This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Kim, Sunjun; Lee, Geehyuk

Design and Evaluation of Semi-Transparent Keyboards on a Touchscreen Tablet

Published in:

Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces

DOI:

[10.1145/3279778.3279801](https://doi.org/10.1145/3279778.3279801)

Published: 25/11/2018

Document Version

Peer reviewed version

Please cite the original version:

Kim, S., & Lee, G. (2018). Design and Evaluation of Semi-Transparent Keyboards on a Touchscreen Tablet. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces* (pp. 155-166). ACM. <https://doi.org/10.1145/3279778.3279801>

Design and Evaluation of Semi-Transparent Keyboards on a Touchscreen Tablet

Sunjun Kim

Department of ComNet, Aalto Univ.
Espoo, Finland
sunjun.kim@aalto.fi

Geehyuk Lee

School of Computing, KAIST
Daejeon, Republic of Korea
geehyuk@gmail.com

ABSTRACT

As tablet computers are hosting more productivity applications, efficient text entry is becoming more important. A soft keyboard, which is the primary text entry interface for tablets, however, often competes with applications for the limited screen space. A promising solution to this problem may be a semi-transparent soft keyboard (STK), which can share the screen with an application. A few commercial STK products are already available, but research questions about the STK design have not been explored in depth yet. Therefore, we conducted three experiments to answer 1) the effect of the transparency level on usability, 2) exploration of diverse design options for an STK, and 3) the effect of an STK on the different text caret positions. The results imply that STKs with 50% opacity showed a balanced performance; well-designed STKs were acceptable in both content reading and typing situations; which could reach 90-100% of an opaque keyboard in terms of overall performance; and the text caret could intrude the STK down to the number row.

Author Keywords

Semi-transparent; soft keyboard; virtual keyboard; text entry; transparency; translucent; touchscreen; tablet; mobile.

INTRODUCTION

Touchscreen-based mobile devices are becoming a platform for productivity applications. For example, Microsoft Office or Apple iWorks both provide mobile applications designed for touchscreen devices. Tablet computers, in particular, are more suitable for such applications because they provide the large workspace and full-size soft keyboard. A commonly noted problem, however, is that a keyboard claims too much workspace and leaves little for an application. For efficient typing, the key size of a soft keyboard should be as large as that of a physical keyboard, at least 16 mm [12, 24]. A full-sized soft keyboard cover about half of the tablet screen (assuming 8"-11" devices). This limits the utilization of the workspace during text entry and diluting the advantage of the large screen of tablet computers.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ISS '18, November 25–28, 2018, Tokyo, Japan

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM. ISBN 978-1-4503-5694-7/18/11...\$15.00

DOI: <https://doi.org/10.1145/3279778.3279801>

Figure 1. An example semi-transparent keyboard (STK) design

In response to this problem, semi-transparent keyboards (STK) has been introduced. An STK is a software keyboard that overlaps an application workspace (e.g., Figure 1). STK allows to see through the underneath application and text entry simultaneously. Some STK implementation examples in the market are KlearKey [25] for Android and TouchMousePointer [17] for Microsoft Windows.

Designing an STK encounters a trade-off between the text entry performance and the readability of background contents. A transparency of an STK may be a primary factor in this trade-off. A more careful blending of the two layers, keyboard and application contents, may enable higher recognizability of both layers, such as Multiblending technique by Baudisch and Gutwin [3]. An STK that is carefully designed considering many blending options may satisfy both keyboard efficiency and content readability. Furthermore, unlike opaque soft keyboards, an STK provokes another unfamiliar user experience: text under the keyboard. By default, key labels may distract users from concentrating on the typed text. If a caret moves below the Q-row, typing hands may occlude the text.

In this paper, we discuss following research questions related to STK design. First, how would the transparency level of an SKT affect the trade-off between keyboard efficiency and content readability? Second, what graphical features, in addition to the transparency level, would be important for STK design? Third, which caret positions would be acceptable beneath an STK? We designed and conducted three experiments to answer these questions. In the first experiment, we measured keyboard efficiency and content readability for four different

transparency levels. In the second experiment, we tried to answer the second question by performing an expert-driven STK design workshop. In the third experiment, we evaluated two best STKs from previous experiments and observed the effect of caret positions on keyboard efficiency and content readability to answer the third question.

In the remainder of the paper, we review related works, describe the design and results of the three experiments mentioned above, and summarize and discuss our findings.

RELATED WORKS

In this section, we briefly review the literature related to the idea of STK. The text entry performance is one of the most significant concern in STK design. In general, a larger touchscreen is more suitable for typing. Multiple studies pointed out the importance of sufficiently large key size. Kim et al. [12] found that the keyboards with smaller than 18mm key pitch decrease the typing performances. Similarly, Sears et al. [24] found that the typing speed and the error rate worsened with smaller keys. For this reasons, we focused STKs on tablet devices in this study.

Even with the larger touchscreen tablets, researchers has been sought to reduce the keyboard size as small as possible. 1Line keyboard [16] tried to merge three rows in the keyboard into one row. Arif et al. [1] introduced the minimal keyboard layout by assigning on keyboard gestures for space-hogging key, such as space bar. Likewise, Findlater et al. [5] eliminated the number row by providing multi-touch shape gesture for numbers and symbols. This study would be a new attempt in this context.

More radically, completely invisible keyboards have been tried. Zhu et al. [28] introduced an invisible keyboard on mobile devices by utilizing a decent touch point decoder. Invisiboard [21] facilitated a 9-key (T9) invisible gesture keyboard on a smart watch. One problem of invisible keyboard is a out-of-vocabulary word, because they are mostly rely on a probabilistic decoding.

In a semi-transparent interface, users can input a word in deterministic manner. However, two layers are sharing user's attention (*divided-attention*), so careful attention is required on its design. Researchers have tried to find a balance between background and foreground contents with transparent interfaces. Harrison et al. [8] found that overlapping solid images are less confused than text or wire-frame images. For more complex techniques, Multiblending [3] tried various image processing techniques such as desaturation, emboss, and blur on the different type of background images. In this work, assigning different blur, luminance and hue was a useful treatment for separation of two layers of text segments. Paley [23] explored design features for overlaid text, such as transparency, outline, hue or font variation, and motion. Leykin and Tuceryan [15] explored the text readability on textured backgrounds. They found that the contrast of text and the spatial frequency of background texture are most affects the text readability.

Regarding interactions, ToolGlass and Magic Lenses [4] introduced a concept of transparent see-through tool that works

like a filter. Kamba et al. [10] proposed transparent buttons that are selected by different touch durations. Hudson et al. [9] selected a target with various spatial gestures on an *overloaded* button. Hudson et al.[9] and Kamba et al.[10] tried to choose a layer by specific gestures. Gupta et al. [6] mapped different fingers on different layers within a single screen.

EXPERIMENT 1: EFFECT OF TRANSPARENCY

An STK needs to satisfy both the content readability and keyboard usability. In this study, we considered text-dominant applications with black characters on white background, which is common for word processors. Within this situation, the characters are distinctly distinguished by its extreme luminance contrasts and high spatial frequency features. By avoiding these features, an STK design may perceptually separate [7] the keyboard from the text and satisfy the both goals.

Figure 2. The reference keyboard and four derived STKs based on the Windows 8 stock keyboard.

From these aspects, the Microsoft Windows 8 stock keyboard could be a good starting point. This design does not have high-frequency decorative features, has only two solid dark colored backgrounds [8], and have simple white labels with a sans-serif font which is different typeface from the text [23]. By replicating this, we designed the reference soft keyboard as shown in Figure 2. By applying different transparency levels, we made four simple STKs as shown in Figure 2 (lower). In fact, the STK used in TouchMousePointer [17] shares the same approach.

Design

In this experiment, we used a within-subject design to evaluated the readability and usability of these STKs. The opacity level is the independent variable having four levels: 10%, 30%, 50% and 70%. Additionally, we added the reference keyboard condition as the baseline. Therefore, we have five conditions in total. The dependent variables were reading and writing performances. We used following metrics.

- ReadWPM = $\frac{\text{Length of the given text}}{5 \times \text{Elapsed reading time (min.)}}$

- $\text{WriteWPM} = \frac{\text{Length of the typed text}}{5 \times \text{Elapsed typing time (min.)}}$
- $\text{HarWPM} = \frac{2 \times \text{ReadWPM} \times \text{WriteWPM}}{\text{ReadWPM} + \text{WriteWPM}}$
- $\text{AriWPM} = \frac{\text{ReadWPM} + \text{WriteWPM}}{2}$
- Corrected Error Rate (CER, errors committed but corrected) and Not Corrected Error Rate (NCER, errors left in the transcribed task) as defined by Soukoreff and MacKenzie [26].

We defined HarWPM and AriWPM as the harmonic mean and arithmetic mean of ReadWPM and WriteWPM. HarWPM and AriWPM calculates the expected text processing speed in two different situations: when a user reads and writes the equal amount of text (HarWPM); reads and writes for the equal amount of time (AriWPM).

Participant

From a local university online bulletin board, we recruited ten paid (8\$) participants (mean age=23.6 years and $SD=3.17$ years). They have normal or corrected-to-normal vision, approximately CEFR (Common European Framework of Reference for Languages) B1 level English users, and are familiar with QWERTY layout.

Apparatus

The test program and STKs were implemented with Windows Presentation Foundation (WPF) on Microsoft Surface Pro 3, equipped with Windows 8.1, 12-inch touchscreen, 4GB RAM and 4th gen Intel i5 processor (Figure 3). The tablet has an adjustable flap for angle adjustment. We blocked the *windows* button at the right side of the device with a 1 mm thick felt pad and prevented any accidental interruption during the experiments. For the visual consistency, we calibrated the screen as sRGB color space and gamma 2.2 using Datacolor Spyder4Pro colorimeter. We controlled the display luminance to be 200 cd/m^2 . The ambient light illuminance during experiments was measured to around 500 lx.

We disabled all legacy touch events by tweaking the Windows registry¹, and implemented a device-level touch event hooking routine with Raw Input [20] for maximal responsiveness and to take control of every touch. We added a short beep sound (800Hz, 10ms) for every keystroke. We maintained the keyboard layout throughout the experiments and only changed its appearances. The keyboard only works with alphabets, enter, comma, period, quote, space, and backspace keys. All other keys were dummies. The text is displayed on the keyboard design and started at the Q row with 15mm margins from the screen edges. The text font was set to 12 pt regular Times New Roman, which has been a common formatting for document body. The key label font was set to 16pt regular Segoe UI, which has been a standard UI font for Windows 8.

Task

Participants performed two tasks: READ and WRITE task. One READ task requires reading text overlaid on a stimulus

Figure 3. Experiment apparatus

STK as shown in Figure 4a. A participant starts the task by clicking the *Reading* button with a mouse, then the text appears. The phrases in the text are randomly picked from the *memorable test set* by Vertanen and Kristensson [27]. The text length was controlled to be at least 400 characters. By clicking the *Reading* button again, a participant ends the reading. In the reference condition, only the text with white background appears.

One WRITE task requires transcribing the given phrase with a stimulus STK as shown in Figure 4b. The given phrase appears in between function and number rows to avoid visual interference. In contrast, the typed phrase overlaps the number row, and the STK is covered with dummy text. This task design maximizes the interference during typing with an STK. A participant starts a task by typing the first character and finalizes it by pressing the enter key. In the reference condition, the opaque keyboard in Figure 2 appears without the dummy text. The reference keyboard does not have number and function rows; therefore the given and typed phrases are clearly visible. The phrases are randomly picked from the *memorable test set* by Vertanen and Kristensson [27] with the length of 25 ± 5 characters. We converted all characters to lowercase characters and removed all punctuations.

Procedure

As a participant arrived, we instructed and demonstrated the READ and WRITE tasks. For READ task, we instructed participants to read the text in their usual speed, like reading a newspaper or an email. For WRITE task, we instructed participants to type as fast and accurate as possible. To minimize the performance fluctuation due to the error correction, we requested participants to correct errors only within the composing word and ignore the errors in the past words. In one block, each participant performed five READ trials and five WRITE trials for one condition. We have five conditions, therefore $5(\text{conditions}) \times (5 \text{ READ} + 5 \text{ WRITE}) = 25 \text{ READ} + 25 \text{ WRITE}$ trials were performed in one block. Each participant performed one practice block and two test blocks. During the practice block, we requested participants to adjust the tablet angle, placement, and chair position to find their best comfort typing posture. To minimize the order effect, we randomized the presentation order of STKs within a block. We instructed participants to rest for about 30 seconds every ten tasks. The whole procedure took about one hour.

¹KEY_CURRENT_USER/Software/Microsoft/Wisp/Touch/TouchGate=0

(a) Read task

(b) Write task

Figure 4. Screenshots of READ and WRITE task. The presented STK is in 50% opacity condition.

Results

We merged data from the two test blocks and removed erroneous tasks (two READ and one WRITE trials). We calculated averaged ReadWPM, WriteWPM, HarWPM, AriWPM, CER, and NCER for each participant. To address the individual skill differences, we normalized ReadWPM, WriteWPM, HarWPM, and AriWPM about the reference condition. We used the normalized values for the statistical analysis. The results are described below and displayed in Table 1 and Figure 5.

	10%	30%	50%	70%	Ref
<i>Read</i>	211	206	204	191	216
<i>N. Read</i>	97.5%	95.7%	94.5%	*89.0%	*100%
<i>Write</i>	42.0	44.6	46.8	45.2	47.4
<i>N. Write</i>	87.7%	92.8%	98.7%	96.0%	100%
<i>Har.</i>	68.6	72.1	75.0	72.3	77.0
<i>N. Har.</i>	88.6%	92.8%	97.4%	94.3%	100%
<i>Ari.</i>	126.3	125.2	125.3	118.1	131.9
<i>N. Ari.</i>	95.8%	94.9%	+95.0%	+89.5%	100%
<i>CER</i>	*12.0%	10.5%	*8.72%	9.16%	7.11%
<i>NCER</i>	1.05%	0.95%	0.84%	1.12%	1.27%

Table 1. Experiment 1 results. N. indicates the normalized values about the reference. Boldfaced values indicate the best performance conditions. The asterisk sign (*) means significantly different pairs and the plus sign (+) means marginally different pairs. (Abbreviations: Ref.=Reference, CER=Corrected Error Rate, and NCER=Not Corrected Error Rate.)

N. ReadWPM: A Repeated-Measure Analysis of Variance (RM-ANOVA) with Greenhouse-Geisser correction (due to the violation of sphericity $p=.009$) showed the main effect of the opacity level on normalized ReadWPM ($F(2.145, 19.307)=7.277$,

Figure 5. Scatter Plot of normalized ReadWPM and WriteWPM. Closer to the Reference condition (black) is better.

$p=.004$, $\eta^2_{\text{partial}}=.447$). *Post hoc* comparison with Bonferroni adjustments showed that Ref. condition was significantly faster ($p=.028$) than 70% condition in reading speed.

N. WriteWPM: A RM-ANOVA did not show the main effect of the opacity level on normalized WriteWPM ($F(4, 36)=1.839$, $p=.143$, $\eta^2_{\text{partial}}=.170$).

N. HarWPM: A RM-ANOVA did not show the main effect of the opacity level on normalized HarWPM ($F(4, 36)=2.141$, $p=.096$, $\eta^2_{\text{partial}}=.192$).

N. AriWPM: A RM-ANOVA showed the main effect of the opacity level on normalized AriWPM ($F(4, 36)=5.437$, $p=.002$, $\eta^2_{\text{partial}}=.337$). *Post hoc* comparison with Bonferroni adjustments showed no significant difference between conditions; but 70% condition was meaningfully slower than others, considering the large effect size ($\eta^2_{\text{partial}}=.0337$).

CER and NCER: A RM-ANOVA showed the main effect of the opacity level on CER ($F(4, 36)=3.099$, $p=.027$, $\eta^2_{\text{partial}}=.256$). *Post hoc* comparison on CER with Bonferroni adjustments showed that the 50% keyboard produced significantly less error than 10% keyboard ($p=.048$). A RM-ANOVA failed to find the main effect of the opacity level on NCER ($F(4, 36)=.263$, $p=.702$, $\eta^2_{\text{partial}}=.057$).

Conclusion

Overall, ReadWPM was decreased with the opacity. 70% opacity keyboard exhibited a severe readability problem. WriteWPM was increased with the opacity, but the effect was not significant. WriteWPM was apparently saturated at 50% opacity condition. 10% opacity condition exhibited more errors in WRITE task. In HarWPM and AriWPM as the unified metrics, the effect of opacity was weak. This could be explained by the trade-off relationship between reading and writing with different opacities. Amongst the conditions, we considered 50% opacity condition is well-balanced because it was the closest one to the reference (Figure 5). This result is consistent with a previous literature [8].

Figure 6. Controls for the toolkit. We grouped features (R1-R4) for easier explanation. The shaded features are excluded after the pilot study.

Figure 7. STK design toolkit screenshot and its controllable features.

EXPERIMENT 2: EXPERT-DRIVEN STK DESIGN

The goal of this experiment is to explore STK design space as diverse as possible. Designing STK involves numerous features, such as contrast between components [15], font styles [23], outlining [7], and blur [3]. For this experiment, we implemented an STK design toolkit with wide coverage of design features. Using the toolkit, we conducted a design workshop with six professional graphic designers for two months.

STK Design Toolkit

For the selection of features, we discussed with a variety of people: colleagues, designers, professors, and students in HCI and Industrial Design fields. Figure 6 and Figure 7 illustrate the STK design toolkit interface. The following list is description of features. Refer Figure 6 for the region labels (R1-R4).

- **Order(*)²**: Set the Z-order of content text and keyboard. With *Keyboard over Content* option, the keyboard opacity and colors affect the color of content text. With *Content over Keyboard* option, content text always appears in black.
- **R1**: Set the colors of key label and its shadow.
 - With *Luminance* option, the value slider (0 to 50) sets lightness difference between the key color and label/shadow colors. If the *L Inv.* checkbox is on, the

²The features marked with the asterisk(*) were excluded afterward.

label will be lighter, and shadow will be darker. If the checkbox is off, they will be reversed.

- With *Color* option, a designer manually assigns the label and shadow colors by a color-picker tool.
- *Shadow offset* slider (0 to 10 pixels, 2.5 pixels $\approx$ 1 pt) sets the distance between label and shadow. If this value sets to zero, a glow effect will be presented.

- **Opacity(*)**: Set the overall opacity ranging from 0.0 (invisible) to 1.0 (opaque).
- **R2**: Set the keyboard background and key colors in lightness³. Small slider bar(*) controlled its opacity.
- **R3**: Set the label font properties. The typeface was fixed to *Segoe UI* [19], which is a san-serif font purposely being apart from the serif font of the content text.
 - *Size*: 4 pt to 48 pt. 1 pt is 1/72 inch or 0.3527 mm.
 - *Weight*: 1 (thin) to 9 (black)
 - *Rotation*: -60 to 60 deg.
 - *Blur*: 0 to 20 pixels. Gaussian blur radius for labels.
- **R4**: Controls for save, load, and test functions.

Note: To minimize the color-related interference [3, 8], we allowed only solid gray colors to background and key colors. To avoid high-frequency spatial appearance, we applied blur [3] to the key borders.

The resultant toolkit has fifteen controls. Their combinations are almost infinite and, consequently, exploring this vast design space is cumbersome. We tried to scale down the complexity of the toolkit through a pilot study. Four voluntary participants (three design-major Ph.D. students and one HCI professor) examined the toolkit for an hour. We collected their opinions and simplified the toolkit as follow:

- We fixed **Order** to *Content over Keyboard*. Three out of four participants favored this option.
- By the decision above, we removed all three opacity sliders for overall, background, and key. Because the content text always appears in black on top of the keyboard, adjusting their colors can produce the equivalent effect.

The simplified toolkit was served for the following workshop.

³We used the lightness value in CIE Lab color model [18], which linearly maps perceived brightness of human vision, ranging from 0 (black) and 100 (white).

STK Design Workshop

The objective of this workshop was to obtain properly designed STKs by expert user interface (UI) designers. Six professional UI designers developed their own STKs using the STK design toolkit, through systematic design iterations.

Participants

Six voluntary participants (mean age=33.0 years and $SD=2.16$ years) were recruited. They have been working as UI designers for 5.3 years in average ($SD=1.75$ years) in a software company (Hancorn Inc.). They design UIs for office programs, including a word processor.

Apparatus

The STK design toolkit is implemented on two Microsoft Surface Pro 3 tablets, which are equivalent to the Experiment 1 apparatus. We additionally offered a simple evaluation tool that performs one READ or one WRITE task as in Experiment 1 and displays the measured speed. To avoid potential bias, they were not informed the result of Experiment 1.

Procedure

The workshop took eight days in two months (once in a week $\times$ eight weeks). At the beginning of the workshop, we thoroughly explained all the features and controls in the STK design toolkit and the evaluation procedures. We asked participants to design their STKs to be their best. We also collected free comments at the end of each day.

On the first day, each participant designed five STKs. On the second day, they evaluated the five STKs. The evaluation procedure is almost identical to Experiment 1 except the number of the trials performed: 10 READ trials and 24 WRITE trials per an STK, considering the experiment time. Among the five STKs evaluated, the experimenter picked two STKs which are, (1) The STKs showing statically better performance than others in ReadWPM or WriteRPM in pairwise t -test, or (2) If the statistical tests failed to find any significance, we picked the STKs preferred by the designers.

On the third day, we asked participants to design one new STK and two derived STKs from the two best STKs picked in the previous step. Therefore, three new STKs were obtained on the third day. On the fourth day, The participant evaluated three new STKs and two old but best STKs. Then the experimenter picked the two best STKs again. Rest of the days repeated the third-day and fourth-day procedures.

Finally, two best STKs per a participant were obtained from the last-day evaluation.

Results

Twelve STK designs (6 participants $\times$ 2 STKs) were obtained from the workshop. The resultant designs and their parameters are displayed in Table 2. From these designs, we found several converging features.

- All designers matched the background and key lightness values to eliminated the key border, namely borderless designs.
- Labels were brighter than the backgrounds (except C1).

- Designers did not prefer shadow offset, font rotation, and label blur.

From the comments, the major features they considered were background lightness, label font size and weight, and the color of label and shadow. The label only keyboard was sufficient for typing, and the label design may be the most important consideration in STK design. Also, brighter labels might help the perceptual separation of key labels from the black text-white background application color scheme.

Performance Evaluation of the Expert-Designed STKs

Although READ and WRITE task were considered for the evaluations during the workshop, designers also considered the other features, e.g., aesthetics. To evaluate the performances of the obtained STKs properly, we conducted an additional experiment. This experiment used a within-subject design with twelve STKs (Table 2) obtained from the workshop and the reference keyboard (Figure 2). Therefore, we have thirteen keyboards in total.

Participants

We recruited twenty six paid (8\$) participants (mean age=22.6 years and $SD=3.12$ years) who have normal or corrected-to-normal vision, approximately CEFR B1 level English users, and are familiar with QWERTY layout.

Procedure

We reused the procedure of Experiment 1 with a slight modification. Instead of the practice block, we instructed participants to examine both READ and WRITE tasks for at least five minutes. Each participant performed five READ trials and eight WRITE trials for each condition. We have thirteen conditions, therefore $13(\text{conditions}) \times (5 \text{ READ} + 8 \text{ WRITE}) = 65 \text{ READ} + 104 \text{ WRITE}$ trials were performed in total. The whole procedure took about one hour.

Results

Due to a program error, we removed 13 improperly logged data (3 READ and 10 WRITE trials) and calculated averaged ReadWPM, WriteWPM, HarWPM, AriWPM, CER, and NCER. We normalized the ReadWPM and WriteWPM about the reference. The results are displayed in Table 3 and Figure 8. Unlike Experiment 1, ANOVA analysis is not practical here because too many independent variables were involved. Instead, we tried to pick the probable best STK designs by choosing the local optimum instances. On Figure 8, A2, E1 and D2 form a Pareto front, which is a set composed of the local optimal points.

Conclusion

All A2, E1 and D2 are the best designs; A2 showed the best ReadWPM; E1 showed the best AriWPM and the least NCER; and D2 showed the best WriteWPM, HarWPM, and the least CER. In particular, D2 and E1 are interesting because they have similar appearances but are designed by different designers. They have about 60% lightness backgrounds, which are in fact 40% gray. This result is similar to the 30% and 50% opacity conditions in Experiment 1. Although the difference between designs are small, D2 showed better performances

	A1	A2	B1	B2	C1	C2	D1	D2	E1	E2	F1	F2
Bg	85.0	95.0	41.3	39.3	100.0	91.1	65.0	63.8	61.3	68.2	62.8	61.3
Key	85.0	95.0	40.3	40.3	100.0	91.9	65.0	63.8	61.7	68.0	61.6	62.4
Lbl.	FFFAAD	C4DFC2	F5F5F5	F5F5F5	B295FE	FFFFFF	FDF5E6	FDF5E6	FFF9D9	E6E0DB	-	-
Lbl.L.	83.9	81.8	96.1	96.1	79.0	100	94.7	94.7	92.5	88.0	68.6	69.0
Sdw.	DBDBDB	FFFFFF	331848	00CED1	FFFFFF	9C9C9C	FFF8DC	BDBCBC	F0F8FF	E6E0DB	-	-
Sdw.L.	85.9	100	18.8	41	100	61.2	93.1	73.9	97.1	88.0	54.5	55.8
Ofs	0	0	5	0	0	5	0	0	0	0	4	2
F.S	36.0	48.0	13.0	13.0	24.1	32.0	28.3	28.3	23.8	23.8	38.4	31.9
F.W	7	6	5	5	5	7	2	2	4	4	8	6
F.R	0	0	1	1	0	0	0	0	0	0	0	0
Blr	0	2	0	0	0	0	0	0	5	4	0	0

Table 2. The resultant designs obtained from the workshop. The alphabet (A-F) in an STK name indicates the designer code. The first column labels are abbreviated to save space. Their full names are listed in the following: Background lightness, Key lightness, Label color (hex), Label lightness, Shadow color (hex), Shadow lightness, Shadow offset (pixel), Font size (pt), Font weight, Font rotation (degree), and Blur radius (pixel). Note: 1) 1 pt = 2.5 pixel. 2) F1 and F2 used *Luminance* option; therefore the label and shadow colors are not written on the table. The full-sized pictures are in the supplement material.

	Read (n.)	Write (n.)	Har.	Ari.	CER	NCER
A1	207 88.4%	40.2 92.5%	66.6	123.5	13.1	1.74
A2	216 91.7%	40.1 91.5%	66.9	128.0	11.6	1.17
B1	203 87.0%	39.7 90.7%	65.8	121.3	12.8	1.25
B2	204 87.4%	40.9 93.2%	67.6	122.4	12.0	1.20
C1	206 88.1%	39.6 90.1%	65.5	122.6	13.7	1.04
C2	207 88.6%	40.0 91.2%	66.5	123.5	13.5	1.54
D1	207 88.4%	40.4 92.8%	67.1	123.6	13.3	1.34
D2	208 88.9%	42.1 96.7%	69.2	125.1	11.4	1.71
E1	212 90.1%	41.8 95.3%	68.3	126.8	12.9	0.87
E2	209 88.1%	40.0 91.7%	66.2	124.3	13.6	1.35
F1	204 87.6%	39.6 90.8%	65.8	121.9	13.6	2.08
F2	212 90.7%	40.1 91.2%	66.5	125.9	12.7	1.22
R.	239 100%	43.8 100%	73.1	141.6	11.2	1.42

Table 3. Experiment 2 results. Refer Table 1 for abbreviations.

Figure 8. Scatter plot of normalized ReadWPM and WriteWPM. A2, E1 and D2 form a Pareto front

than others in many measures. To conclude, we picked D2 as the one representative STK from this experiment.

EXPERIMENT 3: EFFECT OF CARET POSITIONS

Previous experiments examined STKs with a caret fixed at the number row. Unlike an opaque keyboard, a text caret may go under hands while using an STK. For more realistic situation, we unleashed the text caret and let it roam all around the screen in this experiment. We used a within-subject design with two STKs, namely KBD1 and KBD2 (Figure 9).

The goal of Experiment 3 was two-fold. The first was to empirically examine the effect of the caret position on the reading and writing performance. For this goal, we modified the experiment program to place the caret on 12 positions and measured reading speed, typing speed and error rates at each position. The second goal of the experiment was to compare the two best STKs from Experiment 1 and 2, respectively, in more diverse perspectives. They are two representative examples from the broad design space of STK. Although they were similar in terms of performance metrics, we hoped that the design option variations would make a difference beyond performance differences.

Figure 9. The two best STKs from Experiment 1 (KBD1) and 2 (KBD2).

Participants

From a local university online bulletin board, we recruited fifteen paid (8\$) participants (mean age=19.9 years and $SD=2.06$ years) who have normal vision, approximately CEFR B1 level English users, and are familiar with QWERTY layout.

Apparatus

On the apparatus of Experiment 1, we additionally implemented the caret position testing program (Figure 10). It can place a caret at any location inside an loaded STK. We made a 3×4 grid for test caret positions. We choose three horizontal grids considering left, center, and right align in document editing. We choose four vertical grids considering the occlusion by an STK and hands. The top row positions are entirely free from any type of occlusions. In the rest of the positions, the loaded STK and hands will occlude the typed text during typing. When testing one caret position, the rest part of the keyboard area were covered by dummy text (Lorem Ipsum...).

Figure 10. (a) The screenshot of apparatus. The red boxes show the twelve caret positions tested during Experiment 3. (b) Graphical explanation of ROAMING task.

Task

To evaluate the effect of caret positions, we designed ROAMING task. One ROAMING task requires a participant to memorize and transcribe a short phrase with a stimulus STK. As shown in Figure 10a, a given text appears at the top center of the screen. As shown in Figure 10b, a yellow-colored highlighter appears at one of the twelve test caret positions. The given text and highlighter disappear when a participant starts typing. Pressing enter finalizes a task. The phrases are randomly picked from the *memorable test set* by Vertanen and Kristensson [27]. We filtered out phrases having more than 25 characters for easier memorization. We converted all characters to lowercase characters and removed all punctuations. In this task, the reference condition was not possible to type with a cursor on the keyboard, so only the two STKs were tested.

Procedure

As a participant arrived, we instructed and demonstrated the READ and WRITE tasks and ask 5-minute practice as like in Experiment 2. The participant performed the READ and WRITE tasks with three conditions (two STKs + reference),

therefore $(3 \text{ conditions}) \times (5 \text{ READ} + 5 \text{ WRITE}) = 15 \text{ READ} + 15 \text{ WRITE}$ trials were performed. We composed this tasks to measure the baseline performance of two keyboards.

After short rest, we instructed the ROAMING task. We ensured participants to memorize the text and check the caret location before typing, type as fast and accurate as possible, and to ignore errors in the past words. The presentation order of STKs and caret positions were randomized. We assigned eight trials for each position, therefore each participant performed $(2 \text{ conditions}) \times (12 \text{ positions}) \times 8 = 192$ ROAMING trials. We instructed participants to rest for about 30 seconds every ten tasks. At the end of the test, we collected subjective score and comments about two STKs concerning reading, writing, and overall user experiences. The whole procedure took about one hour.

Results - READ and WRITE

As a baseline, READ and WRITE task (refer Figure 4) performance were measured. Table 4 shows the result. A RM-ANOVA showed the main effect on normalized ReadWPM only ($F(2, 28)=6.387, p=.005$), and *Post hoc* comparison with Bonferroni adjustments showed that the reading speed of KBD2 is slower than that of Reference ($p=.006$). All other metrics did not showed the significant main effects.

	Read (n.)	Write (n.)	Har.	Ari.	CE	NCE
KBD1	202 94.4%	45.9 102%	74.1	124	7.44	0.72
KBD2	198 92.5%	45.4 101%	73.1	122	7.06	0.33
Ref.	216 100%	45.6 100%	74.6	131	7.22	0.66

Table 4. Experiment 3 results for READ and WRITE tasks. Refer Table 1 for abbreviations.

Results - ROAMING

With ROAMING task, we measured typing speed on 12 caret positions (refer Figure 10). We excluded data points showing NCER larger than 20%. Because the ROAMING task required memorization, these data points are highly likely to come from the forgotten phrases. 47 trials were removed in total. We then calculated the WriteWPM, CER and NCER at each caret position. We normalized the values about the measured WriteWPM(=45.6 WPM), CER(=7.25%), and NCER(=0.52%) from the WRITE task result. We visualized the normalized WriteWPM at each position in Figure 11.

The two STKs are showing similar patterns. The top three positions clearly show the best performances. The lower left and right corners positions, which are frequently occluded by hands, show higher NCER and lower CER. This result means that participants carelessly typed in these positions without error correction efforts. Interestingly, the (2,2) position also shows the higher NCER than others, which means that more errors were not spotted even though the position is less occluded by hands.

Subjected Preference and Comments

Subjected preference scores (5-point Likert scale) were collected for KBD1 and KBD2. we visualized them using beanplot [11] in Figure12. The Mann-Whitney U test result on

Figure 11. Text entry performance at each position. The values represent a relative typing speed about the reference condition. Blue cells represents the value larger than 1.0, and red cells represents the value smaller than 1.0. For WriteWPM, higher values are better. For CER and NCER, lower values are better.

reading preference is $U=58.0$, $Z=-2.4$, $N=15$, $p=0.016$; on writing preference is $U=64.5$, $Z=-2.07$, $N=15$, $p=0.039$; and on overall preference is $U=75.0$, $Z=-1.66$, $N=15$, $p=0.098$ (marginal). From these results, we concluded that participants favored KBD2 in reading and KBD1 in typing situation. KBD2 was slightly more favored in overall.

Figure 12. A beanplot of subjective preferences. The thick lines indicate the averaged preference scores.

We organized the collected user comments on each STK in Table 5). In general, KBD1 was preferred in typing and KBD2 was preferred in reading.

Conclusion

In this experiment, the two STKs showed almost equivalent performances except for ReadWPM, where the KBD2 exhibited slightly worse reading speed than the other. However, from the survey results, participants generally favored KBD2 and disliked KBD1 in the reading situation.

We suspected two reasons for this contradiction. First, participants may not have enough time for READ task. According to Na and Suk. [22], the reading performance gradually increased over time and reached one's maximal performance after about two minutes. In our experiment, participants finished the READ test in six minutes. One-thirds of READ task data were collected under the non-optimal condition and may harm the statistical test. To confirm this suspicion, we performed the RM-ANOVA again with the last two-thirds of data,

Comments on KBD1

Positive	Negative
(10) Its clear segmentation of keys helped typing. (4) text reading was easier than expected. (4) I liked its familiar look-and-feel.	(7) The borders were distracting. (5) It occluded the text more. (3) It was harder to find the highlighter. (2) I want more brighter keyboard. (1) It was harder to see the keyboard.

Comments on KBD2

Positive	Negative
(7) Its borderless design helped reading. (4) It was easier to detect errors in the text. (4) This keyboard is neat and less distracting. (3) Because I know all the key positions, this design was better. (2) The keyboard had less contrast and it helped reading.	(6) I was hesitating to type without the key boundaries. (2) I made more mistakes. (1) It was more blurry. (1) Sometimes the labels distracted the text. (1) I prefer smaller labels.

Table 5. Positive and negative comments on both designs (sorted by their frequency)

and the effect of keyboard design on ReadWPM disappeared ($F(2, 28)=2.33$, $p=.116$, n.s). Second, participants was exposed to both READ and ROAMING tasks. The survey was conducted at the last part of the procedure. We suspect that KBD2 gained positive impressions during ROAMING task.

With ROAMING task, we reached a very commonsensical conclusion. Typing accuracies were degraded at the positions under hands (Figure 11). At the lower left and lower right positions, participants just typed the memorized phrases without trying to correct errors. Even though the typing speeds on those positions were not significantly reduced in this experiment, but it will eventually degrade the typing speed in long-term typing.

SUMMARY OF FINDINGS

This paper contribute to research on semi-transparent keyboard as follow:

- 1) Formal evaluation of on different opacities.
- 2) Exploration of STK design space.
- 3) Formal evaluation of newly introduced interaction (a text caret underneath a keyboard) on STK.

In the first experiment, we began with four STKs made by applying different opacity levels to a replica of stock Windows 8 keyboard. We found that 10% opacity is too transparent and degraded the typing performance, and 70% opacity is too opaque to read the text. The 30% and 50% opacity was balanced opacities. The overall performance (=HarWPM and AriWPM) of 50% condition reached 97% about the reference keyboard.

In the second experiment, we conducted an expert-driven STK design workshop. Six professional UI designers carefully designed STKs through the iterative design procedure in two months. The resultant STKs were clearly distinguished by their common borderless designs. They tried to improve the text readability by minimizing the contrast of the keyboard while conveying enough cues for typing through its labels. The performance evaluation on these STKs revealed that the best instance could reach 95% about the reference keyboard performance.

In the third experiment, we found that the two best STKs from the previous experiments are equally effective, and their performances are comparable to the reference keyboard. In user survey, STK with borders (KBD1) was preferred in typing, and the borderless STK (KBD2) was preferred in reading. Through the caret position experiment (ROAMING), we confirmed that a text caret may intrude the STK down to the number row without severe performance reduction. This result was consistent with the two tested STKs.

DISCUSSION

In this section, we discuss design recommendations, limitations, and new possibilities for future investigations.

Design Recommendation

From Experiment 3, we found that the borderless design was generally favored. However, some special keys may be confusing to be borderless. As a hybrid design, we suggest a keyboard design as shown in Figure 1. It maintains the look-and-feel of the borderless STK (KBD2), but some special key borders are visible (KBD1).

The STK preference may change according to the expertise of a user or the usage context. Novice typist pays more visual attention to the keyboard, but skilled typist relatively takes less attention [14]. In Experiment 3, the comments imply that the expert typists may prefer the borderless design. If typing can be done with minimal visual cues, or when a user performs a read-extensive task, less distracting STKs may be better, such as A2 design in Table 2.

With STK, a text caret may intrude the keyboard area and the keyboard blocks the direct manipulation of it. We recommend adding some simple on-keyboard gestures, such as multi-touch dragging gestures [13] to control a text caret. Additionally, following Experiment 3, we recommend the system with an STK to maintain a text caret above the home row (Figure 13).

Limitation

Throughout the experiments, we only considered text-only application content and the text always placed on top of the STK. With this approach, a large object, such as an image, will occlude the STK. We may overcome this problem by applying the Multiply blend function ($B(Cb, Cf) = Cb \times Cf$, where Cb and Cf are colors of background and foreground image) in the real implementation of STK. This yields equivalent appearance to *Content over Keyboard* option. For images, we simulated Multiply blending on a complex colored background using Adobe Photoshop (Figure 14). In addition, different blend functions may be applied with different contents.

Figure 13. Recommended caret position underneath an STK.

Figure 14. Multiply blending of an STK on a complex colored object

Although we designed STKs with our best effort, it was not possible to explore the entire design space. Moreover, the static power of the study is limited due to the small number of participants and exiguous performance difference between STKs. Notwithstanding, we suggested some plausible STK designs in the first and second experiment. Better STK designs may be obtained by longitudinal and iterative design improvements in situ.

New possibilities

A straightforward advantage of STK is the unobstructed application workspace. In other words, an STK is allowed to have less concern about its occupation, and it may show more keys such as number and function keys. These additional keys may reduce the mode switching cost and improve the text entry efficiency [2]. Also, a user can access shortcuts more easily.

Dynamic opacity change may be considered. An STK may stay less opacity (e.g., 10%) in idle, but progressively become more opaque (up to 50%) as a user types. With this approach, according to Table 1, reading speed may reach 97% and typing speed may reach 99% compared to the reference position.

Acknowledgment

This work was supported by the National Research Foundation of Korea(NRF) grant funded by Ministry of Science, ICT (NRF-2018X1A3A1070644).

REFERENCES

1. Ahmed Sabbir Arif, Michel Pahud, Ken Hinckley, and Bill Buxton. 2014a. Experimental Study of Stroke Shortcuts for a Touchscreen Keyboard with Gesture-redundant Keys Removed. In *Proceedings of*

- Graphics Interface 2014 (GI '14)*. Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 43–50.
<http://dl.acm.org/citation.cfm?id=2619648.2619657>
2. Ahmed Sabbir Arif, Michel Pahud, Ken Hinckley, and Bill Buxton. 2014b. Experimental Study of Stroke Shortcuts for a Touchscreen Keyboard with Gesture-redundant Keys Removed. In *Proceedings of Graphics Interface 2014 (GI '14)*. Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 43–50.
<http://dl.acm.org/citation.cfm?id=2619648.2619657>
 3. Patrick Baudisch and Carl Gutwin. 2004. Multiblending: Displaying Overlapping Windows Simultaneously Without the Drawbacks of Alpha Blending. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. ACM, New York, NY, USA, 367–374. DOI:
<http://dx.doi.org/10.1145/985692.985739>
 4. Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. 1993. Toolglass and Magic Lenses: The See-through Interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93)*. ACM, New York, NY, USA, 73–80. DOI:
<http://dx.doi.org/10.1145/166117.166126>
 5. Leah Findlater, Ben Lee, and Jacob Wobbrock. 2012. Beyond QWERTY: Augmenting Touch Screen Keyboards with Multi-touch Gestures for Non-alphanumeric Input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 2679–2682. DOI:
<http://dx.doi.org/10.1145/2207676.2208660>
 6. 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>
 7. Beverly L. Harrison, Hiroshi Ishii, Kim J. Vicente, and William A. S. Buxton. 1995a. Transparent Layered User Interfaces: An Evaluation of a Display Design to Enhance Focused and Divided Attention. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 317–324. DOI:
<http://dx.doi.org/10.1145/223904.223945>
 8. Beverly L. Harrison, Gordon Kurtenbach, and Kim J. Vicente. 1995b. An Experimental Evaluation of Transparent User Interface Tools and Information Content. In *Proceedings of the 8th Annual ACM Symposium on User Interface and Software Technology (UIST '95)*. ACM, New York, NY, USA, 81–90. DOI:
<http://dx.doi.org/10.1145/215585.215669>
 9. James Allan Hudson, Alan Dix, and Alan Parkes. 2005. User Interface Overloading: A Novel Approach for Handheld Device Text Input. In *People and Computers XVIII – Design for Life*, Sally Fincher, Panos Markopoulos, David Moore, and Roy Ruddle (Eds.). Springer London, 69–85. DOI:
http://dx.doi.org/10.1007/1-84628-062-1_5
 10. Tomonari Kamba, Shawn A. Elson, Terry Harpold, Tim Stamper, and Piyawadee Sukaviriya. 1996. Using Small Screen Space More Efficiently. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '96)*. ACM, New York, NY, USA, 383–390. DOI:
<http://dx.doi.org/10.1145/238386.238582>
 11. Peter Kampstra and others. 2008. Beanplot: A boxplot alternative for visual comparison of distributions. *Journal of statistical software* 28, 1 (2008), 1–9.
 12. Jeong Ho Kim, Lovenoor Aulck, Ornwipa Thamsuwan, Michael C Bartha, and Peter W Johnson. 2014. The effect of key size of touch screen virtual keyboards on productivity, usability, and typing biomechanics. *Human Factors: The Journal of the Human Factors and Ergonomics Society* 56, 7 (2014), 1235–1248.
 13. 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>
 14. Kip Knox. 2012. Designing the Windows 8 touch keyboard. Microsoft Blog. (2012). Retrieved September 14, 2016 from <https://blogs.msdn.microsoft.com/b8/2012/07/17/designing-the-windows-8-touch-keyboard/>.
 15. Alex Leykin and Mihran Tuceryan. 2004. Automatic Determination of Text Readability over Textured Backgrounds for Augmented Reality Systems. In *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR '04)*. IEEE Computer Society, Washington, DC, USA, 224–230. DOI:
<http://dx.doi.org/10.1109/ISMAR.2004.22>
 16. Frank Chun Yat Li, Richard T. Guy, Koji Yatani, and Khai N. Truong. 2011. The iLine 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>
 17. Lovesummertrue Software. 2015. TouchMousePointer. (2015). Retrieved June 1, 2018 from <http://www.lovesummertrue.com/touchmousepointer/en-us/>.
 18. K McLaren. 1976. XIII - The development of the CIE 1976 (L*a*b*) uniform colour space and colour-difference formula. *Journal of the Society of Dyers and Colourists* 92, 9 (1976), 338–341.
 19. Microsoft. 2013. SegoeUI. Font. (10 November 2013). Retrieved September 7, 2016 from <https://www.microsoft.com/typography/fonts/family.aspx?FID=331>.

20. Microsoft. 2016. Using Raw Input. MSDN. (2016). Retrieved September 7, 2016 from [https://msdn.microsoft.com/library/windows/desktop/ms645546\(v=vs.85\).aspx](https://msdn.microsoft.com/library/windows/desktop/ms645546(v=vs.85).aspx).
21. Aske Mottelson, Christoffer Larsen, Mikkel Lyderik, Paul Strohmeier, and Jarrod Knibbe. 2016. Invisiboard: Maximizing Display and Input Space with a Full Screen Text Entry Method for Smartwatches. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. ACM, New York, NY, USA, 53–59. DOI : <http://dx.doi.org/10.1145/2935334.2935360>
22. Nooree Na and Hyeon-Jeong Suk. 2014. Adaptive luminance contrast for enhancing reading performance and visual comfort on smartphone displays. *Optical Engineering* 53, 11 (2014), 113102. DOI : <http://dx.doi.org/10.1117/1.OE.53.11.113102>
23. W Bradford Paley. 2003. Designing better transparent overlays by applying illustration techniques and vision findings. In *Proceedings of the 16th annual ACM symposium on User interface software and technology, Supplement*. 57–58.
24. Andrew Sears, Doreen Revis, Janet Swatski, Rob Crittenden, and Ben Shneiderman. 1993. Investigating touchscreen typing: the effect of keyboard size on typing speed. *Behaviour & Information Technology* 12, 1 (1993), 17–22.
25. Touchscape Software. 2013. KlearKey. Application. (10 November 2013). Retrieved September 3, 2016 from <https://play.google.com/store/apps/details?id=com.touchscapesoftware.klearkeys.phone>.
26. 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>
27. Keith Vertanen and Per Ola Kristensson. 2011. A Versatile Dataset for Text Entry Evaluations Based on Genuine Mobile Emails. In *MobileHCI '11: Proceedings of the 13th International Conference on Human-Computer Interaction with Mobile Devices and Services*.
28. 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>