UOIT Keyboard: A Constructive Keyboard for Small Touchscreen Devices

Tamer AbuHmed, Kyunghee Lee, and DaeHun Nyang

***Abstract*—Many techniques have been proposed for reducing errors during text input on touchscreens. However, the majority of these techniques suffer from the same limitation, i.e., the keyboard keys are overcrowded on a small screen, resulting in high error rates and slow text inputs. To address this situation and resolve the problems associated with overcrowdedness, we introduce a new text-entry method called the "UOIT keyboard." The idea behind the UOIT keyboard is to compose letters using "drawing-like typing" on the UOIT keyboard, which has 13 large keys that replace the 26 small keys that exist in the QWERTY keyboard. We describe the design, keys, and mechanism of the UOIT keyboard. A 24-participant user study was conducted to evaluate the speed and accuracy of the proposed entry method as compared with the QWERTY and multitap entry methods. As part of the evaluation, a questionnaire was used to collect participants' preferences. The UOIT keyboard has a mean entry speed of 11.3 words/min. The UOIT keyboard significantly reduces the typing errors with 3.8% total error rate comparing with 11.2% and 16.3% for QWERTY and multitap entry methods, respectively.**

***Index Terms*—Mobile computing, computer–human interaction, entry method, input text, keyboard, touch interface.**

I. Introduction

Virtual keyboards, also known as "software keyboards" or "on-screen keyboards," have been commonly used for input in portable devices such as smartphones and tablets with touchscreens that do not have physical keyboards. These devices usually utilize a graphically rendered image of an on-screen keyboard for text input [1]. One of the main features of software keyboards is the flexibility of customization to support multiple languages, key layouts, and screen orientations. However, on-screen keyboards have several shortcomings, including no tactile feedback for enhancing touch-typing accuracy [2] and the small size of the keys. When the keyboard appears, it takes up part of the screen during the text input. As a result, there is only a limited amount of space on the screen for the keyboard. This results in an overcrowded keyboard with small keys. This overcrowdedness causes users not only to type more slowly on a software keyboard than on a physical keyboard, but also leads to more typing errors [3], [4]. Usually, smartphone users prefer to use their fingers rather than a stylus to enter text [5]. However, when fingers are used with small keys, they occlude the keys; therefore, the input rate decreases and the error rate increases. A number of strategies have been pursued in the design of virtual keyboards for mitigating typing errors and increasing typing speed. These strategies are based on character frequencies [6], maintaining distance between the character pairs, resizing the keys [7]–[9], and visually highlighting the keys [10].

In this study, we tackle the problem of space limitation of a software keyboard by proposing a new text-entry method for small-sized touchscreen devices. The idea is to input English alphabets using drawing-like typing. Unlike the Graffiti [11] and free-hand writing [12] techniques, which are even slower and more error-prone than QWERTY virtual keyboards for ordinary users, our entry method uses deterministic key combinations for the constructions of letters. As a result, our method requires no more than two keystrokes per letter and avoids the letter-recognition problems associated with Graffiti writing and free-hand writing [13].

The motivation behind the new constructive keyboard is to provide users with an intuitive, simple, and uncrowded entry method. We aim to give users an experience that combines the ease of the QWERTY-keyboard-typing experience with the *intuitive construction* of characters from the Graffiti and free-hand writing techniques. The design is shown in Fig. 1(b). The naming convention for the new keyboard is similar to the naming convention for the QWERTY keyboard. The keyboard is named the "UOIT keyboard," where U, O, I, and T are the letters in the top row. In this study, we address the usability of the proposed keyboard in terms of input speed, accuracy, and task load. Along with the QWERTY keyboard [see Fig. 1(a)], we used a multitap keyboard [see Fig. 1(c)]. This multitap keyboard has the same number of keys as the UOIT keyboard and has an unorthodox combination of the half-Qwerty [14] layout and multitap input functionality. In the multitap keyboard, two letters on each key are assigned, and thus, a single tap inputs the first letter on the key and double tap inputs the second letter.

II. Related Work

We survey the text-entry systems for keyboard-based mobile devices. In [7], Faraj *et al.* suggest that the size of the keys in the QWERTY keyboard should be increased and found that users were faster and more accurate in their typing. In [8], Floodkey developed a method for resizing the keys by modeling the keyboard as a Voronoi diagram. Magnien *et al.* [10] suggested the use of visual clues such as bolding of the keys that are likely to be the next key based on previously typed letters. Although the key size and visual highlighting helped with typing accuracy, overcrowding was still a problem. In [4] and [15], key-target resizing approaches were developed to mitigate typing error. Key-target resizing algorithms change the size of the individual keys dynamically as the user types based on the language and user's typing behavior. By guessing more accurately, the

Manuscript received January 22, 2014; revised June 30, 2014, January 12, 2015, and April 15, 2015; accepted June 12, 2015. This work was supported by the Ministry of Science, ICT and Future Planning, Korea, under the Information Technology Research Center support program (IITP-2015-H8501-15-1008) supervised by the Institute for Information and communications Technology Promotion. This work was also supported by Inha University. *(Corresponding author: DaeHun Nyang.)*

T. AbuHmed and D. Nyang are with the School of Computer and Information Engineering, Inha University, Incheon 402-751, Korea (e-mail: tamer@inha.ac.kr; nyang@inha.ac.kr).

K. Lee is with the Department of Electrical Engineering, University of Suwon, Suwon 442-081, Korea (e-mail: khlee@suwon.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/THMS.2015.2449309

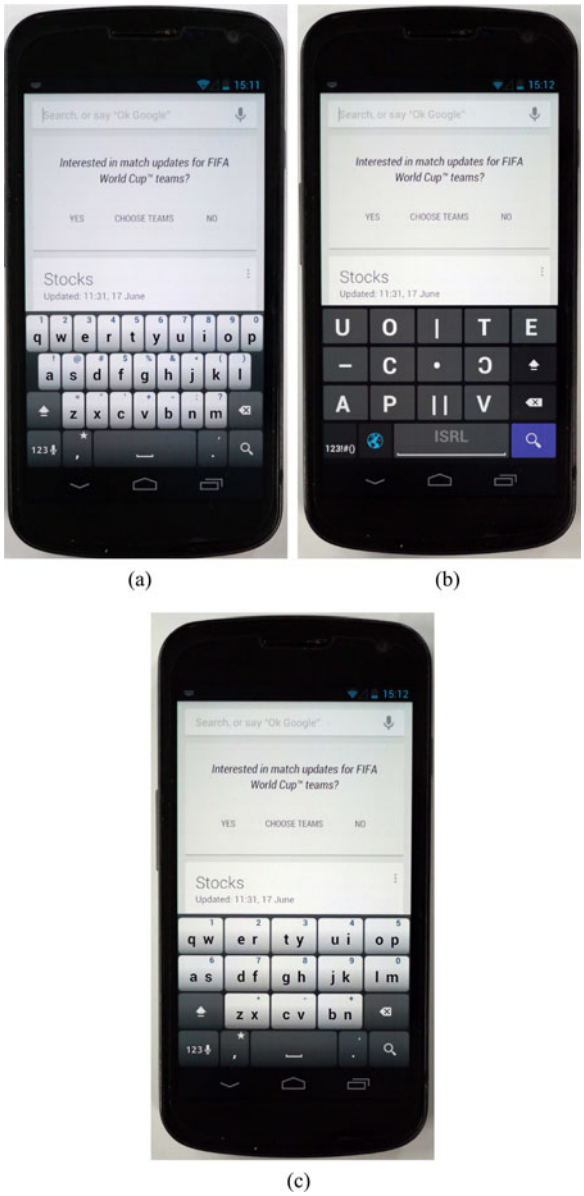


Fig. 1. (a) QWERTY, (b) UOIT, and (c) multitap keyboards on Samsung Galaxy Nexus SHW-M420K used for the study.

intent of the user and overall typing performance is improved. However, Himberg *et al.* [9] showed that the constant changing of key sizes was potentially distracting to users. Moreover, the results from applying the key-target resizing model varied from one user to another. A personalized touch model that focuses on the typing behavior of each user should be considered [9], [16]. In [17], Kwon *et al.* investigated estimating the regional error correction, which is a predictive text-entry method that activates the key corresponding to the actual activation point and also other keys within an activation area. Their work enhances the error rate compared with the conventional text-entry method. However, they found that it is hard to find an optimal solution to estimate the activation area as the performance of the regional error correction solution mainly depends on finding methods of optimizing the size and shape of the activation area

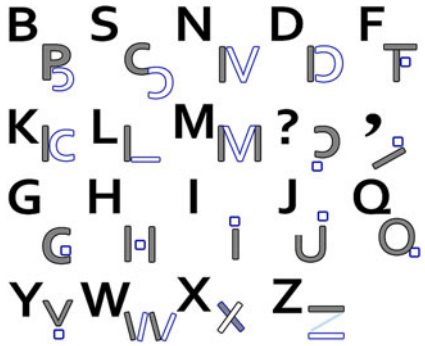


Fig. 2. Alphabet construction in UOIT. Drawing components shown in gray are the first input and those shown in white are the second input.

when the user taps his finger on the keys. Their work supports the use of large key sizes to enhance the performance of the regional error correction algorithms.

III. UOIT Constructive Approach

The design of the UOIT keyboard is inspired by the need for a less crowded text-entry method that allows for easier and more error-free tapping. Several constructive entry methods have been invented to assist with the entry of the valid Korean characters on a small screen. In the Korean language, one character or one syllabic block is constructed by placing two to five consonant and vowel letters horizontally and/or vertically within a small imaginary square. The number of consonants and vowels are 14 and 10, respectively, and the number of mathematically possible syllabic blocks is 11 172, more than can be accommodated by any size of screen.

For English letters, we divided the 26 letters into two groups: eight require one stroke and 18 require two strokes. For the 18-letter group, we broke each letter into two drawing segments and extracted the common drawing components: five symbolic keys ($||$, $-$, $\cdot$, $\supset$, $|$) and U, O, T, C, P, and V, where U, O, T, C, P, and V along with A and E are single-stroke letters. The 18 letters can be composed by combining the two drawing components in the correct order as illustrated in Fig. 2. Obviously, the number of strokes is increased in the UOIT keyboard, but the size of the keys is enlarged.

A. Keys of the UOIT Keyboard

The UOIT keyboard has 13 letter keys that can be used to express 31: 26 English letters and five common punctuation marks (i.e., period, question mark, comma, dash, and apostrophe). It uses only 41.9% ($=100 \times 13/(26+5)$) of the keys from the standard QWERTY keyboard. Six functional keys are also included (i.e., space, shift, number flip key, enter, keyboard switch, and delete key), as shown in Fig. 1(b). The number flip key at the bottom left of the UOIT keyboard is used to flip to input numbers and special symbols as in the standard QWERTY keyboard, whereas the keyboard switch key is to switch to another keyboard layout. Because the number of keys in the UOIT keyboard is cut approximately in half, each key can occupy twice the width of a key in the QWERTY keyboard, as

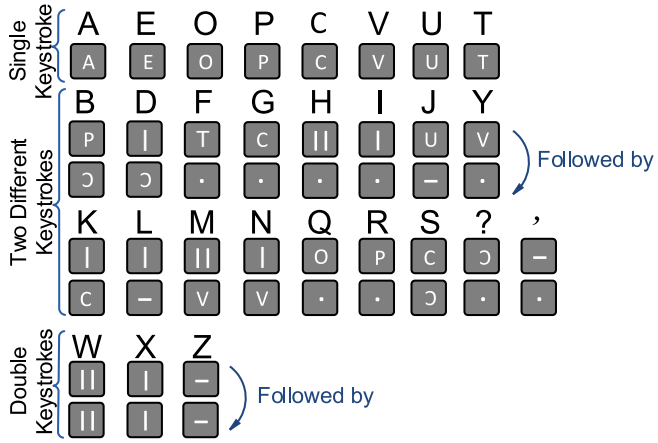


Fig. 3. Alphabet construction on UOIT keyboard, where the alphabet is categorized into three groups (i.e., letters with a single keystroke, letters with two different keystrokes as in the case of pressing C followed by dot key to construct G letter, and finally letters with double keystrokes as in the case of pressing | key twice to construct X letter).

shown in Fig. 1. These large keys help a user to experience exact tapping of intended keys.

The UOIT keyboard was designed in a way that the construction processes for individual letters bore similarities to the drawings of uppercase letters. In other words, most of the letters that require two keystrokes are constructed using two drawing components that imitate the shape of the letter in its uppercase form. As a result, the UOIT keyboard includes eight letters (i.e., A, C, E, O, P, T, V, U) and five symbol keys as shown in Fig. 1(b). The letters and symbols represent the drawing components for the construction of the 31 symbols. In choosing of the eight letters, we also considered the average number of strokes needed to input one character. The eight single-stroke characters are the most frequently used letters (considering also the role of drawing components). The sum of the frequencies for these eight letters is 45.8% [18]. Therefore, using the frequencies of these letters in English text, the average theoretical number of key strokes per letter in the UOIT keyboard is 1.51. Finally, to increase familiarity with UOIT, we arranged the functional keys such as the space, shift, delete, and number flip keys in positions as in the QWERTY keyboard.

B. Input Mechanism for UOIT Keyboard

The users of the UOIT-entry method can type any letter using either a single keystroke for the eight letters on the UOIT keyboard or using two keystrokes for the rest. Most letters that require two keystrokes are constructed using two drawing components that imitate the shape of the letter in its uppercase form (see Fig. 3). For example, pressing | followed immediately by − constructs "L" without requiring any delay or a kill key. The full list of combinations for UOIT is shown in Fig. 3.

Two exceptions are "i" and "j," which are constructed in a way that imitates their lowercase shapes to avoid ambiguity. If the letter 'i' were constructed by pressing only |, then sequence of keystrokes | and c can be interpreted as either "ic", or "k." In addition, the order of the keystrokes to construct the letter 'i' and 'j' is important to avoid ambiguity. For example, assume that 'i' was constructed by · followed by |. Then, the sequence of keystrokes c, ·, |, and c can be interpreted as either "gk," or "cic." However, if the letter 'i' is constructed by pressing |, then ·, the sequence of keystrokes c, |, ·, and c can only be interpreted as "cic." Therefore, we can eliminate any ambiguity in the letters construction.

The UOIT-entry mechanism uses a deterministic finite automaton (DFA) to determine whether or not the letter-input process is complete. This process is illustrated in Fig. 4. For example, when a user taps T, the input is not complete because there are two possibilities that can occur on the basis of the next input. If the next input is ·, the letter T is converted to F and the DFA is reset. Otherwise, if the next input is any key except the · key, the letter T is committed and the DFA returns to the starting state with the newly pressed key as the initial key. Thus, unlike the multitap-entry method [19], the UOIT-entry method does not require a kill key or a time-out before an attempt is made to enter the next key-based digraph. The UOIT keyboard currently supports common punctuation marks: 1) Pressing − followed by · constructs the comma (,). 2) With the shift key ON, pressing − followed by · constructs the apostrophe ('). 3) Pressing ⊃ followed by · constructs the question mark (?). 4) Pressing double space is used for period (.), which is a common technique in current smartphones.

It is convenient to input the commonly used punctuation marks without changing the keyboard layout. The standard QWERTY keyboard requires a change of the input mode to input the punctuation marks.

C. Productivity Features

We provide automatic completion, hint-highlighting, and alternative drawing components to provide UOIT users with better user experiences. We also tried to minimize finger-traveling distances.

1) *Automatic completion:* The UOIT design includes unique methods for fixing typographical errors. Missing the second key of a two strokes letter causes the automaton to be in an unfinished state. If the user presses | and continues typing without constructing any of the letters starting with | symbol (L, K, I, X, D, or N), then the UOIT keyboard automatically replaces this incomplete letter with I. A similar rule has been applied for ||, which is an H letter. If an English dictionary is available, typographical errors can be fixed by looking up the typed phrase.
2) *Minimum finger traveling distance:* The UOIT keyboard is designed to minimize the finger-traveling distance from one key to the next during text entry. The · is the most used key and is placed in the center of the keyboard with the related keys in adjacent positions. Fig. 5(a) and (b) illustrates finger movements for the compositions of various letters in the UOIT keyboard.
3) *Highlighting hints:* To aid with UOIT letter construction, we highlight the candidate-drawing components based on the first-drawing component (see Fig. 6). For example, when C is pressed, ⊃ is highlighted for S and · is highlighted for G.
4) *Alternative drawing components:* The UOIT keyboard can be designed using an optional alphabet construction if there is no typing

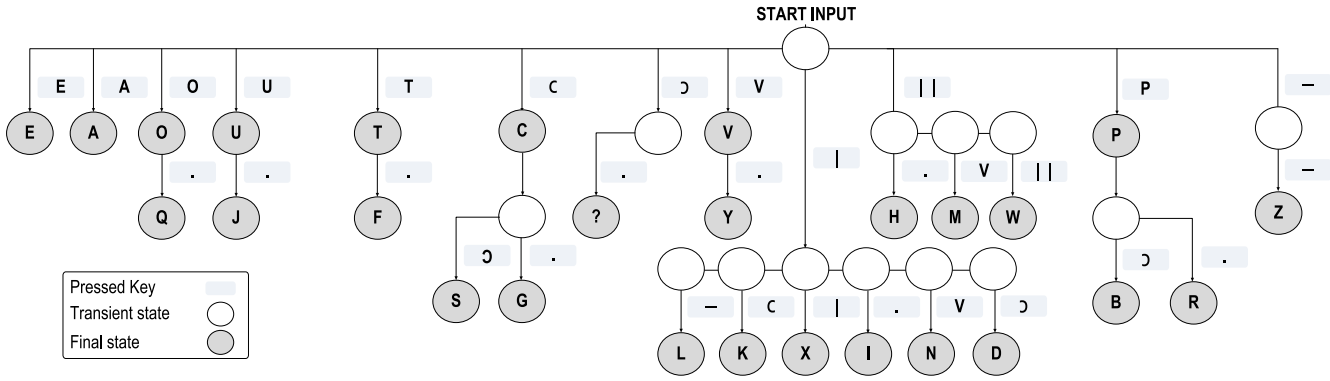
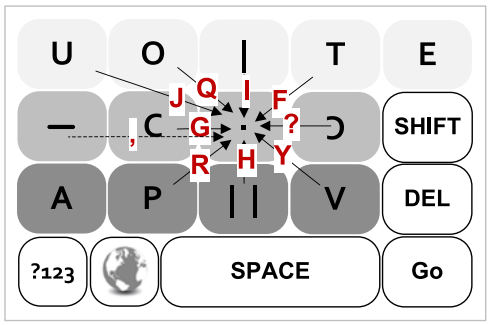
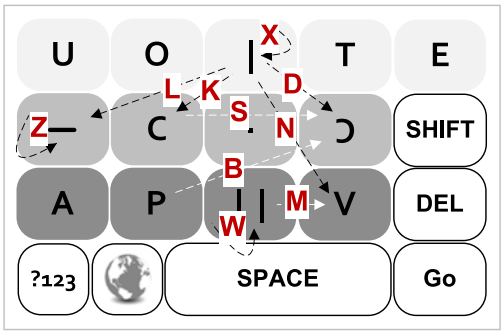


Fig. 4. Deterministic finite automata for the UOIT-entry method. Not shown is a hidden reset transition from each transient state, when the next input did not match the label on the transition.



(a)



(b)

Fig. 5. Design of UOIT keyboard. (a) Finger movements for letters ending with the dot key. (b) Finger movements of letters not ending with the dot key.

ambiguity in the DFA. For example, · can be used instead of − for F, H, and R, and | instead of · for Y letters. These alternative constructions provide more flexible patterns.

IV. Methods

To evaluate the UOIT entry method, we conducted a controlled experiment. The experiment had two rounds in which participants used UOIT, QWERTY, and multitap keyboards to evaluate their performance in terms of error rates and input speeds, using various metrics.

A. Apparatus

A prototype of the UOIT keyboard was implemented using the Java *inputmethodservice* package for Android 4.2.2 [see Fig. 1(b)]. The input device was a Samsung Galaxy Nexus smartphone. For the QWERTY and multitap keyboards, we used the built-in keyboards [see Fig. 1(a) and (c)]. A notepad app measured the time. Every keystroke including errors were logged along with the time for later analysis. Only the standard input sequence from Fig. 4 was allowed during the study. The hint-highlighting aid was used during the experiment. However, aids such as word prediction/correction using a dictionary, automatic completion, and alternative drawing components were not used. During the experiment, the participants responded to a brief questionnaire about their opinions regarding the UOIT-entry method. The questionnaire included 16 statements and the participants were asked to indicate their level of agreement using a 4-point Likert scale (1 = disagree strongly, 2 = somewhat disagree, 3 = somewhat agree, 4 = agree strongly).

Fig. 6. Highlighting hints: C is pressed. As a result, drawing components for G and S are highlighted in red.

B. Participants

Twenty-four unpaid volunteers from our university, including students, staff personnel, and professors, participated in this study (five females, 19 males). They ranged in age from 19 to 53, with an average age of 27 years (SD = 7.64). None of the participants were English natives, and none had experience with the UOIT keyboard before the study.

C. Procedure

The study was conducted individually in an isolated room at our college. Each participant watched a short tutorial video

TABLE I
TOTAL NUMBER OF LETTERS FOR EACH INPUT METHOD

Participants	QWERTY	MULTITAP	UOIT-R1	UOIT-R2
Mean	219.9	224.8	243.4	309.7
SD	29.3	29.5	36.9	39.9
Sum	5278	5394	5842	7433

that described the UOIT keyboard at the beginning of the experiment. In addition, a short explanation of the multitap use was given for each participant at that time. The experiment was introduced based on [20].

During the training session of UOIT, we displayed the construction method in Fig. 3 on a separate computer screen. The user study started after the training session. Each participant was given three sets of paragraphs for entry and each was asked to input the three sets of random paragraphs using the UOIT, QWERTY, and multitap keyboards in a specified order. We did not display Fig. 3 during the first-round test. After the completion of the first-round test, each participant was instructed to fill out a questionnaire (Q1–Q16 in Table II). After a week, the participants were invited back and were instructed to perform the second-round test. In the second-round test for the UOIT-entry method, we did not display Fig. 2. They entered the phrases using the UOIT keyboard only. The participants were asked to fill out another questionnaire (Q17 in Table II) to determine whether they were able to remember the UOIT construction.

D. Tasks

Participants were asked to transcribe randomly selected phrases. The phrase-set used in the study was taken from MacKenzie [20], which contains more than 500 phrases with no punctuation symbols, and just a few instances of uppercase characters. Consequently, users were allowed, but not forced, to delete previously entered text and correct any errors that they noticed. Participants were instructed to complete sentences as quickly and accurately as possible.

During a training session, the participants entered short paragraphs with an average of 53.6 words and 50.2 word for the UOIT and multitap keyboards, respectively. At the first round, each participant was instructed to complete three text-entry tasks using QWERTY, UOIT, and multitap entry methods. For UOIT, participants entered short paragraphs with an average of 48.6 words for each participant, which was equivalent to an average of 243.41 letters. For QWERTY, there was an average of 44 words, which was equivalent to an average of 219.9 letters. For the multitap keyboard, there was an average of 44.8 words, which was equivalent to an average of 224.7 letters. For the second round, participants instructed to type an average of 61.8 word phrases (i.e., the equivalent of 309.7 letters) using UOIT only. Table I summarizes the total number of letters the participants entered for each input method.

E. Dependent Measures

We evaluated the performance of the three keyboards in terms of speed and errors. The speed was assessed using words per minute (WPM). The error rates were evaluated using four metrics: minimum string distance (MSD), key strokes per character (KSPC), total error rate (TER), and utilized/wasted bandwidth (UB/WB).

1) Speed: We used the WPM text-input measure to assess the speed. Entry speed in WPM is calculated by taking the typed characters per minute and dividing it by five characters per word [21].

2) Error: To evaluate the UOIT keyboard, we used the Soukoreff-MacKenzie [22] metrics, which are based on delineating participant's keystrokes into the following four categories.

a) *Correct (C) keystrokes:* Alphanumeric keystrokes that were correct and matched the characters from the presented text.
b) *Incorrect and not fixed (INF) keystrokes:* Errors that were not noticed by the typist and remained in the transcribed text.
c) *Incorrect but fixed (IF) keystrokes:* Erroneous keystrokes in the participant's input stream that were corrected.
d) *Fixes (F):* Keystrokes that were required to perform the corrections (i.e., delete, backspace, cursor movement).

The four derived metrics were determined.

1) The MSD error rate evaluates the number of errors that were committed by the user while entering the presented text and were not corrected (i.e., remained in the transcribed text):

$$\text{MSD} = \frac{INF}{C + INF} \times 100\%. \tag{1}$$

2) KSPC evaluates the cost of fixing the errors that were committed:

$$\text{KSPC} = \frac{C + INF + IF + F}{C + INF}. \tag{2}$$

3) The TER [22] evaluates the total errors that have been committed (i.e., either corrected or remaining in the transcribed text.):

$$\text{TER} = \frac{INF + IF}{C + INF + IF} \times 100\%. \tag{3}$$

If text entry is viewed as information transfer, then C represents the amount of useful information transferred, and INF, IF, and F represent WB. The proportion of bandwidth that represents useful information transfer is defined as:

4) The UB signifies the number of correct characters entered using all the keystrokes:

$$\text{UB} = \frac{C}{C + INF + IF + F}. \tag{4}$$

5) The WB is the proportion of wasted information transferred:

$$\text{WB} = \frac{INF + IF + F}{C + INF + IF + F}. \tag{5}$$

WB signifies the number of keystrokes that were wasted (or not used) for correct characters.

In order to perform a subjective evaluation of participants satisfaction of using the UOIT entry method, we asked participants to answer a satisfaction questionnaire after the experimental evaluation. The questions addressed the UOIT design, letter's construction, keys size, learning efforts, and other aspects (see Table II).

TABLE II
QUESTIONNAIRE RESULTS USING A LIKERT SCALE (1 = DISAGREE STRONGLY, 4 = AGREE STRONGLY)

	Questions	Scale 1%	Scale 2%	Scale 3%	Scale 4%
Q1	Easy to use	0.0	4.2	58.3	29.2
Q2	Letters are intuitive	0.0	0.0	29.2	62.5
Q3	Easy to memorize letter construction	0.0	4.2	33.3	54.2
Q4	Easy to learn	0.0	4.2	37.5	50.0
Q5	Fun to use	0.0	4.2	25.0	62.5
Q6	More familiar, as I practice more	0.0	0.0	16.7	75.0
Q7	Key hint helps during the practice	0.0	4.2	33.3	54.2
Q8	Key are big enough	0.0	0.0	25.0	66.7
Q9	Key size of QWERTY and Multi-tap Keyboards contributes to writing mistakes	0.0	8.3	33.3	50.4
Q10	QWERTY requires aiming and concentration	4.2	33.3	45.8	8.3
Q11	Multi-tap requires aiming and concentration	0.0	16.7	12.5	62.5
Q12	UOIT requires aiming and concentration	0.0	16.7	54.2	25.0
Q13	Prefer using UOIT soft keyboard rather than QWERTY keyboard to decrease typos	0.0	4.2	37.5	50.0
Q14	Like UOIT Design	0.0	4.2	45.8	41.7
Q15	Like UOIT: The way keys are constructed in	0.0	0.0	29.2	62.5
Q16	Like to use UOIT keyboard as default keyboard after few practices	0.0	12.5	20.8	58.3
Q17	Remembering alphabets (**After a week**)	0.0	0.0	37.5	54.2

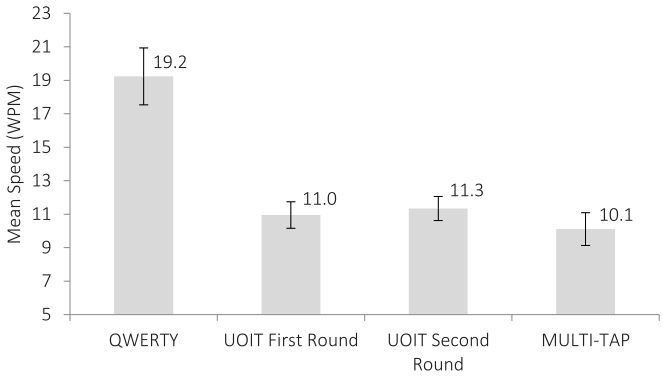


Fig. 7. Entry speed in WPM for QWERTY, UOIT, and multitap keyboards, where the error bars denote 95% CIs.

F. Experimental Design and Data Analysis

The experiment used a within-subject design with three conditions [i.e., UOIT (U), QWERTY (Q), and multitap keyboards (M)]. Each participant was presented with the conditions in a counterbalanced order to avoid any ordering or learning effect. That is, each of six orders (UMQ, UQM, MQU, MUQ, QUM, QMU) was given to four participants. The UOIT-entry method was tested once more after a week (i.e., a second round with a new set of paragraphs) to determine whether the UOIT-entry method was easy to remember and to see if there was any improvement in the entry speed and the error rate. For dependent variables that fit a normal distribution, we used a repeated measures ANOVA and Bonferroni post-hoc test in further analyses. We adopted a nonparametric Friedman test for Questionnaire ordered values that did not fit a normal distribution.

V. RESULTS

Speed: As shown in Fig. 7, the mean entry speed for the 24 participants for the QWERTY-entry method was 19.23 WPM, whereas that for the UOIT-entry method in the first round was 10.95 WPM. The UOIT-entry method mean speed improved in the second round to 11.34 WPM. The WPM speed of the UOIT keyboard was a little more than the half of the QWERTY speed, but it was faster than the speed of the multi-tap keyboard that had a mean speed of 10.11 WPM and maximum speed of 15 WPM. The vertical bar for each point in Fig. 7 denotes a 95% confidence interval. The effect of keyboard type on entry speed scores was significant ($F[2,46] = 92.22$, $p = 0.04$). Post-hoc pairwise comparisons using a Bonferroni correction indicated significant speed difference between QWERTY and UOIT ($p < 0.001$), and there was a trend between UOIT and multitap ($p = 0.07$).

Error: Each of the five metrics (i.e., MSD, KSPC, TER, UB, and WB), shown in Fig. 8(a)–(e), respectively.

The effect of keyboard type on entry MSD scores was significant ($F[2,46] = 2.24$, $p = 0.04$). Post-hoc pairwise comparisons using a Bonferroni correction indicated a difference between QWERTY and UOIT ($p = 0.01$) and between UOIT and multitap ($p = 0.01$).

The effect of keyboard type on entry KSPC scores was significant ($F[2,46] = 18.82$, $p < 0.001$). Post-hoc pairwise comparisons using a Bonferroni correction indicated significant difference between QWERTY and UOIT ($p < 0.001$).

The effect of keyboard type on entry TE scores was significant ($F[2,46] = 19.16$, $p < 0.001$). Post-hoc pairwise comparisons using a Bonferroni correction indicated significant TE difference between QWERTY and UOIT ($p < 0.001$), and between UOIT and multitap ($p < 0.001$).

This results support our argument that the UOIT keyboard reduces errors. Finally, a statistical comparison was conducted between the UOIT results for the training, first round, and second round using a repeated measures ANOVA test for the total error metric; the comparison showed a significant difference ($F[2,46] = 33.10$, $p < 0.001$). This difference resulted mainly from the improvement in the total number of errors after the training session. However, the comparison between the first and second rounds showed also marginally

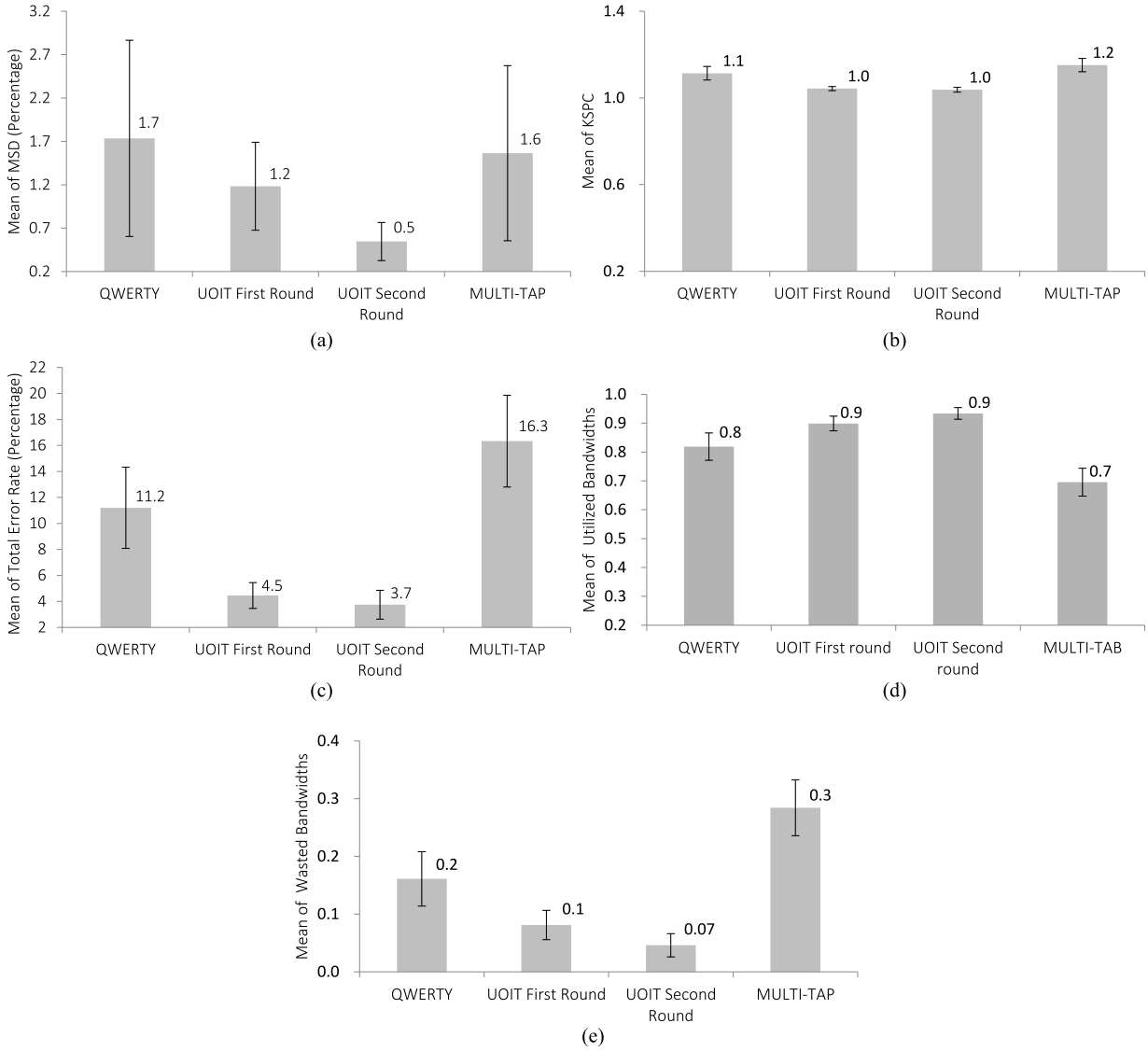


Fig. 8. Evaluation metrics used to measure participants' performance. The error bars denote 95% CIs. (a) MSD evaluation of the Qwerty, UOIT, and multitap keyboards. (b) KSPC evaluation of the Qwerty, UOIT, and multitap keyboards. (c) Total error rate of the Qwerty, UOIT, and multitap keyboards. (d) UB of the Qwerty, UOIT, and multitap text entry. (e) WB of the Qwerty, UOIT, and multitap text entry.

significant difference as illustrated in Fig. 8(c). For the other metrics, we noticed that the results for TER and MSD for the second UOIT round were slightly better than the results from the first round, but the KSPC and UB/WB results remained almost the same for both the rounds.

A. *Questionnaire Results*

As shown in Table II, most of the responses for the questionnaire were higher than 3. The one exception was the statement in Q10, which indicates that the QWERTY keyboard had the lowest aiming and concentration level among the three keyboards used in the study (54.1% of the participants agree). In Q11, most of the participants agreed that the multitap entry method has the highest aiming and concentration burden (79.2% of the participants agree). To confirm if there is a significant difference in participants responses to Q10, Q11, and Q12, a nonparametric Friedman's test was conducted and showed a statistical significance of the participants responses, $X^2(2) = 8.4$, $p = 0.01$. On the other hand, the statement in Q16, which was a statement about using the UOIT Soft Keyboard as the default soft keyboard after a few practice sessions, received a higher level of acceptance. Our questionnaire results indicate that most of the users agreed that the UOIT-entry method was easy to use, intuitive for learning and memorization, and better for reducing typing errors. These survey results support our error evaluation results.

VI. Conclusion

In this paper, we have introduced a novel text-entry method referred to as the UOIT-entry method, which can be used as a keyboard for mobile devices with a small screen. The UOIT design was based on the constructive drawing of letters using

either single or double keystrokes. Our study showed that users were able to type more accurately with the UOIT keyboard than with the QWERTY and multitap keyboards. The entry speed for the UOIT keyboard was shown to be faster than the entry speed for the multitap keyboard. However, it was slower than with the entry speed for the QWERTY keyboard. Our study showed that the participants were positive about using the UOIT keyboard, despite their familiarity with the conventional QWERTY and multitap keyboards.

Our results are consistent with other studies which investigated the relation between the keys size and error rate. For instance, Mizobuchi [23] and Parhi *et al.* [24] found that the error rate improved with a larger key size, and although Brewster [25] was specifically interested in the interaction between target size and audio feedback on performance, he also found a significant improvement in throughput when targets size increased. These works are consistent with Fitts' law, which predicts that the time for a person to make a rapid aimed movement to a target increases with a distance and decreases with the target size [26], [27]. Thus, Fitts' law predicted that the smaller targets would be more difficult, and thus slower to hit [24].

UOIT takes two advantages from Qwerty and multitap. That is, UOIT has large keys like multitap, while it has only one symbol in one key like Qwerty (less crowded). We note that Qwerty does not have such a large key like UOIT and multitap, and multitap does not have a design that has one symbol in one key like UOIT and Qwerty. The comparison result with multitap having similar key size shows that participants with multitap were slower than with UOIT, which says that participants were not familiar with the typing with the multitap keyboard neither. Now, if we compare these two keyboards in terms of error rates, UOIT is better than multitap. This indicates that the less error rates of the UOIT keyboard are not because of the unfamiliarity but due to the UOIT design. The main difference between UOIT and multitap is that UOIT has only one symbol in a key, whereas multitap has two symbols in a key. In summary, UOIT small error rates comes from 1) big keys and 2) one symbol in one key (less crowded).

REFERENCES

[1] M. Kolsch and M. Turk, "Keyboards without keyboards: A survey of virtual keyboards," in *Proc. Sensing Input Media-Centric Syst.*, 2002.

[2] E. Rabin and A. M. Gordon, "Tactile feedback contributes to consistency of finger movements during typing," *Exp. Brain Res.*, vol. 155, no. 3, pp. 362–369, 2004.

[3] E. Hoggan, S. A. Brewster, and J. Johnston, "Investigating the effectiveness of tactile feedback for mobile touchscreens," in *Proc. 26th Ann. SIGCHI Conf. Human Factors Comput. Syst.*, 2008.

[4] A. Gunawardana, T. Paek, and C. Meek, "Usability guided key-target resizing for soft keyboards," in *Proc. 15th Int. Conf. Intell. user Interfaces*, 2010, pp. 111–118.

[5] D. Vogel and P. Baudisch, "Shift: A technique for operating pen-based interfaces using touch," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2007, pp. 657–666.

[6] I. S. MacKenzie and S. X. Zhang, "The design and evaluation of a high-performance soft keyboard," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 1999, pp. 25–31.

[7] K. Al Faraj, M. Mojahid, and N. Vigouroux, "Bigkey: A virtual keyboard for mobile devices," in *Proc. 13th Int. Conf. Human-Comput. Interaction. Part III: Ubiquitous Intell. Interaction*, 2009, pp. 3–10.

[8] G. Aulagner, R. François, B. Martin, D. Michel, and M. Raynal, "Floodkey: Increasing software keyboard keys by reducing needless ones without occultation," in *Proc. 10th WSEAS Int. Conf. Appl. Comput. Sci.*, 2010, pp. 412–417.

[9] J. Himberg, J. Häkkilä, P. Kangas, and J. Mäntyjärvi, "On-line personalization of a touch screen based keyboard," in *Proc. 8th Int. Conf. Intell. User Interfaces*, 2003, pp. 77–84.

[10] L. Magnien, J. Bouraoui, and N. Vigouroux, "Mobile text input with soft keyboards: Optimization by means of visual clues," in *Proc. Mobile Human-Comput Interaction*, 2004, vol. 3160, pp. 337–341.

[11] H. Tinwala and I. S. MacKenzie, "Eyes-free text entry with error correction on touchscreen mobile devices," in *Proc. 6th Nordic Conf. Human-Comput. Interaction: Extending Boundaries*, 2010, pp. 511–520.

[12] A. Bharath and S. Madhvanath, "Freepad: A novel handwriting-based text input for pen and touch interfaces," in *Proc. 13th Int. Conf. Intell. User Interfaces*, 2008, pp. 297–300.

[13] T. Költringer and T. Grechenig, "Comparing the immediate usability of graffiti 2 and virtual keyboard," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2004, pp. 1175–1178.

[14] E. Matias, I. S. MacKenzie, and W. Buxton, "Half-qwerty: A one-handed keyboard facilitating skill transfer from qwerty," in *Proc. Confe. Human Factors Comput. Syst.*, 1993, pp. 88–94.

[15] J. Goodman and G. Venolia, "Language modeling for soft keyboards," in *Proc. 89th National Conf. Artif. Intell.*, 2002, pp. 419–424.

[16] D. Rudchenko, T. Paek, and E. Badger, "Text text revolution: a game that improves text entry on mobile touchscreen keyboards," in *Proc. 9th Int. Conf. Pervasive Comput.*, 2011, pp. 206–213.

[17] S. Kwon, D. Lee, and M. K. Chung, "Effect of key size and activation area on the performance of a regional error correction method in a touch-screen {QWERTY} keyboard," *Int. J. Ind. Ergonomics*, vol. 39, no. 5, pp. 888–893, 2009.

[18] H. Beker and F. Piper, *Cipher Systems: The Protection of Communications* (ser. New electronics 'communications international' book). Chambersburg, PA, USA: Northwood Books, 1982.

[19] C. L. James and K. M. Reischel, "Text input for mobile devices: Comparing model prediction to actual performance," in *Proc. . SIGCHI Conf. Human Factors Comput. Syst.*, 2001, pp. 365–371.

[20] I. S. MacKenzie and R. W. Soukoreff, "Phrase sets for evaluating text entry techniques," in *Proc. SIGCHI Conf. Human Factors Comput. Syst.*, 2003, pp. 754–755.

[21] H. Yamada. (1980). *A Historical Study of Typewriters and Typing Methods, From the Position of Planning Japanese Parallels*. Journal of Information Processing. [Online]. Available: http://books.google.co.kr/books?id=KtWuGwAACAAJ

[22] R. W. Soukoreff and I. S. MacKenzie, "Metrics for text entry research: An evaluation of MSD and KSPC, and a new unified error metric," in *Proc. SIGCHI Conf. Human Factors. Comput. Syst.*, 2003, pp. 113–120.

[23] S. Mizobuchi, K. Mori, X. Ren, and Y. Michiaki, "An empirical study of the minimum required size and the minimum number of targets for pen input on the small display," in *Proc4th Int. Symp. Mobile Human-Comput. Interaction*, 2002, pp. 184–194.

[24] P. Parhi, A. K. Karlson, and B. B. Bederson, "Target size study for one-handed thumb use on small touchscreen devices," in *Proc8th Conf. Human-Comput. Interaction Mobile Devices Services*, 2006, pp. 203–210.

[25] S. Brewster, "Overcoming the lack of screen space on mobile computers," *Personal Ubiquitous Comput.*, vol. 6, no. 3, pp. 188–205, Jan. 2002.

[26] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement." *J. Exp. Psychol.*, vol. 47, no. 6, pp. 381–391, 1954.

[27] I. S. MacKenzie, "Fitts' law as a research and design tool in human-computer interaction," *Human-Comput. Interact.*, vol. 7, no. 1, pp. 91–139, 1992.