0.27 (0.18)	0.20 (0.17)

(b)

Table 4: Touch procedure features for (a) different fingers and (b) keys on different rows, standard deviations showed in parenthesis.

Subjective Ratings

We collected users' subjective ratings using a 5-point Likert-scale questionnaire. Dimensions included perceived speed, perceived accuracy, fatigue and overall preference. All responses were assigned a score between 1 and 5, with 5 being the most positive and 1 being the most negative response. Figure 6 showed the ratings from the participants. As expected, *Posture* yielded a significant effect on fatigue ($Z = -2.92, p < .01$). However, no significant difference was found on perceived speed ($Z = -1.15, p = .25$), perceived accuracy ($Z = -0.25, p = .80$) or overall preference ($Z = -0.98, p = .33$). This confirmed that resting the palm could reduce the fatigue, but not affecting other preferences.

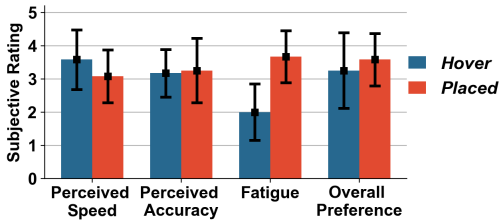


Figure 6: Subjective ratings from the participants. 1 means the most negative and 5 means the most positive. Error bar shows one standard deviation.

IMPROVING THE STATISTICAL DECODING ALGORITHM

In this section, we iteratively carried out a series of simulations to improve the classical statistical decoding algorithm. Our aim was to achieve the optimal input prediction accuracy while supporting palm rejection.

Touch Detection and Palm Rejection

As Sensel Morph only reported pressure image data, the very first step of text input was to detect touch based on the data. In Study 1, we have implemented a naïve touch detection algorithm. Here, we revisited and improved this algorithm according to the collected data. The pipeline of touch detection was illustrated in Figure 7.

The touch detection algorithm performed real time detection on each data frame. Upon the arrival of the pressure image

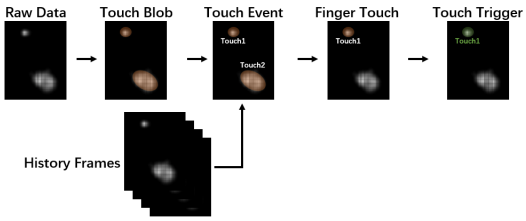


Figure 7: Pipeline of the touch detection and unintentional touch rejection process, a finger is touching with palm resting.

data, the algorithm first detected the touch blob based on the contact size and level of pressure. To minimize the possibility of false negative, the thresholds for contact size and pressure were empirically set to 5.13 mm^2 and 0.01 N respectively.

For each touch blob, the algorithm then constructed the touch events according to previous frames. It first calculated the centroid of each touch as the weighted center of the blob. If the center lied very close to an existing touch event, it would be connected to it, and labelled as "touch move". Otherwise a new touch event would be constructed, and it would be labelled as "touch down". Meanwhile, the algorithm would label existing touch events that had no blob to connect to as "touch up".

The third step was to distinguish intentional touches. As users rarely rested their fingers onto the pad, we only needed to distinguish finger touches from palm touches. As with in Study 1, we analyzed the pressure features from all detected touches of the palm. The average T, $\hat{S}_t$ and $\hat{Pre}_{max,t}$ of palm touches was 1129.0 ms (SD = 594.5), 1271.5 mm^2 (SD = 948.8) and 0.15 N (SD = 0.19) respectively, which was very distinguishable from finger touches (see Table 4). We built a decision tree with x, y and the three metrics of a touch as features, the algorithm reach a F-1 score of 1.0 on the data from Study 1. After palm rejection, the algorithm would trigger intentional touch events at "touch up". This design is consistent with most smartphones. According to Figure 5, this would yield a delay less than 77 ms.

Input Decoder Simulation Design

We optimize the input decoder based on the Bayesian method [14]. Specifically, we optimized three factors one-by-one through a series of simulations to find the optimized algorithm with highest input prediction accuracy: layout (BIKEY vs. QWERTY), pressure (with vs. without) and keyboard model (absolute vs. relative). For each combination, we tested both the general and the personalized model. In Table 5, each algorithm was named as <layout>-<pressure>-<model>, where <layout> takes the value of 'BIKEY' and 'QWERTY', <pressure> takes the value of 'W' (with) and 'O' (without), <model> takes the value of 'ABS' and 'REL'.

We used all the touch data in Study 1 for testing. As the data in *Hover* and *Placed* conditions did not yield significant differences in terms of touch point distribution, we merged all the data to increase the external validity of the simulation results. For each candidate algorithm, we took the intentional touch points for each word as input, and checked the output word. In total, we tested 7,995 words. We used the top 10,000 words in American National Corpus [2] for calculating $P(W)$. This corpus contained English words with corresponding frequencies summarized from over 22 M words of written English. According to Nation et al. [27], this was sufficient to cover over 90% of the common English words. We reported top-1 to top-5 word-level accuracies.

Step 1: Keyboard Layout

Figure 3 showed that the touch points significantly overlapped in one-handed touch-typing. In practice, ambiguous keyboards could significantly increase the key-level accuracy by enlarging keys, which has been successfully implemented on various scenarios (e.g. tablet [21] and one-handed text entry [36]). However, this would also increase input ambiguity, especially for words with low clarity [38]. Therefore, we tested whether employing ambiguous keyboard could benefit the input performance. Specifically, we focused on testing different underlying models for input prediction, which is implicit to users. Therefore, the data from Study 1 can be used assuming that the displayed layout is always QWERTY (which is common).

According to Table 1, the touch spreads' width was about twice as its height. Therefore, we tested two keyboard layouts: QWERTY and Bi-Key. The Bi-Key layout followed the QWERTY layout, but grouped two horizontal keys together, yielding 14 keys (see Figure 8). For QWERTY, we fitted a Gaussian distribution for each individual keys to calculate $P(I_i|W_i)$. For Bi-Key, we set the center of each Gaussian distribution at the centroid of the merged key (calculated by averaging all touch points for the key). As we will show in Table 6, the size of a merged key was approximately $1.8cm \times 1.6cm$. Therefore, we used the standard deviations from [28], as it was measured when users performed two-handed touch-typing on 1.9 cm-wide keys. We scaled the standard deviations in both x and y dimensions according to our fitted keyboard size (i.e., $\times 1.6/1.9$).

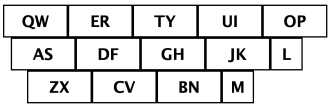


Figure 8: The Bi-key layout.

Table 5 showed the prediction accuracy of different keyboard models (BIKEY-O-ABS and QWERTY-O-ABS). As expected, personalized keyboard models outperformed general keyboard models for both Bi-Key and QWERTY. Interestingly, when using general models, Bi-Key yielded a higher top-1 accuracy than QWERTY (62.1% vs. 52.7%), while when using personalized models, the advantage was reversed (65.2% vs. 74.7%). We speculated that due to the large key size of Bi-Key, it was able to, to some extent, compensate for the difference in touch point distribution among participants. Therefore, its performance was not so affected by personalization. However, as the performance of QWERTY keyboard relied heavily on the precision of the touch model, it could be significantly benefited by personalization [5, 9, 18, 28]. As personalized QWERTY keyboard achieved the best performance among the four keyboards in the top two rows (top-1 accuracy: 74.7%,

	General					Personalized				
	Top-1	Top-2	Top-3	Top-4	Top-5	Top-1	Top-2	Top-3	Top-4	Top-5
BIKEY-O-ABS	62.1 (8.4)	73.8 (8.0)	78.7 (7.7)	81.4 (7.2)	83.2 (6.5)	65.2 (7.2)	77.2 (6.5)	81.4 (5.7)	83.7 (5.2)	85.1 (4.9)
QWERTY-O-ABS	52.7 (18.2)	63.2 (17.5)	68.4 (16.6)	72.3 (15.5)	75.1 (14.2)	74.7 (9.0)	85.8 (6.9)	89.6 (5.6)	91.7 (4.4)	93.4 (3.7)
QWERTY-W-ABS	60.2 (12.7)	71.1 (12.5)	76.3 (11.4)	79.4 (10.4)	81.3 (9.6)	74.2 (9.8)	84.9 (7.8)	88.9 (6.5)	91.1 (5.5)	92.6 (4.8)
QWERTY-W-REL	74.4 (6.2)	86.4 (4.7)	90.5(3.9)	92.6 (3.0)	93.9 (2.5)	76.3 (5.4)	87.5 (3.8)	91.5 (2.8)	93.4 (2.1)	94.6 (1.5)

Table 5: Prediction accuracy of different keyboard models (%), standard deviations are showed in parenthesis.

SD = 9.0%; top-5 accuracy: 93.4%, SD = 3.7%), we chose QWERTY layout.

Step 2: Incorporating Implicit Pressure Data

Feasibility of Key Row Prediction

Existing works have showed that incorporating fingering information could significantly benefit the input prediction performance [21, 40]. In Study 1, we also found that pressure image data could be helpful for distinguishing touches for different keys (see Table 4). However, keys on different rows yielded more distinguishable features than with different fingers. Besides, the key-to-finger mapping was different among participants (see Table 2). Therefore, we focused on distinguishing keys from different rows.

Specifically, we built a naïve Bayesian classifier for key row prediction, with the three features in Table 4b, as well as three other features: x and y coordinate of the touch, and $\hat{Pre}_{mean,t}$, which is the maximum of the mean pressure of each frame. We defined $\mathbf{F} = [F_1, F_2, ...F_6]^T$ as the feature vector for row prediction, where F_1–F_6 denoted the above six features respectively. For each row $R_i, i \in [1,4]$, the classifier calculated:

$$P(R_i|\mathbf{F}) = \frac{P(\mathbf{F}|R_i) \times P(R_i)}{P(\mathbf{F})} = \frac{\prod_{j=1}^{6} P(F_j|R_i) \times P(R_i)}{P(\mathbf{F})} \quad (5)$$

where $P(F_j|R_i)$ was modelled using Gaussian distribution, and $P(R_i)$ was calculated based on the language model.

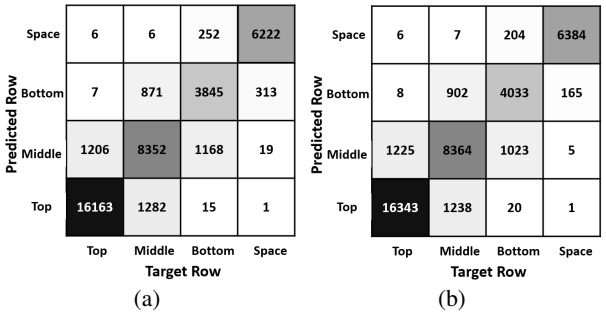


Figure 9: Confusion matrix of (a) general and (b) personalized classifier.

Figure 9 showed the classification accuracy of this classifier on the data from Study 1. The F-1 score of the general and personalized classifier was 0.85 and 0.87 respectively, confirming the feasibility of key row prediction. As expected, personalized classifier achieved slightly higher accuracy than the general classifier (88.0% vs 87.1%).

Augmented Bayesian Decoder

We then incorporated this classifier into the decoding algorithm, and proposed an augmented Bayesian decoder. When entering each word, we denote $\mathbf{F}_i = [F_{i1}, F_{i2}, ..., F_{i6}]^T$ as the feature vector of the ith touch input. Then, instead of calculating $P(W|I)$, we calculated $P(W|I,\mathbf{F})$. According to Bayesian theorem, this is equivalent to calculating

$$P(I,\mathbf{F}|W) \times P(W) = P(I|W) \times P(\mathbf{F}|W) \times P(W) \quad (6)$$

where

$$P(\mathbf{F}|W) = \prod_{i=1}^{n} P(\mathbf{F}_i|W_i) = \prod_{i=1}^{n} \prod_{j=1}^{6} P(\mathbf{F}_{ij}|R_k) \quad (7)$$

Here, R_k is the corresponding row of W_i, $k \in \{1,2,3\}$ (e.g., R_1 for 't'). By augmenting the Bayesian decoder with pressure data, the decoder was expected to leverage key row information to achieve higher performance.

We verified the performance of this decoder using simulation. Table 5 (QWERTY-W-ABS) showed the top-1 to top-5 accuracy. For general keyboards, augmented Bayesian decoder increased the top-1 and top-5 accuracy from 52.7% and 75.1% to 60.2% and 81.3% respectively. This confirmed that the information from pressure image could help improving input prediction. Surprisingly, the performance of personalized keyboard remained unchanged, which was still higher than general models.

Step 3: Absolute vs. Relative Model

Validity of Relative Model

Previous works have suggested that users were more likely to exhibited a relative model when performing eyes-free typing with muscle memory [23, 28]. In Study 1, participants performed touch-typing on a touchpad while looking at the laptop screen. Therefore, we also validated whether users' touch point exhibited a relative keyboard model.

Similar with [23], we fitted a "standard keyboard" to the users' typing data. Table 6 showed the fitted keyboard parameters. For all the keyboard models, the key size in y axis was about 1.7 times of that in x axis, which was consistent with existing touch models in eyes-free typing [23]. For personalized keyboards, the R^2 of absolute and relative models all reached 0.88, confirming that the emergent keyboard was very similar to QWERTY layout. No significant difference in R^2 was found between absolute and relative models ($F_{1,11} = 1.77, p = .21$),

or between x and y axis ($F_{1,11} = 0.10, p = .76$). These results confirmed that users' one-handed touch-typing behavior contained both absolute and relative components.

	Personalized				General			
Model	**Absolute**		**Relative**		**Absolute**		**Relative**	
Axis	**X**	**Y**	**X**	**Y**	**X**	**Y**	**X**	**Y**
Key Size (mm)	9.3 (1.5)	16.3 (2.0)	9.9 (1.6)	16.8 (1.7)	9.2	16.1	9.9	13.3
Offset (mm)	19.7 (5.8)	177.2 (10.5)	0.2 (0.4)	0.0 (0.2)	23.3	181.8	0.2	-0.1
R^2 Value	0.90 (0.05)	0.88 (0.04)	0.90 (0.03)	0.91 (0.02)	0.89	0.87	0.88	0.89

Table 6: Fitted keyboard parameters. Standard deviations are showed in parenthesis. Note the offset has different meanings in the absolute and relative model.

Augmented Markov-Bayesian Decoder

Based on this finding, we further incorporated our augmented Bayesian decoder (Equation 6) with the relative keyboard model (Equation 3), and proposed the augmented Markov-Bayesian decoder. The simulated performance of the decoder was showed in Table 5 (QWERTY-W-REL). Comparing with the decoder with absolute keyboard model, relative model further improved the top-1 and top-5 accuracy from 60.2% and 81.3% to 74.4% and 93.9% respectively. Again, the performance of the personalized keyboards was not so affected by using a relative keyboard model. Although RM-ANOVA found significant difference between general and personalized QWERTY-W-REL models ($F_{1,11} = 13.4, p < .01$), the performance of these two models were very close. Considering the applicability of different models, we chose the general model as the final optimized decoder.

STUDY 2: TYPING PERFORMANCE IN REAL TASKS

So far, we have improved the statistical decoding algorithm of indirect one-handed touch-typing. In this section, we developed PalmBoard, a technique with the final optimized algorithm (QWERTY-W-REL with general model), and evaluated users' typing performance with PalmBoard against two baseline techniques in real text entry tasks.

Participants and Apparatus

We recruited another 10 participants from the campus (9 male, 1 female) with a mean age of 21.0 (SD = 0.5). All participants were right-handed, and used touchscreen devices at a daily basis, but none of them tried one-hand touch-typing before. Each participant was compensated $10. We used the same apparatus as in Study 1.

Experiment Platform

The experiment platform was similar as in Study 1, except that during typing, the platform showed five candidate words for the user to select from, in the order of probability (see Figure 10). In addition to the keyboard algorithms, we also designed gestures for PalmBoard to support word selection and deletion. During typing, users could swipe any finger from left to right to circularly select words from the candidate list, and tap the thumb to confirm the selection. The system would automatically append a space key upon each selection. Also, users could swipe left with any finger to delete the last input word.

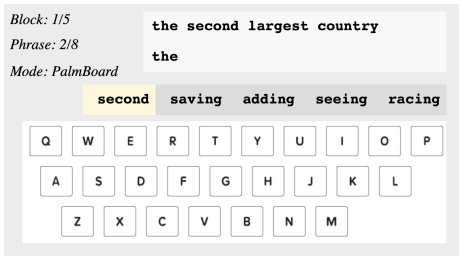


Figure 10: Experiment Platform.

Experiment Design

In order to understand the performance of PalmBoard with more details, we also tested two baseline techniques: *General* and *Rectangle*. These two smart keyboard techniques both used Equation 1 and Equation 2 to perform input prediction. The three techniques all shared the same user interface and interaction design (palm rejection, word selection and deletion), and used the same unigram word-level language model with 10,000 words as in the previous section.

For touch model, all techniques used the general model rather than the personalized model. *General* used the QWERTY-O-ABS in Table 5 for input prediction. *Rectangle* used a standard QWERTY keyboard layout, which may better fit the layout on physical keyboards. We maximized the size of the *Rectangle* keyboard (13 cm) given the restriction of the size of our input device (13.8 cm in width). For each key, the mean of the Gaussian distribution was set to the key center, and the standard deviation was empirically set to 6.93 mm for both x and y axis according to Table 1 and Table 6.

Procedure

Each participant completed three sessions of text entry tasks, each corresponding to one keyboard technique. In each session, they entered 40 phrases from the T-40 phrase set in [38], evenly divided into 5 blocks. The order of different sessions and phrases were randomized, and a one-minute break was enforced between different blocks. During typing, the participant was seated in front of a desk, and used their dominant hand to enter text on the touchpad, while looking at the experiment platform on the laptop screen. They were instructed to type "as quickly and as accurately as possible", and were allowed to rest their palms onto the pad. If they found any error during typing, they could choose to delete and retype it, or leave it uncorrected. After the study, interviews and questionnaires were carried out to gather their subjective feedback.

Results

Input Speed

Figure 11 showed the text entry speed of different techniques. Across different blocks, the average speed of PalmBoard, *General* and *Rectangle* was 32.8 WPM (SD = 4.7), 28.4 WPM (SD = 4.9) and 23.5 WPM (SD = 4.7) respectively. RM-ANOVA found a significant difference between the input speed of different techniques ($F_{2,18} = 27.5, p < .0001$). PalmBoard achieved the highest input speed among the three techniques. The input speed was close to the one-hand typing speed on smartphones [4], and was about 79% of the two-handed touch-typing speed

on touchscreens [28]. Meanwhile, it was much faster than existing one-handed text entry techniques (e.g., [13, 36]).

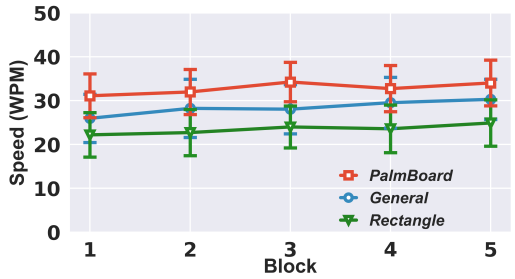


Figure 11: Input speed of different techniques, error bar showed one standard deviation.

In block 1, the input speed of PalmBoard was 31.1 WPM (SD = 5.0), and it increased to 34.0 WPM (SD = 5.2) in block 5. A significant effect of *Block* was found on input speed ($F_{4,36} = 3.97, p < .01$), indicating a learning effect. In block 5, the three participants with highest input speed even reached an average speed of 40.2 WPM (SD = 4.4), confirming the potential of PalmBoard.

Error Rate

We measured the uncorrected error rate using CER, which is a common metric for evaluating smart keyboard techniques (e.g., [32, 29]). CER could be interpreted as the minimum number of insertions, deletions and substitutions that are required to transform the input string into the target string, divided by the number of characters in the target string.

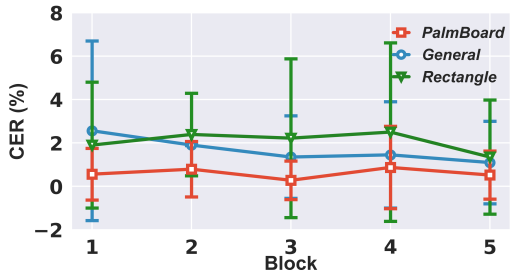


Figure 12: Error rate of different techniques, error bar showed one standard deviation.

Figure 12 showed the error rate of different techniques. Across different blocks, the average error rate of PalmBoard, *General* and *Rectangle* was 0.6% (SD = 0.6%), 1.7% (SD = 2.2%) and 1.9% (SD = 24%) respectively. No significant effect of *technique* ($F_{2,18} = 1.70, p = .21$) or *Block* ($F_{4,36} = 0.84, p = .51$) on error rate was found.

Interaction Statistics

We looked more into how users interacted with the techniques. Table 7 showed the ratio of word-affecting actions for different techniques. For all the three techniques, the most frequent operation was *Match*, i.e., select the first candidate. PalmBoard yielded the highest ratio of *Match* among the three techniques, while *Rectangle* was the lowest. This was consistent with the results on input speed (see Figure 11) and error rate (see Figure 12). The ratio of *Selecting* other candidate words were about 20% for all the three techniques. *Undo* and *Delete* means the user deleted the input before and after word selection, respectively. PalmBoard yielded the lowest ratio of *Undo* and *Delete*, while *Rectangle* was the highest. RM-ANOVA found a significant effect of *technique* on the ratio of *Match-yes* ($F_{2,18} = 11.5, p < .001$), *Select-yes* ($F_{2,18} = 8.79, p < .01$) and *Undo* ($F_{2,18} = 9.39, p < .01$), but not on *Delete* ($F_{2,18} = 0.22, p = .80$). This suggested that PalmBoard was more effective in predicting the user's input, allowing users to successfully select the candidate words more often. Therefore, users could achieve higher input speed with less retype when using PalmBoard.

	Match		Select		Undo	Delete
Correct	Yes	No	Yes	No		
PalmBoard	75.4	0.6	19.0	0.1	3.4	1.5
General	68.1	0.9	22.3	0.5	6.6	1.6
Rectangle	60.0	1.3	26.0	0.4	10.5	1.8

Table 7: Ratio of word-affecting actions (%) for different techniques. *Correct* indicates whether the selected word was identical to the intended word.

Subjective Ratings

We used the same questionnaire as in Study 1 to gather user feedback. Figure 13 showed the ratings from the participants.

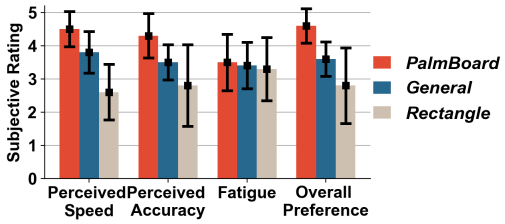


Figure 13: Subjective ratings from the participants. 1 means the most negative and 5 means the most positive. Error bar shows one standard deviation.

PalmBoard received the highest rating for all the dimensions, while *Rectangle* received the lowest rating. As all three techniques allowed users to rest their palms on the touchpad, the scores on fatigue were very close. Friedman test found a main effect of *technique* on perceived speed ($\chi^2(2) = 16.0, p < .001$), perceived accuracy ($\chi^2(2) = 8.0, p < .05$), and overall preference ($\chi^2(2) = 13.5, p < .001$), but not on fatigue ($\chi^2(2) = 0.29, p = .87$).

DISCUSSION

In this paper, we investigate how to incorporate implicit touch pressure into text decoding. We not only modelled users' typing behavior in one-handed touch-typing scenario, but also proposed applicable technique for real use. We now discuss some insights from our results.

Effectiveness of Leveraging Implicit Touch Pressure

The pressure image data played an important role in the algorithm of PalmBoard. First, incorporating the touch pressure data substantially improved the text entry decoding: it improved the top-1 decoding accuracy from 52.7% to 75.1% (Table 5). Second, touch pressure was also utilized to identify unintentional touch (e.g., palm resting), which was crucial for successfully supporting indirect multi-finger typing.

Although we focused on indirect text entry scenarios, we expected that implicit pressure could benefit direct text entry. For example, pressure data could help rejecting palm touches on tabletop [28], and correct injection errors from accidental touches (which may be lighter than common touches). Additionally, users will likely exhibit distinct touch pressure patterns for different keys, because they type different keys with different fingers and may apply different touch pressure on different locations. Leveraging such patterns could improve the decoding accuracy, following the similar principle proposed in this paper.

In addition to indirect and direct touch typing, implicit pressure may benefit other touch input scenarios. For example, touch pressure could be used to infer finger orientation to improve pointing precision [33], or infer the input finger to support finger identification based interaction [16, 17].

Typing Behavior in One-Handed Touch-Typing

The study 1 showed that users could well perform one-handed touch-typing. Participants achieved about 34 WPM input speed, which was 80% of the two-handed touch-typing speed [28]. Allowing the users to rest their palms not only reduced fatigue, but also encouraged them to leverage more fingers (see Table 2).

The touch point distribution of one-handed touch-typing has significant overlap for different keys, implying a very noisy input. However, the collective centroids of touch points followed well with QWERTY layout (see Table 6). Interestingly, consistent with existing work [8], users adopted a wide variety of key-to-finger mapping strategies, which led to different perceived keyboard layouts (see Figure 4). These results suggested that users could transfer the muscle memory on physical keyboards to one-handed touch-typing.

Feasibility of One-Handed Touch-Typing

Our investigation showed that one-handed touch-typing was feasible and promising. Although the classical Bayesian decoder could only reach a top-1 accuracy of 52.7% Table 5, leveraging the pressure image data and using a relative keyboard model could both benefit the input prediction performance. By using an augmented Markov-Bayesian decoder, the top-1 accuracy could be increased to 74.4%, which was competitive with personalized keyboard models. In Study 2, participants achieved an average speed of 32.8 WPM, which was close to their typing speed under "ideal" condition (see Figure 2). This implied that PalmBoard could effectively compensate for the input imprecision, allowing the users to type fluently without correcting much errors (see Table 7).

We envision PalmBoard is a one-handed text entry technique that works on a broad range of touch surfaces (e.g., tabletop), not limited to touchpads, although touch surfaces with constrained size (e.g., touchpad) will particularly benefit from it. Besides, PalmBoard opens up chances to explore "one-hand for text entry + one-hand for other interaction (e.g., controlling cursor position)" scenarios in future work. Supporting one-handed text entry frees the other hand for concurrent tasks and provides chances for two-hand cooperation.

LIMITATION AND FUTURE WORK

We now discuss the limitations of this work, which we also see as opportunities for future work. First, we only verified the feasibility of implicit pressure data in one-handed touch-typing scenarios. It is possible that direct typing scenario will affect users' typing behavior due to visual feedback. And using different typing postures will also yield different features. Therefore, it is worthy to validate our results in direct typing scenarios, and with more typing postures (e.g. two-handed touch-typing).

Second, our augmented Markov-Bayesian decoder only extracted simple features from the pressure data. It is worthy to test models with more features to further improve the performance of the decoder. In Study 1, we combined data from both conditions for analysis, as the difference was very small (e.g. <10% for T and S_t). However, if the decoder could distinguish these two conditions based on more features, it will be able to further improve the prediction accuracy by using posture-specific touch models.

Third, the proposed decoder could only correct substitution errors. As insertion, deletion and transposition are also major kinds of typing error, especially when performing touch-typing with multiple fingers. Therefore, it is worthwhile to further improve the decoder to account for these kinds of errors.

CONCLUSION

In this paper, we explored the feasibility of incorporating implicit touch pressure data to improve statistical decoding in indirect text entry. Focusing on one-handed touch-typing scenario, we first examined users' typing behavior in a user study. We found that users exhibited distinct key-to-finger mappings, and implicit touch pressure feature was distinguishable when hitting different keys. Based on simulations on the collected data, we found that a Markov-Bayesian decoder incorporated with pressure data could significantly improve the input prediction accuracy than classical Bayesian decoder. In evaluation, our prototype PalmBoard reached an average speed of 33 WPM. Expert users could even reach 40 WPM. We concluded that implicit touch pressure was effective for text entry in indirect touch scenarios.

ACKNOWLEDGEMENT

This work is supported by the National Key Research and Development Plan under Grant No. 2018YFB1005000, the Natural Science Foundation of China under Grant No. 61672314, 61572276 and 61902208, China Postdoctoral Science Foundation under Grant No. 2019M660647, and also by Beijing Key Lab of Networked Multimedia.

REFERENCES

[1] 2016. A note on calculating text entry speed. (2016). http://www.yorku.ca/mack/RN-TextEntrySpeed.html.

[2] 2016. American National Corpus. (2016). http://www.americannationalcorpus.org/OANC/index.html.

[3] R.U. Ayres and K. Martinás. 2005. *On the Reappraisal of Microeconomics: Economic Growth and Change in a Material World*. Edward Elgar. https://books.google.com/books?id=ksxK7J95IF8C

[4] Shiri Azenkot and Shumin Zhai. 2012. Touch Behavior with Different Postures on Soft Smartphone Keyboards. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '12)*. ACM, New York, NY, USA, 251–260. DOI: http://dx.doi.org/10.1145/2371574.2371612

[5] Tyler Baldwin and Joyce Chai. 2012. Towards Online Adaptation and Personalization of Key-target Resizing for Mobile Devices. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces (IUI '12)*. ACM, New York, NY, USA, 11–20. DOI: http://dx.doi.org/10.1145/2166966.2166969

[6] Xiaojun Bi, Yang Li, and Shumin Zhai. 2013. FFitts Law: Modeling Finger Touch with Fitts' Law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 1363–1372. DOI: http://dx.doi.org/10.1145/2470654.2466180

[7] 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. DOI: http://dx.doi.org/10.1145/1613858.1613870

[8] Anna Maria Feit, Daryl Weir, and Antti Oulasvirta. 2016. How We Type: Movement Strategies and Performance in Everyday Typing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4262–4273. DOI:http://dx.doi.org/10.1145/2858036.2858233

[9] Leah Findlater and Jacob Wobbrock. 2012. Personalized Input: Improving Ten-finger Touchscreen Typing Through Automatic Adaptation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 815–824. DOI: http://dx.doi.org/10.1145/2207676.2208520

[10] Leah Findlater, Jacob O. Wobbrock, and Daniel Wigdor. 2011. Typing on Flat Glass: Examining Ten-finger Expert Typing Patterns on Touch Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2453–2462. DOI: http://dx.doi.org/10.1145/1978942.1979301

[11] Mayank Goel, Leah Findlater, and Jacob Wobbrock. 2012. WalkType: Using Accelerometer Data to Accomodate Situational Impairments in Mobile Touch Screen Text Entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2687–2696. DOI:http://dx.doi.org/10.1145/2207676.2208662

[12] Mayank Goel, Alex Jansen, Travis Mandel, Shwetak N. Patel, and Jacob O. Wobbrock. 2013. ContextType: Using Hand Posture Information to Improve Mobile Touch Screen Text Entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 2795–2798. DOI:http://dx.doi.org/10.1145/2470654.2481386

[13] Jun Gong, Zheer Xu, Qifan Guo, Teddy Seyed, Xiang 'Anthony' Chen, Xiaojun Bi, and Xing-Dong Yang. 2018. WrisText: One-handed Text Entry on Smartwatch Using Wrist Gestures. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 181, 14 pages. DOI: http://dx.doi.org/10.1145/3173574.3173755

[14] 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. DOI: http://dx.doi.org/10.1145/502716.502753

[15] 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. DOI: http://dx.doi.org/10.1145/2858036.2858242

[16] Aakar Gupta, Muhammed Anwar, and Ravin Balakrishnan. 2016. Porous Interfaces for Small Screen Multitasking Using Finger Identification. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 145–156. DOI: http://dx.doi.org/10.1145/2984511.2984557

[17] Aakar Gupta and Ravin Balakrishnan. 2016. DualKey: Miniature Screen Text Entry via Finger Identification. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 59–70. DOI: http://dx.doi.org/10.1145/2858036.2858052

[18] Johan Himberg, Jonna Häkkilä, Petri Kangas, and Jani Mäntyjärvi. 2003. On-line Personalization of a Touch Screen Based Keyboard. In *Proceedings of the 8th International Conference on Intelligent User Interfaces (IUI '03)*. ACM, New York, NY, USA, 77–84. DOI: http://dx.doi.org/10.1145/604045.604061

[19] Min-Chieh Hsiu, Da-Yuan Huang, Chi An Chen, Yu-Chih Lin, Yi-ping Hung, De-Nian Yang, and Mike Chen. 2016. ForceBoard: Using Force As Input Technique on Size-limited Soft Keyboard. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI '16)*. ACM, New York, NY, USA, 599–604. DOI: http://dx.doi.org/10.1145/2957265.2961827

[20] Sunjun Kim, Jeongmin Son, Geehyuk Lee, Hwan Kim, and Woohun Lee. 2013. TapBoard: Making a Touch Screen Keyboard More Touchable. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 553–562. DOI: http://dx.doi.org/10.1145/2470654.2470733

[21] 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. DOI: http://dx.doi.org/10.1145/2047196.2047257

[22] Yiqin Lu, Chun Yu, Shuyi Fan, Xiaojun Bi, and Yuanchun Shi. 2019. Typing on Split Keyboards with Peripheral Vision. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 200, 12 pages. DOI:http://dx.doi.org/10.1145/3290605.3300430

[23] Yiqin Lu, Chun Yu, Xin Yi, Yuanchun Shi, and Shengdong Zhao. 2017. BlindType: Eyes-Free Text Entry on Handheld Touchpad by Leveraging Thumb's Muscle Memory. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 2, Article 18 (June 2017), 24 pages. DOI:http://dx.doi.org/10.1145/3090083

[24] 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. DOI: http://dx.doi.org/10.1145/765891.765971

[25] 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. DOI:http://dx.doi.org/10.1145/2407796.2407810

[26] 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. DOI:http://dx.doi.org/10.1145/1520340.1520693

[27] Paul Nation and Robert Waring. 1997. Vocabulary size, text coverage and word lists. *Vocabulary: Description, acquisition and pedagogy* 14 (1997), 6–19.

[28] Weinan Shi, Chun Yu, Xin Yi, Zhen Li, and Yuanchun Shi. 2018. TOAST: Ten-Finger Eyes-Free Typing on Touchable Surfaces. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 33 (March 2018), 23 pages. DOI:http://dx.doi.org/10.1145/3191765

[29] R. William Soukoreff and I. Scott MacKenzie. 2003. Metrics for Text Entry Research: An Evaluation of MSD and KSPC, and a New Unified Error Metric. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 113–120. DOI: http://dx.doi.org/10.1145/642611.642632

[30] Keith Vertanen, Crystal Fletcher, Dylan Gaines, Jacob Gould, and Per Ola Kristensson. 2018. The Impact of Word, Multiple Word, and Sentence Input on Virtual Keyboard Decoding Performance. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 626, 12 pages. DOI: http://dx.doi.org/10.1145/3173574.3174200

[31] Keith Vertanen, Dylan Gaines, Crystal Fletcher, Alex M. Stanage, Robbie Watling, and Per Ola Kristensson. 2019. VelociWatch: Designing and Evaluating a Virtual Keyboard for the Input of Challenging Text. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 591, 14 pages. DOI: http://dx.doi.org/10.1145/3290605.3300821

[32] Keith Vertanen, Haythem Memmi, Justin Emge, Shyam Reyal, and Per Ola Kristensson. 2015. VelociTap: Investigating Fast Mobile Text Entry Using Sentence-Based Decoding of Touchscreen Keyboard Input. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 659–668. DOI: http://dx.doi.org/10.1145/2702123.2702135

[33] Feng Wang, Xiang Cao, Xiangshi Ren, and Pourang Irani. 2009. Detecting and Leveraging Finger Orientation for Interaction with Direct-touch Surfaces. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 23–32. DOI: http://dx.doi.org/10.1145/1622176.1622182

[34] Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. 2014. Uncertain Text Entry on Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2307–2316. DOI:http://dx.doi.org/10.1145/2556288.2557412

[35] 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. DOI: http://dx.doi.org/10.1145/2468356.2468591

[36] Pui Chung Wong, Kening Zhu, and Hongbo Fu. 2018. FingerT9: Leveraging Thumb-to-finger Interaction for Same-side-hand Text Entry on Smartwatches. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 178, 10 pages. DOI: http://dx.doi.org/10.1145/3173574.3173752

[37] Hui-Shyong Yeo, Xiao-Shen Phang, Taejin Ha, Woontack Woo, and Aaron Quigley. 2017. TiTAN: Exploring Midair Text Entry Using Freehand Input. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 3041–3049. DOI:http://dx.doi.org/10.1145/3027063.3053228

[38] Xin Yi, Chun Yu, Weinan Shi, Xiaojun Bi, and Yuanchun Shi. 2017b. Word Clarity As a Metric in Sampling Keyboard Test Sets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4216–4228. DOI: http://dx.doi.org/10.1145/3025453.3025701

[39] Xin Yi, Chun Yu, Weinan Shi, and Yuanchun Shi. 2017a. Is it too small?: Investigating the performances and preferences of users when typing on tiny QWERTY keyboards. *International Journal of Human-Computer Studies* 106 (2017), 44–62.

[40] Xin Yi, Chun Yu, Mingrui Zhang, Sida Gao, Ke Sun, and Yuanchun Shi. 2015. ATK: Enabling Ten-Finger Freehand Typing in Air Based on 3D Hand Tracking Data. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 539–548. DOI: http://dx.doi.org/10.1145/2807442.2807504

[41] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. 2017. Tap, Dwell or Gesture?: Exploring Head-Based Text Entry Techniques for HMDs. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4479–4488. DOI: http://dx.doi.org/10.1145/3025453.3025964

[42] Mingyuan Zhong, Chun Yu, Qian Wang, Xuhai Xu, and Yuanchun Shi. 2018. ForceBoard: Subtle Text Entry Leveraging Pressure. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 528, 10 pages. DOI:http://dx.doi.org/10.1145/3173574.3174102

[43] Suwen Zhu, Tianyao Luo, Xiaojun Bi, and Shumin Zhai. 2018. Typing on an Invisible Keyboard. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 439, 13 pages. DOI: http://dx.doi.org/10.1145/3173574.3174013

[44] Suwen Zhu, Jingjie Zheng, Shumin Zhai, and Xiaojun Bi. 2019. I'sFree: Eyes-Free Gesture Typing via a Touch-Enabled Remote Control. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 448, 12 pages. DOI: http://dx.doi.org/10.1145/3290605.3300678