Flickey: Flick-Based QWERTY Software Keyboard for Ultra-small Touch Screen Devices

Akira Ishii(✉), Hiroyuki Hakoda, and Buntarou Shizuki

University of Tsukuba, Tsukuba, Japan
{ishii,shizuki}@iplab.cs.tsukuba.ac.jp,
hiroyukihakoda@acm.org

Abstract. Ultra-small touch screen devices (e.g., smartwatches) are required to be small and lightweight so that they can be worn on the body with no frustration. For this reason, users often have difficulties in selecting the correct keys, and thus entering texts. Therefore, entering texts on ultra-small touch screen devices is impractical. To address this problem, we present Flickey, a flick-based QWERTY software keyboard for ultra-small touch screen devices. The flick-based selection mechanism of Flickey, in combination with its callout technique, allows users to select a tiny key on the small keyboard more easily than with a tap. To investigate the performance and usability of Flickey, we developed a prototype of Flickey and conducted a comparative experiment with two existing keyboards. The results suggest that Flickey shows high performance when the size of the keyboard becomes small.

Keywords: Smartwatch · Wearable device · Text entry
Small target acquisition · Pointing · Fat finger · Occlusion
Interaction technique

1 Introduction

Ultra-small touch screen devices (henceforth referred to as ultra-small devices) such as smartwatches, are required to be small and lightweight so that they can be worn on the body with no frustration. For this reason, the input area of such devices is limited, and thus, ultra-small devices are more prone to occlusion or the *fat finger problem* [16] than smartphones or tablets. As a result, users often have difficulties in selecting correct keys and thus entering texts. Therefore, entering texts on ultra-small devices is impractical. To address this problem, we present Flickey (Fig. 1), a flick-based QWERTY software keyboard for ultra-small devices. The flick-based selection mechanism of Flickey, in combination with its callout technique, allows users to select a tiny key on the small keyboard easier than with a tap.

In this work, we developed a prototype of Flickey and conducted a comparative experiment with two existing keyboards under three size conditions to

M. Kurosu (Ed.): HCI 2018, LNCS 10903, pp. 31–42, 2018.
https://doi.org/10.1007/978-3-319-91250-9_3

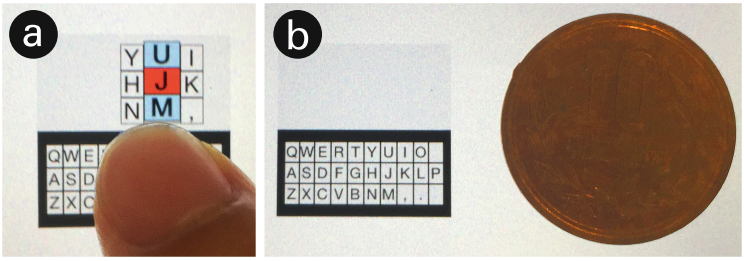


Fig. 1. Flickey. (a) User is entering text using Flickey with his/her index finger. (b) Flickey compared with a 10 JPY coin (diameter: 20 mm).

investigate the performance and usability of Flickey. The results suggest that Flickey shows high performance when the size of the keyboard becomes small.

2 Related Work

Numerous text entry methods for ultra-small devices have been explored. Among them, many researchers have adopted a QWERTY layout on ultra-small devices in designing their input methods; this is because many users are familiar with the QWERTY layout, and thus the learning cost of input methods can be lowered. For example, in ZoomBoard [14], users use touch gestures which trigger iterative zooming (i.e., visual magnification) until a certain level of zoom is reached, where users can press a key easily with their finger. In Swipeboard [3], users use a swipe gesture to select a key, which eliminates ambiguous selection. WatchWriter [6] uses gesture typing to enable users to enter a word per gesture. SplitBoard [8] splits a QWERTY keyboard into two parts to increase the size of each key. DriftBoard [15] is a panning-based input technique using a movable QWERTY keyboard and a fixed cursor point. ZShift [11] uses a callout to display a copy of the occluded area to eliminate the occlusion caused by a finger, thus enabling text entry using a QWERTY layout on ultra-small devices.

By contrast, some keyboards do not use a QWERTY layout. Komnios and Dunlop [9] used a specialized keyboard with six large keys and adopted alternative/next word predictions based on a dictionary to realize text entry on ultra-small devices. DragKeys [4] uses a specialized keyboard that consists of eight circularly arranged large keys, each of which contains multiple small keys. To enter a character, users select a large key with a dragging gesture and then select a small key within the selected large key using another dragging gesture.

In this work, we use a QWERTY layout to reduce the learning cost and adopt the same approach of ZShift, which uses a callout to avoid the occlusion problem. In addition, we use a flick as a key selection trigger to eliminate ambiguous selection.

3 Flickey

Flickey is a flick-based QWERTY software keyboard, which uses a callout technique (similar to ZShift [11]) and adopts a flick as a key selection trigger. This flick-based selection mechanism in combination with the callout technique allows users to select a tiny key on the small keyboard easier than with a tap.

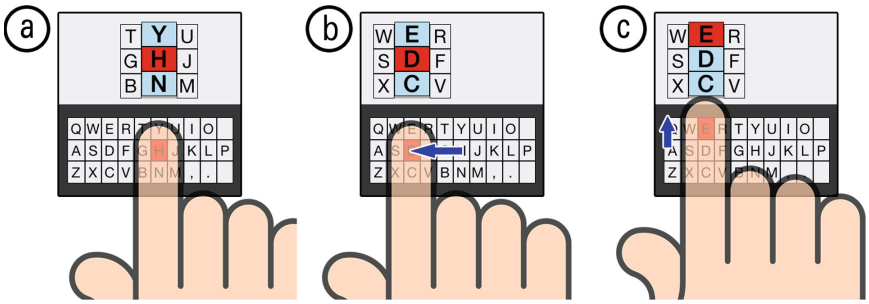


Fig. 2. Text entry procedure of Flickey. To enter a character (a) the users select the column of the key with a touch-down. (b) If the selected column is not the intended one, the users can change the selected column by moving their finger to the right or left. (c) Users select the key by a touch-up or a flick to enter the character.

Figure 2 illustrates the text entry procedure of Flickey. Flickey involves two steps for entering a character. First, the users select the column of the key by a touch-down (Fig. 2a). If the selected column is not the intended one at the first touch-down, users can change the selected column by moving their finger to the right or left (Fig. 2b). Second, the users select a key by a touch-up or a flick (Fig. 2c): the middle-row is selected by a touch-up and the upper-row or lower-row is selected by a flick-up or flick-down, respectively.

This design allows users to select a tiny key on the small keyboard easier than with a tap. In the first step, users concentrate on selecting the horizontal position of the key, which eliminates concern over the vertical position of the finger; in the second step, users simply have to select the row of the key with a touch-up, flick-up, or flick-down, which eliminates the need for precise vertical positioning. In addition, Flickey displays the current key selection in a callout which is placed above the keyboard to remedy the fat finger problem.

4 Evaluation

We conducted an experiment to evaluate the performance and usability of Flickey, comparing it with two existing keyboards: ZoomBoard [14] and ZShift [11]. Experiment participants performed text entry tasks using three different keyboards under three size conditions (Fig. 4), similar to the approach of

Fig. 3. Smartphone attached in a landscape orientation with respect to the non-dominant hand.

previous studies [11,14], to investigate their performance on a wide variety of ultra-small devices[1] (e.g., bracelet-style devices and smartwatches).

4.1 Participants

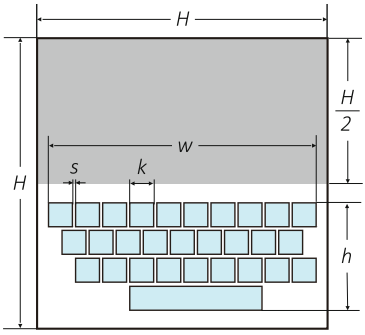
We recruited five participants (four males and one female) aged between 21 and 22 years. All the participants majored in computer science, were right-handed, and were familiar with the QWERTY layout. Each participant received 1,640 JPY (approximately 15 USD) after completion of the experiment.

4.2 Apparatus

We implemented the three keyboards (i.e., ZoomBoard [14], ZShift [11], and ours) on an iPhone 5 smartphone (iOS 8.3, 4 in., $1{,}136 \times 640$ pixels, 326 ppi). Similar to the approaches in [11,14], we used a smartphone for the experiment because its touch screen is more accurate than the touch screen of existing smartwatches. The smartphone was attached in a landscape orientation to the non-dominant hand of the participant using a Velcro strap (D&M Co., Ltd.; knee wrap; 842XUD2786 BLK M) as shown in Fig. 3.

We implemented the three keyboards on the smartphone. Figure 4 shows the layout of the keyboards used in the experiment. The *small* size of the keyboards was determined first, and then the *medium* (small $\times$ 1.33) and *large* (small $\times$ 1.77) sizes were determined in relation to the small size, which is the same approach as in a previous study [11]. All the keyboards are smaller than the smartphone's QWERTY keyboard (the dimension of the small keyboard is approximately 1/20 (0.054x) of the dimension of the QWERTY keyboard on the iPhone 6).

[1] Examples of the screen sizes of the ultra-small devices (width $\times$ height).
- Samsung Gear Fit: 13 mm $\times$ 45 mm
- Apple Watch (38 mm): 21 mm $\times$ 26 mm
- Apple Watch (42 mm): 24 mm $\times$ 30 mm.



Size [mm]	H	w	h	s	k
small	18.0	16.5	6.5	0.2	1.5
medium	24.0	22.0	8.7	0.2	2.0
large	32.0	29.3	11.6	0.3	2.7

Fig. 4. Layout of the keyboards used in the experiment.

4.3 Procedure and Task

The experiment was conducted in a calm office environment. First, the purpose of the experiment was explained to the participants. In addition, they were informed that they could abort the experiment and take a break at any time. The participants were required to sign a consent form and answer a demographic questionnaire. Then, we measured the width of the index finger of each participant's dominant hand using a digital caliper; for the measurement, the digital caliper was aligned with the distal interphalangeal joint (Fig. 5). The average width obtained was 14.3 mm (Standard Deviation = 0.8 mm), which matches the standard size for Japanese people [10].

We explained the three keyboards to the participants through a short demonstration. Then, the participants practiced each keyboard. As practice, the participants entered the short phrase "tsukuba taro" using each keyboard.

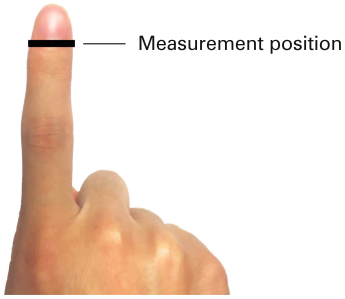


Fig. 5. Measurement position for the index finger.

Table 1. Text entry speed (WPM). Standard deviations are shown in parentheses.

Size	Keyboard			ANOVA	
	ZoomBoard	ZShift	Flickey	$F_{2,12}$	p
Small	7.5 (0.3)	8.5 (0.7)	8.7 (0.3)	8.2	.006
Medium	8.7 (0.5)	10.1 (0.4)	9.4 (0.5)	13.1	.001
Large	8.9 (0.2)	12.4 (0.5)	8.8 (0.9)	58.1	.000

Then, to practice the delete gesture (left swipe on the keyboard), which deletes one character, the participants deleted the inputted short phrase. Then, the participants practiced the space gesture (right swipe on the keyboard) for entering a space between words. For the final stage of practice, the participants entered another short phrase, "taro". The participants practiced the three keyboards in the same order they evaluated them. The duration of the practice was approximately 6 min.

After the practice, the participants entered five phrases (five trials) for one keyboard and one size in a session. The participants were instructed to enter the phrases as quickly and accurately as possible. The participants were also instructed to correct mistakes when they entered a wrong phrase. One session was performed for each condition (i.e., nine sessions were performed in total). Therefore, the participants performed 45 trials (=3 *keyboards* × 3 *sizes* × 5 *phrases*). The conditions were presented to the participants in a random order without redundancy to counterbalance possible biases caused by the order of the conditions. The phrases also were presented to the participants in a random order. The phrases were chosen from the phrase set provided by MacKenzie et al. [12], which contains 500 phrases in English. After each session was completed, the participants were asked to report their impressions regarding the selection of the targets to the experimenter and to respond to the System Usability Scale (SUS) questionnaires [1,2]. In this experiment, we used the Japanese version of SUS [5] because all the participants were Japanese. The participants were also asked to respond to the NASA Task Load Index (NASA-TLX) questionnaires [7]. We used the Japanese version of NASA-TLX [13] for the same reason as above. Then, the participants were asked to take a break of 3 min to reduce the effects of fatigue.

After all sessions were complete, the participants were given a questionnaire to report their impressions of each condition. The duration of the experiment was approximately 120 min.

4.4 Result and Analysis

Text Entry Speed. We used Words Per Minute (WPM) as an index of text entry speed and analyzed the results using a one-way repeated measure analysis of variance (ANOVA). Table 1 and Fig. 6 show the text entry speed and the result of the ANOVA. The result indicated a significant main effect of *keyboard* under

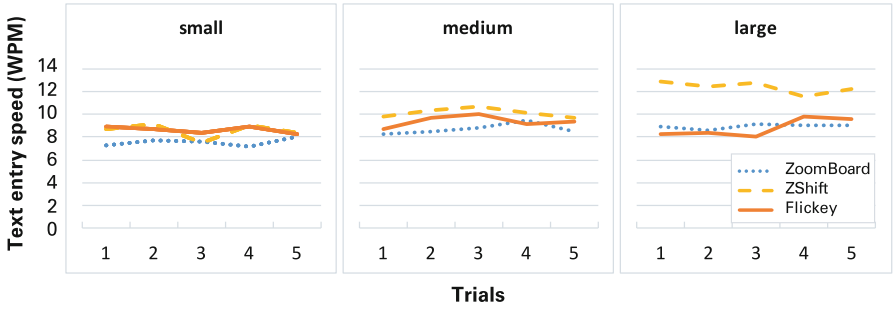


Fig. 6. Text entry speed (WPM).

Table 2. Character error rate (CER).

Size	Keyboard			ANOVA	
	ZoomBoard	ZShift	Flickey	$F_{2,12}$	p
Small	0.0% (0.0)	0.6% (0.3)	0.0% (0.0)	15.8	.000
Medium	0.6% (1.4)	0.4% (0.6)	0.0% (0.0)	0.6	.543
Large	0.0% (0.0)	0.3% (0.6)	0.3% (0.4)	0.7	.519

all size conditions. We used Tukey's test with a significance level of 0.05 for post hoc analysis. The result revealed the following: (1) under the small condition, significant differences were found between ZoomBoard and ZShift ($p < 0.05$) and between ZoomBoard and Flickey ($p < 0.01$), (2) under the medium condition, a significant difference was found between ZoomBoard and ZShift ($p < 0.001$), and (3) under the large condition, significant differences were found between ZoomBoard and ZShift ($p < 0.001$) and between ZShift and Flickey ($p < 0.001$).

In the questionnaire, one participant stated that "*I can enter text using Flickey accurately even if the size of the keyboard is small.*" The results support this comment. The text entry speed under the small condition was equivalent to that of the large condition. In contrast, ZoomBoard and ZShift achieved faster text entry speed as the size of the keyboard became larger, whereas Flickey achieved similar scores for all sizes. One participant stated that "*I could not operate the small size Flickey and the large size Flickey in the same way*" and two participants stated that "*I have to move my finger much farther than I thought to enter a key under the large condition.*" This is because we changed the threshold to detect a flick according to the size of the keyboard, which changed the usability. Thus, the text entry speed could be improved when the thresholds are optimized.

Error Rate. Table 2 shows the error rate. We used character error rate (CER) as the index of the error rate [17] in this analysis, which is calculated as the Damerau-Levenshtein distance between the submitted text and the reference text. It is

Table 3. Corrected error rate (Cerr). SDs are shown in parentheses.

Size	Keyboard			ANOVA	
	ZoomBoard	ZShift	Flickey	$F_{2,12}$	p
Small	4.6% (3.3)	5.5% (1.9)	9.5% (5.2)	2.5	.128
Medium	4.7% (3.7)	9.0% (3.7)	10.0% (3.7)	2.9	.093
Large	2.0% (1.9)	6.5% (5.0)	9.5% (3.4)	5.3	.022

Table 4. Usability (SUS). SDs are shown in parentheses.

Size	Keyboard			ANOVA	
	ZoomBoard	ZShift	Flickey	$F_{2,12}$	p
Small	66.0 (13.3)	60.0 (15.2)	59.0 (12.9)	0.37	.696
Medium	72.0 (11.4)	69.5 (14.1)	62.0 (10.8)	0.91	.427
Large	69.5 (17.0)	81.5 (11.8)	54.0 (9.8)	5.44	.021

normalized by the number of characters in the reference text. This index indicates whether the participants corrected their wrong input or not. In the results, we found that the CER under all conditions was so low that we concluded that the participants corrected their wrong input as they were asked to. We also analyzed the results using a one-way repeated measure ANOVA. The result indicated a significant main effect of *keyboard* under the small condition. We used Tukey's test with a significance level of 0.05 for post hoc analysis. The result revealed that significant differences were found under the small condition between ZoomBoard and ZShift ($p < 0.01$) and between ZShift and Flickey ($p < 0.01$).

We also used corrected error rate (Cerr) as another index of the error rate [18], which is the percentage of the wrong inputs in all inputs and analyzed the results using a one-way repeated measure ANOVA. Table 3 shows the Cerrs and the result of the ANOVA. The result indicated a significant main effect of *keyboard* under the large condition. We used Tukey's test with a significance level of 0.05 for post hoc analysis. The result revealed that a significant difference was found under the large condition between ZoomBoard and Flickey ($p < 0.05$).

We analyzed the cause of the errors under the Flickey condition. We found that a wrong key row which was shifted by one from the correct one was selected. In other words, the participants selected a wrong key row because they accidentally moved their finger too much when they selected a key row. The reason for this might be that the content of the callout is changed discretely in response to the current selection of a key row. Under the ZShift condition, the participants can recognize their finger position accurately when selecting a key because the content of the callout is changed continuously in response to the movement of their finger. In contrast, under the Flickey condition, the participants cannot recognize their actual finger position because the content of the callout is changed discretely.

Table 5. Workload (NASA-TLX). SDs are shown in parentheses.

Size	Keyboard			ANOVA	
	ZoomBoard	ZShift	Flickey	$F_{2,12}$	p
Small	36.4 (24.9)	32.5 (19.7)	37.3 (21.3)	0.07	.937
Medium	40.2 (22.1)	34.7 (25.2)	41.9 (19.8)	0.14	.872
Large	33.7 (13.7)	27.6 (24.1)	49.0 (21.9)	1.46	.271

Usability and Workload. We analyzed the results of SUS using a one-way repeated measure ANOVA. Table 4 shows the scores of SUS and the result of the ANOVA. The result indicated a significant main effect of *keyboard* under the large condition. We used Tukey's test with a significance level of 0.05 for post hoc analysis. The result revealed that a significant difference was found under the large condition between ZShift and Flickey ($p < 0.05$).

Fig. 7. Workload (NASA-TLX).

We analyzed the results of NASA-TLX using a one-way repeated measure ANOVA. Table 5 and Fig. 7 show the scores of NASA-TLX and the result of the ANOVA. The result indicated no significant difference between any conditions.

The above results suggest the following:

– ZShift achieved higher SUS scores as the size of keyboard became larger. This means that the participants could use ZShift as a standard QWERTY

keyboard as the size of the keyboard became larger. Comments in the questionnaire support this observation. Two participants stated that "*I could use ZShift as a standard QWERTY keyboard when the size of the keyboard was large.*"
- ZoomBoard achieved similar scores for both SUS and NASA-TLX under all size conditions. This means that the usability does not change even if the size of the keyboard changes. Comments in the questionnaire support this observation. Two participants stated that "*The input procedure of ZoomBoard was a little bit troublesome because I always had to tap twice to enter a key.*" One participant stated that "*I had to look intensively at the keyboard at all times because the layout of the keyboard changed for every zoom.*"
- Flickey achieved higher NASA-TLX scores under the large condition than under the small condition. This means that the input procedure of using a flick is too complicated even though the participants could select a key directly without a flick under the large condition. This result suggests that Flickey is useful when the size of the keyboard is particularly small as under the small condition.

4.5 Discussion

As a result, with the keyboard size of small (16.5 mm), Flickey achieved a good performance (ZoomBoard: 7.5 WPM, ZShift: 8.5 WPM, Flickey: 8.7 WPM). However, the Cerr of Flickey tended to be higher than the other keyboards. In contrast, the participants could enter text correctly (i.e., the CER was quite low) because they corrected their errors. Moreover, the SUS and NASA-TLX scores of Flickey were similar to other keyboards under the small condition. These results suggest that Flickey can be used practically on ultra-small devices even if the Cerr is high.

Furthermore, the text entry speed of Flickey was fast, in contrast to the high Cerr. This suggests that Flickey has the potential for improving text entry speed if the Cerr is decreased. Therefore, we will explore interaction designs to decrease the Cerr of Flickey.

5 Future Work

We found many potential improvements from the results of our experiment. In our current implementation, the threshold to detect flick changes according to the size of the keyboard, which leads to a change in usability. Therefore, we will conduct further experiments to explore the most suitable thresholds to detect a flick under each size condition. Furthermore, the Cerr of Flickey tended to be higher than other keyboards because the content of the callout is changed discretely in response to the current selection of a key row. To address this problem, we will make improvements in which the content of a callout changes continuously in response to the users' drag operation. After these improvements, we will conduct further experiments to evaluate the performance of Flickey.

In our experiment, we used a smartphone. This design may influence the results, especially those related to the performance around the edge of the screen. Therefore, in the immediate future, we will conduct the above experiment using a real smartwatch.

6 Conclusion

We presented a flick-based QWERTY software keyboard called Flickey for ultra-small devices. Flickey enables users to enter text on ultra-small devices because the flick-based selection mechanism of Flickey, in combination with its callout technique, eliminates ambiguous selection, which allows users to select a tiny key on a small keyboard. To investigate the text entry performance and usability of Flickey, we developed a prototype of Flickey and conducted a comparative experiment with two existing keyboards. As a result, with a keyboard size of 16.5 mm, Flickey achieved a good performance (ZoomBoard: 7.5 WPM, ZShift: 8.5 WPM, Flickey: 8.7 WPM). The results suggest that Flickey shows high performance when the size of the keyboard becomes small, and thus Flickey could be used practically on ultra-small devices.

References

1. Brooke, J.: SUS: A quick and dirty usability scale. In: Usability Evaluation in Industry (1996)
2. Brooke, J.: SUS: A retrospective. J. Usability Stud. **8**(2), 29–40 (2013). http://dl.acm.org/citation.cfm?id=2817912.2817913
3. Chen, X.A., Grossman, T., Fitzmaurice, G.: Swipeboard: a text entry technique for ultra-small interfaces that supports novice to expert transitions. In: Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology, UIST 2014, pp. 615–620. ACM, New York (2014). http://doi.acm.org/10.1145/2642918.2647354
4. Cho, H., Kim, M., Seo, K.: A text entry technique for wrist-worn watches with tiny touchscreens. In: Proceedings of the Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology, UIST 2014 Adjunct, pp. 79–80. ACM, New York (2014). http://doi.acm.org.ezproxy.tulips.tsukuba.ac.jp/10.1145/2658779.2658785
5. Furui, Y., Maeda, T., Matsumoto, S.: A system for capturing the screenshots of educational materials and its experimental evaluation. In: Information Symposium of Hinokuni. Information Processing Society of Japan (2014). (in Japanese)
6. Gordon, M., Ouyang, T., Zhai, S.: WatchWriter: tap and gesture typing on a smartwatch miniature keyboard with statistical decoding. In: Proceedings of the 34th Annual ACM Conference on Human Factors in Computing Systems, CHI 2016, pp. 3817–3821. ACM, New York (2016). http://doi.acm.org/10.1145/2858036.2858242
7. Hart, S.G., Staveland, L.E.: Development of NASA-TLX (Task Load Index): results of empirical and theoretical research. Hum. Ment. Workload **1**(3), 139–183 (1988)
8. Hong, J., Heo, S., Isokoski, P., Lee, G.: SplitBoard: a simple split soft keyboard for wristwatch-sized touch screens. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, pp. 1233–1236. ACM, New York (2015). http://doi.acm.org/10.1145/2702123.2702273

9. Komninos, A., Dunlop, M.: Text input on a smart watch. IEEE Pervasive Comput. **13**(4), 50–58 (2014)
10. Kouchi, M.: Aist the measurement data of the hands of the Japanese (2012). https://www.dh.aist.go.jp/database/hand/index.html. Accessed 10 Feb 2018 (in Japanese)
11. Leiva, L.A., Sahami, A., Catala, A., Henze, N., Schmidt, A.: Text entry on tiny QWERTY soft keyboards. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, pp. 669–678. ACM, New York (2015). http://doi.acm.org/10.1145/2702123.2702388
12. MacKenzie, I.S., Soukoreff, R.W.: Phrase sets for evaluating text entry techniques. In: CHI 2003 Extended Abstracts on Human Factors in Computing Systems, CHI EA 2003, pp. 754–755. ACM, New York (2003). http://doi.acm.org/10.1145/765891.765971
13. Miyake, S., Kumashiro, M.: Subjective mental workload assessment technique - an introduction to NASA-TLX and SWAT and a proposal of simple scoring methods. Hum. Factors Ergon. **29**(6), 399–408 (1993). (in Japanese)
14. Oney, S., Harrison, C., Ogan, A., Wiese, J.: ZoomBoard: a diminutive QWERTY soft keyboard using iterative zooming for ultra-small devices. In: Proceedings of the 31st Annual ACM Conference on Human Factors in Computing Systems, CHI 2013, pp. 2799–2802. ACM, New York (2013). http://doi.acm.org/10.1145/2470654.2481387
15. Shibata, T., Afergan, D., Kong, D., Yuksel, B.F., MacKenzie, I.S., Jacob, R.J.: DriftBoard: a panning-based text entry technique for ultra-small touchscreens. In: Proceedings of the 29th Annual Symposium on User Interface Software and Technology, UIST 2016, pp. 575–582. ACM, New York (2016). http://doi.acm.org/10.1145/2984511.2984591
16. Siek, K.A., Rogers, Y., Connelly, K.H.: Fat finger worries: how older and younger users physically interact with PDAs. In: Costabile, M.F., Paternò, F. (eds.) INTERACT 2005. LNCS, vol. 3585, pp. 267–280. Springer, Heidelberg (2005). https://doi.org/10.1007/11555261_24
17. Soukoreff, R.W., MacKenzie, I.S.: Measuring errors in text entry tasks: an application of the Levenshtein string distance statistic. In: CHI 2001 Extended Abstracts on Human Factors in Computing Systems, CHI EA 2001, pp. 319–320. ACM, New York (2001). http://doi.acm.org/10.1145/634067.634256
18. Soukoreff, R.W., MacKenzie, I.S.: Metrics for text entry research: an evaluation of MSD and KSPC, and a new unified error metric. In: Proceedings of the 21st Annual ACM Conference on Human Factors in Computing Systems, CHI 2003, pp. 113–120. ACM, New York (2003). http://doi.acm.org/10.1145/642611.642632