Accessible Gesture Typing for Non-Visual Text Entry on Smartphones

Syed Masum Billah
Stony Brook University
sbillah@cs.stonybrook.edu

Yu-Jung Ko
Stony Brook University
yujko@cs.stonybrook.edu

Vikas Ashok
Stony Brook University
vganjiguntea@cs.stonybrook.edu

Xiaojun Bi
Stony Brook University
xiaojun@cs.stonybrook.edu

IV Ramakrishnan
Stony Brook University
ram@cs.stonybrook.edu

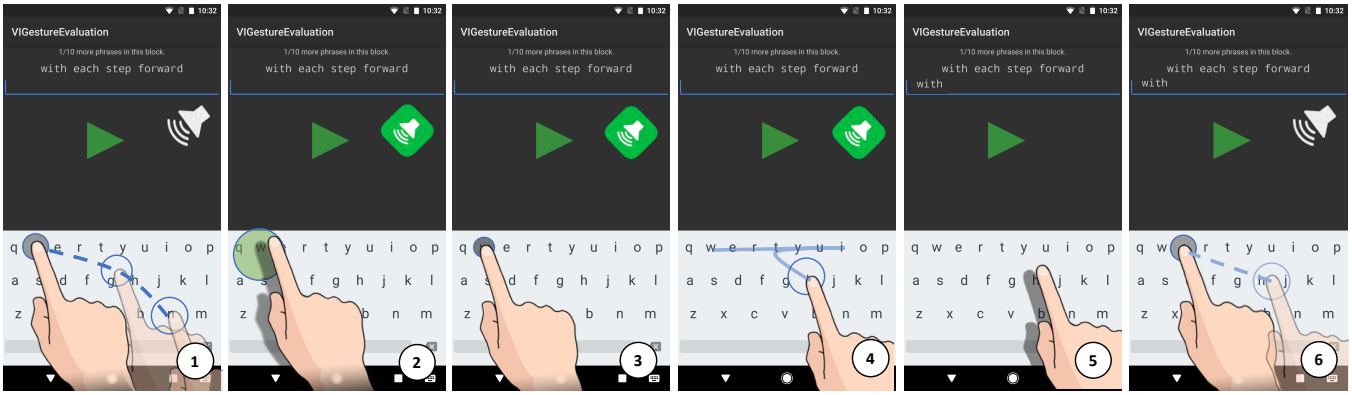

Figure 1: Illustration of word typing in proposed AGTex keyboard. It has two modes: *Exploration* and *Gesture*. (1): User explores the location of the first character of the target word in the Exploration mode. (2): The Gesture mode gets triggered once users lift their fingers off the first character. (3-4): If users land their fingers within the pre-defined sensitivity region centered on the selected key, they can continue to draw the gesture of the word to input the text. (5-6): When drawing the gesture of a word is done, users simply lift their fingers, and AGTex switches back to Exploration mode. The 'white' and 'green' volume-icons represent different earcons in Exploration and Gesture mode.

ABSTRACT

Gesture typing–entering a word by gliding the finger sequentially over letter to letter– has been widely supported on smartphones for sighted users. However, this input paradigm is currently inaccessible to blind users: it is difficult to draw shape gestures on a virtual keyboard without access to key visuals. This paper describes the design of *accessible gesture typing*, to bring this input paradigm to blind users. To help blind users figure out key locations, the design incorporates the familiar screen-reader supported touch exploration that narrates the keys as the user drags the finger across the keyboard. The design allows users to seamlessly switch between exploration and gesture typing mode by simply lifting the finger. Continuous touch-exploration like audio feedback is provided during word shape construction that helps the user glide in the right direction of the key locations constituting the word. Exploration mode resumes once word shape is completed. Distinct earcons help distinguish gesture typing mode from touch exploration mode, and thereby avoid unintended mix-ups. A user study with 14 blind people shows 35% increment in their typing speed, indicative of the promise and potential of gesture typing technology for non-visual text entry.

CHI 2019, May 4–9, 2019, Glasgow, Scotland Uk

ACM ISBN 978-1-4503-5970-2/19/05…$15.00
https://doi.org/10.1145/3290605.3300606

CCS CONCEPTS

• **H.5.2. Information interfaces and presentation: User Interfaces-input devices and strategies.**;

KEYWORDS

Gesture-typing; blind, non-visual; accessible text entry.

1 INTRODUCTION

Smartphone devices, embodying the convergence of computing, communication, and mobility, have inarguably transformed the lives of the vast majority of people worldwide. They have woven themselves inextricably into people's everyday lives and have become their indispensable hub for information and communication throughout the day. Thus, interacting with smartphones has also become an absolute necessity for people with vision impairments, both blind and low-vision, just like the vast majority of people worldwide who have no disability.

People with vision impairments rely on special assistive technology (AT), for interacting with computing devices. For blind people, the "go to" assistive technology is a screen reader that reads aloud the textual content of the screen in serial order, generally ignoring the layout and graphics. Because of the long legacy of desktop computing, many screen reading ATs have been developed for desktops (e.g., JAWS, VoiceOver, NVDA, and many more screen readers listed in [1]). On the other hand, the screen reading AT landscape for smartphones is quite limited, namely, iPhones's VoiceOver [25], and Android's TalkBack [22].

Interaction with smartphone applications very often necessitates text entry. Entering text is one of the most basic communication methods (e.g., emailing and messaging) and a foundation of common smartphone usage (e.g., searching, social networking, etc.). In fact, a recent study [5] has shown that 40% of iPhone users' activities are related to text entry. People with vision impairments typically rely on the smartphone's built-in virtual keyboard for typing text. For blind users, typical interaction with the virtual keyboard entails entering text *one letter at a time*. A blind user drags her finger across the keyboard to locate the letter; the screen reader announces every letter that is touched during this process. When the desired letter is announced, the user registers the letter by issuing any of the screen readers' built-in gestures, such (i) simply lifting the finger off the keyboard, or (ii) double tapping anywhere on the screen, or (iii) using another finger to tap on the screen (also known as Split-tap). This process is repeated until all the letters in the desired word are located and registered. Knowledge of the keyboard layout aids in efficiently locating the letters, but nevertheless entering letters one at a time is a tedious and slow process.

Gesture typing [12, 30] is an alternative text entry paradigm for smartphones that enables fast text entry by allowing users to enter an *entire word* at a time, rather than one letter at a time, as is done traditionally. Additionally, gesture typing also well suits touch interaction, relaxes the requirement of precisely selecting each letter, and supports a gradual and seamless transition to recall-based gesturing once the user gets familiar with the gesture shape and location. Thanks to these advantages, gesture typing is beginning to make inroads in practice; Google keyboard, Microsoft SwiftKey, TouchPal, and SlideIt are virtual keyboard products that support gesture typing.

Recall that a salient aspect of gesture typing is that an approximate knowledge of the key locations is sufficient to construct shapes for different words. This is straightforward for sighted people as they can see the keys all the time. Since blind users do not have access to this visual information, gesture typing is currently inaccessible to them. The accessibility challenge here is to provide a usable non-visual audio feedback that helps blind people figure out approximate key locations for accurate creation of word shapes.

This paper describes the design of AGTex - an **A**ccessible **G**esture typing keyboard for **Tex**t entry. Its development was informed by a pilot user study with blind smartphone users to explore the design space of possible accessible solutions for gesture typing. The study explored several questions that influence accessibility of gesture typing, namely, what kind of gestures are appropriate and usable for exploring the keys as well as for transitioning between key exploration and gesture typing, when and what type of audio feedback should be provided, should this feedback occur at discrete points or be provided continuously, when and what earcons should be used, etc.

The takeaways from the pilot study are reflected in the design of AGTex. In particular, to help blind users locate keys, AGTex incorporates the familiar screen-reader supported touch exploration that narrates the keys as the user drags the finger across the keyboard. AGTex allows users to seamlessly switch between exploration and gesture typing mode by simply lifting the finger. Continuous touch-exploration like audio feedback is provided during word shape construction that helps the user glide in the right direction of the key locations constituting the word. Exploration mode resumes once word shape is completed. Distinct earcons help distinguish gesture typing mode from touch exploration mode, and thereby avoid unintended mix-ups. Figure 1 is an illustration of word typing in AGTex.

A user study with 14 visually impaired people shows 35% improvements in their typing efficiency, indicative of the promise and potential of gesture typing technology for non-visual text entry.

We summarize our contributions as follows:

- Design space exploration for accessible gesture typing solutions on smartphones.
- The design and development of AGTex, the first of its kind accessible gesture typing keyboard.

- Findings from a user study that explored how 14 blind participants used AGTex for text entry on smartphones.

In the rest of this paper, we first review research related to the current work. We then present a pilot study which explored design options for accessible gesture typing keyboard. Informed by the study results, we designed and implemented AGTex, and carried out a user study to evaluate it.

2 RELATED WORK

Accessible Touch Interfaces

Accessible touchscreen interfaces have been a topic of substantial research interest ever since the advent of touch computing. Some of the earliest papers include [6, 24]; Buxton et al. in [6] outlined some of the issues associated with touch interfaces while Vanderheiden [24] proposed an audio-haptic interface for non-visual access to touchscreen appliances. The emergence of mobile devices, particularly smartphones, during the last decade spurred a flurry of research on the accessibility of mobile touchscreens. Problems with touchscreen accessibility and recommendations for addressing them appear in McGookin et al. [15]. Multi-touch interaction techniques are proposed by Kane [9] and Zhao [33], targeting the browsing of menus, selection of menu items, etc., non-visually. Gesture-based interaction with a touchscreen device encoding subway system information is described in Sanchez et al. [20]. In another work of Sanchez et al. [19], distinct functions are associated with fixed sections of the screen and gestures are used to trigger these functions. Kane et al. [10] explored the differences between how blind and sighted people use gestures for interacting with touchscreens.

Gesture Typing

Gesture typing is an alternative text entry paradigm for touchscreen devices. It was first introduced by Kristensson and Zhai in [12, 30]. It has a number of advantages over the typical tap typing: it is well-suited for touch interaction, is immune to one major problem plaguing regular touchscreen typing–the lack of tactile feedback, and allows users to enter words with approximate shape and location finger strokes.

Gesture typing has been extended to accommodate a variety of input modalities and support various scenarios. Bi et al. [4] created a bimanual gesture keyboard which allowed one word to be entered by multiple strokes using both hands; Markussen et al. [14] investigated mid-air gesture typing; Yu [29] explored using head movement to perform gesture typing; Yeo et al. [26] explored using device tilt angle for gesture typing.

Non-Visual Text Entry

A sizable volume of research has been conducted to support text entry for users with little or no vision. Voice input is commonly used, but it is still inadequate for protecting privacy, and socially inappropriate in many situations (e.g., in a bus, or a meeting room). Within the existing literature, most of the text entry techniques rely on tap typing. In Apple's VoiceOver [25] as well as in Android's TalkBack [22], audio feedback is given to the users when the finger touches a key on a QWERTY keyboard. The users can confirm a selection with a split-tap or double-tap. In No-Look Notes [25], letters are grouped and positioned near the center of the screen. To enter a letter, the users need to tap the desired group and then tap the intended letter within the group. In NavTouch [8], the alphabets are rearranged so that users can navigate through the letters with directional gestures, using the vowels as anchors. Tinwala et al. [23] described a text entry system that requires the user to trace complex forms corresponding to different letters (e.g., English letters) on the touch screen with their finger. In Yfantidis and Evreinov [27], a single finger directional gesture in any of the 8 compass directions (North, Northeast, South,..) corresponds to a unique character. Twenty-four characters are used and partitioned into 3 separate layers; the delay of the finger between touching the screen and commencing a gesture is leveraged for traversing the layers. Oliveira et al. [17] describe a study around mobile touch-based text entry tasks with blind users and conclude that individual differences impact performance. They argue that such differences call for "personalized" interaction design spaces. The SpatialTouch system in Guerreiro et al. [7] leverages the full-size Qwerty keyboards in tablets for two-handed multitouch exploration for text entry by blind users. A longitudinal study of the typing performance of blind users is reported in Nicolau et al. [16].

Researchers have also explored Braille-based text entry methods for people with visual impairments. For example, Perkinput [2] and BrailleTouch [21] allow users to enter each Braille letter by tapping multiple fingers on the touchscreen following its dot pattern. BrailleType [18] divides the screen into six large keys representing the six dots in a Braille cell. To enter a Braille letter, the users touch the keys corresponding to the dots pattern in the Braille letter.

In sum, non-visual text entry techniques for touchscreen devices have been well studied. By and large, these systems and their interfaces support one letter at a time text entry interaction. Gesture typing, by contrast, supports an entire word at a time interaction. Making gesture typing accessible, the topic of this paper, has not been explored so far.

3 DESIGN SPACE EXPLORATION FOR ACCESSIBLE GESTURE TYPING

The design and implementation of gesture typing was pioneered by Kristensson and Zhai [12]. In brief, it consists of two components: (1) the language model and (2) the spatial model. The language model provides the probability

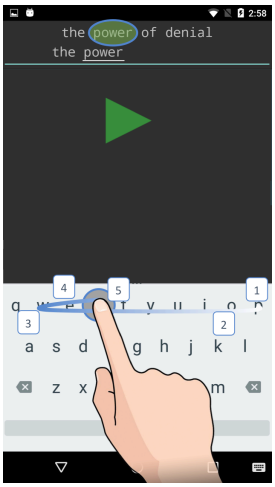

Figure 2: A screenshot of gesture typing. To enter "power", the user starts from the key 'p' (marked as 1) and then swipes towards keys 'o', 'w', 'r', and 'r'.

of a word based on the preceding words, while the spatial model provides the probability of a word based on geometric features of a gesture trace including its shape and location information on a virtual keyboard. Figure 2 shows a screen snapshot of gesture typing. The detailed description of such a decoder appears in [12]. Further, the cognitive theory behind gesture keyboard is the idea of a seamless skill transition from novice to expert behaviour. By repeating a gesture the user reaches motor memory consolidation and can transition from closed-loop visually-guidance based entry to open-loop direct recall from motor memory. Additional details on this theory, which appear in Kristensson et al.'s work [11, 13, 30–32], is beyond the scope of this paper.

As mentioned earlier, gesture typing has been designed with sighted people in mind since it is very straightforward to construct word shapes when one can see the virtual keys during the word construction process, which is done by gliding from one key to another in the general direction of the keys corresponding to the letters in the word. The problem addressed here, that has heretofore remained unexplored, is making it accessible for blind smartphone users.

Let's envision the gesture typing process of a blind user. We assume that these users have a mental map of the virtual keyboard's layout. Let's suppose the user wants to enter the word ***the***. The user will first locate the key for the first letter of the word, ***t*** in this case, by exploring the keyboard layout using screen reader. Next, the user will switch to the gesture typing mode using ***t***'s key as the starting point, i.e., the pivot. In this mode, the user glides in the direction of ***h*** first, and then in the direction of ***e***, without lifting the finger. Since the user has a mental map of the keyboard layout, she knows where ***h*** is located relative to ***t*** in a spatial sense and hence can glide in the direction towards ***h*** from ***t***. Similarly, she can glide from ***h*** in the direction of ***e***.

The key questions to be considered for making this process accessible are: what kind of gestures are appropriate and usable for exploring the keys as well as for transitioning between key exploration and gesture typing, when and what type of audio feedback should be provided, should this feedback occur at discrete points or be provided continuously, when and what earcons should be used. These choices constitute the design space for accessible gesture typing.

Our pilot study with blind users experimented with these choices. The study revealed what works and what does not, and these takeaways shaped the design of AGTex. We list below the choices that were explored in the pilot study. We refer to the key corresponding to the first letter of the word as the *pivot key*.

Design Space

Gestures. For exploring keys, we used the "exploration-by-touch" gesture [9]. For transitioning from exploration to gesture typing—blind users need to know when one ends and the other begins– we experimented with two design choices: a wiggle gesture, and an automatic time-out mechanism. Both require users to engage only one finger. This requirement of using one finger stem from the fact that gesture typing is traditionally done with one finger. Hence, any gesture that requires multi-finger (e.g., Split-Tap [9], earPod [33]) was excluded from our design choice. The wiggle gesture is done on the pivot key with the fingertip. It is implemented via thresholding – specifically, if the swiping direction changes more than 3 times either on the X or Y axis by more than 10 pixels in a second (see Figure 3). In automatic-time-out, which also uses thresholding, the user rests a finger on the pivot key for a predefined amount of time (e.g., 1250 ms) (see Figure 4).

Audio Feedback. Audio feedback was introduced in two places: (1) to announce successful transitioning to gesture-mode, and (2) during gesture typing. For (1), our options were: (i) to provide a distinctive earcon (e.g., long-beep), and/or (ii) speak out loud a distinctive phrase such as "gesture begins". For (2), our options were: (i) announce each key as users glide over them in gesture-typing mode, similar to the key readouts in touch-exploration, or (ii) stay mute throughout gesture typing mode and thereby avoid distractions.

Implementation

Following the principles in Kristensson et al. [12], we implemented our own decoder for the Android AOSP keyboard using a trigram language model with a 60K lexicon size. The description of such a decoder was detailed in [12].

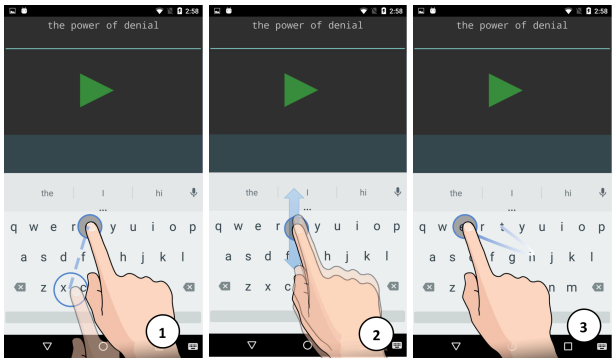

Figure 3: (1-2) Illustration of wiggle gesture-based transition. (3) GesTex1 prototype.

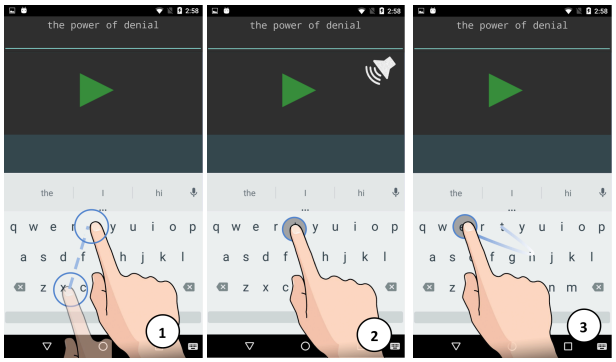

Figure 4: (1) Illustration of Timeout-based transition. (2-3) GexTex2 prototype.

The language model used in our prototypes had 60K unigrams, 20K bigrams, and 13K trigram. It was a Katz-smoothed language model trained on 114 billion words scraped from the publicly accessible web in English. We pruned the model size using entropy pruning, a technique for decreasing the size of a backoff language model. The perplexity on a test set derived from Enron data set was 167. Note that in principle any regular gesture typing decoder will work as building blocks for the techniques introduced in this work.

4 PILOT STUDY

Participants

We recruited participants for our IRB-approved pilot study through local mailing lists and word-of-mouth. After preliminary screening via phone-interviews, 6 blind subjects made the final cut for the study. We included participants who typed regularly on their smartphones using screen readers. None of the participants had any prior experience with gesture typing. Also, none of the participants had any motor impairment.

All participants were right-handed and varied in age from 29 to 56 (mean: 41.16 , SD:11.19), gender (3 men, 3 women), and typing experience on smartphones (3 experts, 2 intermediates, 1 beginner). Table 1 presents the participant demographics.

ID	Age /Sex	Phone Owned	Typing Expertise
P1	35/M	iPhone	Expert
P2	30/F	iPhone	Expert
P3	54/F	iPhone	Beginner
P4	56/M	iPhone	Intermediate
P5	34/M	iPhone	Expert
P6	29/F	iPhone	Intermediate

Table 1: Participant demographics in pilot study.

Task

The task required the participants to transcribe 10 pre-selected phrases of approximately equal length (between 4 to 6 words). These phrases were randomly selected from the T20 phrase set specified in [28]. The same set of phrases were used for all participants.

Design

The study was a within-subject design. We created two accessible prototypes for the pilot study—GesTex1 and GesTex2. These two prototypes captured the most relevant and important choices in the design space of accessible solutions described earlier in Section 3. We also included the default screen-reader based text entry in the study as baseline.

The participants performed the text-entry tasks under these 3 conditions:

- **GesTex1**: This prototype incorporated these choices: wiggle gesture for transitioning from exploration to gesture typing mode; voice confirmation to indicate commencement of gesture typing mode; no readout of keys in gesture typing mode (see Figure 3).
- **GesTex2**: This prototype incorporated these choices: automatic time-out for transitioning from exploration to gesture typing mode; distinctive earcon as well as voice confirmation to indicate commencement of gesture typing mode; continuous readout of keys in gesture typing mode (see Figure 4).
- **Default Screen Reader based text entry** (Default): The screen reader is used to interact with the default virtual keyboard and text is typed-in letter-by-letter.

To minimize the learning effect, we counterbalanced the ordering of study conditions and also the task sentences in each condition. Prior to performing the tasks in each condition, the participants were asked to type practice sentences using the text entry paradigm associated with that condition. We allotted ~15 minutes for each practice session.

The study was conducted with a Google Pixel phone running Android 7.1.2. It had TalkBack installed and was instrumented to log every user action in a log file on the device. The conversations during the study were conducted in English. The evaluation app (shown in Figure 3 and Figure 4) read out each phrase three times and on the fourth time, it read out the spelling of each word in the phrase. After entering a phrase, either the users or the experimenter clicked on the next button (indicated by the green arrow in Figures 3 and 4. for the next phrase. The experimenter took notes during the session. All sessions were video recorded and transcribed. Each session lasted from 1.5 to 2 hours.

Data Collection and Analysis

We analyzed the experimenter's notes, logs and recorded videos to measure the following metrics: (i) words per minute; (ii) word error rate; and (iii) number of backspaces pressed per word. At the end of the experiment, the participants were asked to give their feedback and recommendations for improvements.

Since the primary objective of the pilot study was to observe and understand users' gesture typing behavior, and furthermore since our sample size was small, we did not run any statistical significance test.

Results

We report here descriptive statistics from the pilot study along with our observations and users' feedback.

Input Speed (WPM). Typing speed was measured using the equation:

$$WPM = \frac{|S-1|}{T} * \frac{1}{5}$$

Here, S is the total number of transcribed characters during the task, T is the time elapsed in minutes to transcribe all the sentences in the task. Every five characters including the space were counted as one word.

The mean typing speeds under each condition are shown in Figure 5.a. The mean typing speeds (in WPM) were 4.80 for the default (SD: 2.65), 4.32 (SD: 4.05) for GesTex1, and 5.40 (SD: 1.93) for GesTex2. Compared to Default, the typing speed increased 12% in GesTex2, but decreased 10% in GesTex1, which indicates that even though both of our prototypes used the same gesture-typing underneath, the design choices made could potentially impact the typing speed.

Word Error Rate (WER). Word error rate [3], based on Minimum Word Distance (MWD), is the smallest number of word deletions, insertions, or replacements needed to transform the transcribed string into the expected string. The word error rate is defined as:

$$error = \frac{MWD(S,P)}{LengthInWords(P)} * 100\%$$

where $MWD(S,P)$ is the minimum word distance between transcribed phrase S and the target phrase P, and $LengthInWords(P)$ is the number of words in phrase P.

The mean error rates under each condition are shown in Figure 5.b. On average, participants made the most errors in GesTex1 (Mean: 16.75, SD: 13.22), followed by Default (Mean: 14.50, SD: 6.20), and GesTex2 (Mean: 14.30, SD:9.70). The error rate for GesTex1 was somewhat surprising because the gesture typing model gives the correct word as long as the user creates the word shape by gliding in the direction of the letters. It turns out that without any audio cues during word shape construction, as was implemented in GesTex1, users tended to deviate to a large degree from the neighborhood of the right key locations which resulted in unintended word shapes. But observe that announcing each key as the user glides over the keyboard, as was implemented in Gestex2, resulted in lower errors rates. Interestingly, these error rates were comparable to Default, implying that one can use the more efficient word-at-a-time text entry with error rates comparable to the letter-at-a-time text entry.

Deletes Per Word. Participants used over 600% more backspace-per-word in GesTex1 (Mean: 1.91, SD: 1.69) than Default (Mean: 0.25, SD: 0.21). However, they used 17% more in GesTex2 (Mean: 0.29, SD: 0.17) compared to Default. This is not surprising, since error rates and deletes are correlated.

Since the side effect of hitting the Backspace key is deletion of the word in gesture typing, and neither prototypes provided any explicit mechanism to "register" the Backspace key prior to deleting, we observed that participants were accidentally going over the Backspace key and deleting words unintentionally. Yet another observation was that even if users intended to delete a word, out of habit they would sometimes press and hold the Backspace button, which resulted in erasing multiple words.

(a) (b) (c)

Figure 5: Descriptive statistics from the pilot study; (a) Typing speed (WPM), (b) Error rate (WER), and (c) number of deletes per word in the 3 study conditions: Default, GesTex1, and GesTex2. Error-bars show ±1 SD.

Subjective Evaluation

We summarize here users' preferences and their recommendations that emerged from the pilot study.

Default vs. Gesture Typing. The participants stated that they were not accustomed to the "self-correction" capabilities of gesture typing in the sense one could create the word without having to know the exact key locations. Some felt disoriented when they heard some unrelated letters and tended to do course correction explicitly rather than letting the model do its work.

Transitioning with Wiggle vs. Time-out. All participants consistently rated the wiggle gesture as less usable than automatic time-out. Doing the wiggle gesture was found to be cumbersome and frustrating. The source of difficulties in doing the wiggle gesture included: (a) very slow finger movement; (b) fat fingers; (c) exaggerated finger movements; and (d) unintended finger movement to a neighboring key. (a) would cause failure to make the transition to gesture typing while all the others would result in picking the incorrect starting letter.

In the case of automatic timeout: 2 of the 6 participants mentioned they needed shorter time-out (less than 1250ms), because they could rapidly locate the first key (within 600 ms); 2 others wanted a longer time-out, as they were exploring the keyboard slowly and pausing briefly on each key before moving on to the next key. The remaining 4 participants who were comfortable with the time-out, said that they would not use it in practice, because they felt that upon locating the pivot key they had to dwell a little longer on the key for registering the transition to gesture typing mode. Instead they preferred a mechanism that would "instantaneously" confirm, akin to how a letter is typed in VoiceOver and TalkBack – users can confirm a selection by simply lifting their finger or double-tapping.

Audio Feedback: Earcon vs Voice Confirmation for Announcing Transitioning. All participants preferred the earcon and the voice confirmation equally and wanted both to co-exist. However, 3 participants expressed concerns about the "long beep" earcon used in the prototype. It reminded them of the beeps of a microwave . They also found it to be annoying. They suggested we instead use the familiar-sounding earcons of smartphone screen readers.

Audio Feedback: Key Readout in Gesture Typing Mode. With the exception of one participant, all participants wanted continuous read out of the keys in gesture typing mode. Without audio feedback, they were less confident and grew frustrated as they needed to do a lot of guess-work to figure out what keys they had encountered. In the absence of key read outs, they had difficulty controlling the direction and length of the strokes during each gesture. This problem was further exacerbated while trying to enter longer words such as 'shopping' and 'environment', where the participants had to repeatedly glide their finger from one end of the keyboard to another.

DesignParameter	Options	Comments
Pivoting	Wiggle	Strongly unfavorable
	Time-out	Neutral; Usability improvements recommended
Announcing Transition Completion	Speech	Favorable
	Earcon	Favorable; Changes recommended
Audio Feedback during Gesture Typing	Mute	Strongly unfavorable
	Key Readout	Strongly favorable

Table 2: Summary of design exploration.

P2 in Table 1 was the only participant who preferred not having any key readout during gesture typing. This participant remarked that she could picture the entire keyboard layout mentally, and even if she was uncertain about crossing the right letter, the gesture typing model was able to pick up the right word, thereby making the key read out unnecessary for her.

The findings of the design choices are summarized in Table 2. In the table "Pivoting" corresponds to the two mechanism choices for transitioning from exploration to gesture typing mode; "Announcing Transition Completion" corresponds to the two choices for audio feedback modalities; "Audio Feedback in Gesture Typing" corresponds to the two choices for providing audio feedback during gesture typing, namely the mute mode and read-aloud mode where the keys are announced as and when the user glides over them. The "Comments" column corresponds to how the choices were rated by the pilot study participants.

5 ACCESSIBLE GESTURE TYPING

The findings and recommendations of the pilot study revealed the most appropriate design choices that should go into an accessible gesture typing keyboard. These were adopted in AGTex, an accessible gesture typing keyboard for blind smartphone users. The design and evaluation of AGTex is described next.

Transitional Gesture

Neither the wiggle gesture nor the automatic timeout was well received as mechanisms for transitioning from exploration to gesture typing mode. Moreover, as soon as the pivot key is located, i.e., when they hear it read out, they do not immediately let go of the key as they either have to wiggle or

wait for a predefined time for registering the transition. They preferred a mechanism that would let them "instantaneously" let go of the pivot key as soon as it is located - akin to how they locate and enter a letter in VoiceOver and TalkBack.

In AGTex we provide such a mechanism. In particular, the user lets go of the pivot key as soon as it is located in exploration mode. At this point, the pivot key gets marked and transition to gesture typing mode is noted. The user touches the marked key again or any other region within the pivot key's locality, a configurable parameter, to complete the transition and begin gesture typing (see Figure 1.(1-2)). The idea of locality provides a degree of robustness by not requiring the user to land precisely on the pivot key and tap it to begin gesture typing.

If the user lands farther from the pivot key (either intentionally or unintentionally), AGTex automatically goes to exploration mode and plays the earcon chosen for this mode to notify the user.

In sum, the transitional gesture we introduced in AGTex can be expressed as an *explore-and-lift-and-re-land* gesture.

Multiple Earcons

Earcons were found to be very useful. In AGTex we provide two distinctive earcons, one for touch exploration, and another to indicate successful transition from touch-exploration to gesture mode. Users also recommended that we provide earcons that sound familiar. We recorded the earcon associated with touch-exploration in VoiceOver and used this familiar sounding earcon for touch-exploration in AGTex.

Keyboard Enlargement and Key Rearrangements

The height of the keyboard, and therefore the height of each key was increased to improve likelihood of correctly navigating to the neighborhood of the intended key while gesture typing. Placement of Enter and Backspace keys were altered. The former was removed, and Backspace was pushed down to the last row (see Figure 1), to create a set of rows exclusively for alphabet keys, thereby reducing the likelihood of unintended typing errors.

Placing the Backspace key on a separate row was unrelated to our transitional gesture, but was rather based on user behavior during gesturing. Specifically, participants were relying on the phone's left and right edges to quickly locate certain keys, e.g., ***p***, ***l***, and ***m*** near the right edge. Therefore, to avoid unintentionally deleting the entire word (while gesturing) by accidentally gliding over the Backspace key, which is to the immediate right of ***m*** at the right edge, we relocated Backspace key to a separate row.

Altering Default Behavior of the Backspace Key

The pilot study revealed that the default behavior of the Backspace key—delete on press— led to too many unintended typing mistakes. We overrode this behaviour so as not to delete on a single press. In the modified behaviour, the Backspace key will be readout first. Following the readout, a press-and-hold will delete the entire word, instead of a single char.

Again, the design of press-and-hold to activate the Backspace key was inspired by user behavior. We observed that participants tended to press-and-hold the Backspace for a longer duration even for correcting a single char mistake. We retained that behavior but added a protection so that it would not delete more than a word in a single press-and-hold.

ID	Age /Sex	Phone Owned	Typing Expertise
P1	35/M	iPhone	Expert
P2	30/F	iPhone	Expert
P3	38/M	iPhone	Expert
P4	56/M	iPhone	Intermediate
P5	31/F	Android	Intermediate
P6	60/F	iPhone	Beginner
P7	72/F	iPhone	Beginner
P8	68/M	iPhone	Intermediate
P9	25/F	iPhone	Intermediate
P10	46/M	iPhone	Beginner
P11	36/M	iPhone	Expert
P12	43/M	iPhone	Beginner
P13	29/F	iPhone	Expert
P14	34/M	iPhone	Intermediate

Table 3: Participant demographics in AGTex study.

Single Character Input

AGTex lets users input a single character as if it were a word with one character (e.g., ***a***)—after re-landing, a user simply lifts the finger up to enter a single character.

6 EVALUATION OF AGTEX

We conducted a new IRB-approved user study, three weeks after the initial pilot study, this time to evaluate AGTex.

Participants and Apparatus

We recruited 14 participants, 11 new and 3 (P1, P2, P4) were from the pilot study. All participants were VoiceOver users, and varied in age from 25 to 75 (mean: 42.92, median: 37, SD:15.3), gender (8 men, 6 women). Based on their self-reported typing skills, 5 were experts, 5 were intermediates, and the remaining 4 were beginners. Table 3 presents the participant demographics.

The experiment was conducted on the Google Pixel phone running Android 7.1.2 with TalkBack installed—the same phone that was used in the pilot study.

Task and Design

The participants performed the same transcription task that was used in the pilot study. Since participants spent 20-30

(a) (b) (c)

Figure 6: Evaluation of AGTex. (a) Typing speed, (b) Error rate, and (c) number of delete per word in 2 conditions: Default and AGTex. Error-bars show ±1 SD.

minutes for entering 10 phrases in each condition in that study, we did no increase the number of phrases. Increasing this number would introduce fatigue as a confounding factor. However, we chose a different set of 10 sentences that were randomly drawn from Yi et al. [28].

The study was also a within-subject design. Since our objective was to improve the typing performance on smartphones, we retained the same baseline that was used in the pilot study, namely, default screen reader typing. The participants performed the tasks under these 2 conditions:

- **AGTex**: Participants use AGTex for text entry (Figure 1).
- **Default Screen Reader based Typing (Default)**: Participants use the screen-reader to interact with the default keyboard, and type in text letter by letter.

The rest of the study design mirrored the pilot study, discussed in Section 4.

Results

Paired t-tests are used to determine statistical significance. We used Shapiro-Wilk to check normality, and for non-normal data that required a non-parametric test, we used the Wilcoxon Signed Rank test. Figure 6 shows the descriptive statistics in the final study.

Input Speed (WPM). We measured the input speed in WPM defined in Section 4. The participants were 35% faster with AGTex (Mean: 5.66 WPM, SD: 2.29) compared to the baseline (Mean: 4.16 WPM, SD: 2.32), which was statistically significant ($t_{13} = -3.224, p < .007$).

An increase in typing speed was expected. Recall, we already observed an increase in typing speed in the GexTex2 prototype in the pilot despite a latency 1250ms per word, which is how long it took to transition to gesture typing mode from exploration mode in GesTex2. On the other hand, in AGTex, transition to gesture-mode occurs almost as soon as users let go of their finger from the pivot key.

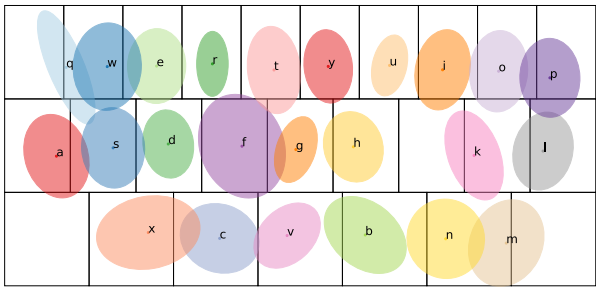

Figure 7: Touch point distribution in AGTex (95% confidence ellipses).

Word Error Rate (WER). Overall, error rate (defined in Section 4) decreased 62% in AGTex (Mean: 14.87, SD: 10.21) compared to baseline (Mean: 23.77, SD: 22.17), which was expected. However, this reduction was not statistically significant ($t_{13} = 1.827, p = .091$). Note that, with Default, the mean and standard deviation (SD) of WER went up compared to the pilot study (mean: 14.50 → 23.77, SD: 6.20 → 22.17). One explanation is the typing skills of the participants in our pilot study were mostly intermediate to expert, whereas in the final study, skills varied widely.

Deletes per Word. The deletes/word also decreased 47% in AGTex (Mean: 0.49, SD: 0.51) compared to baseline (Mean: 0.93, SD: 0.83). However, a Wilcoxon signed ranks test did not show statistical significance (Z = -1.538, p = .124), which was quite surprising. We assumed that since the role of the delete key is different in AGTex which is erasing a whole word as opposed to erasing a single character, participants would have used it less frequently.

Touch-point Distribution

To obtain a deeper understanding of how blind users glided over the keys during gesture typing in AGTex, we investigated their touch-point distribution (see Figure 7). Note that in the T20 evaluation set, no word contained the letters ***z*** and ***j***, hence there is no distribution associated with these two keys. The distribution shows that participants deviated the most for letters ***q*** and ***m***, whose keys are located at the top-left and bottom-right respectively. Observe in the figure that distribution center of these two keys crossed their key boundaries. The deviation for the keys in the middle was limited. These deviations reveal visually that users need not know the exact key location. They can deviate from the exact key locations and yet the gesture typing model will shape match to the correct word.

Observations and Subjective Feedback

All participants stated that gesture typing is faster than default character-by-character typing since users can ignore mistakes while moving their finger from one character to another while constructing the shape gesture for a word; they can rely on the self-correction capability of the gesture typing model to correctly guess the intended word. However, in default typing, whenever an incorrect character is entered, the general tendency of users is to erase the incorrect character and then retype the correct one.

Participants learned how to transition from exploration to gesture typing in AGTex within a few (under 3) minutes. However, we observed that a few participants forgot to lift their finger before making a gesture. But, they immediately realized their mistake upon hearing the exploration mode's earcon and were able to recover quickly.

We also observed that at the beginning of their practice, some participants were lifting their finger way up and missed the "sensitivity region" when landing back. But they quickly adjusted this behaviour – very soon lifting their finger only a tiny bit over the key became the norm.

With the exception of P7 and P8, all participants rated AGTex as highly favorable over default text entry and expected to use it in the future. P2 said that AGTex was "very addictive", and P14 said that typing in a smartphone is so frustratingly slow that he sometimes turned off VoiceOver (using shortcut) during composing a text, but with AGTex he believed he could type as fast as sighted users.

7 DISCUSSION

The original gesture keyboard envisions that a user would: (i) transit from the more visual novice mode to the less visual recall-based expert mode, and (ii) draws gestures in a continuous stroke rather than key-by-key entry. But these 2 properties did not hold for blind users, as (i) absence of visual feedback of gesture traces hinders the memorization of gesture shapes and locations, preventing them from transiting to recall-based expert mode; and (ii) unlike sighted users who can visually locate the target char and directly gesture to it, blind users navigate character-by-character to reach the target key (via audio cues). AGTex retains the gesture typing paradigm by including new features to compensate for the absence of visual feedback.

Metrics for Typing Speed and Error Rate. We used the same metrics (e.g., WPM and WER) for both gesture-typing and baseline to make the performance comparison easy to comprehend. For this reason, we kept the numerator in WPM calculation in gesture-typing to $|S-1|$, even though the appropriate measure was $|S|$ since the SPACE key (i.e., the "-1" part) was not required in gesture-typing. The effect of this change was barely noticeable as the average phrase length in our study was 35-45 chars long. For example, the mean and the SD in AGTex would have changed from 5.66 to 5.86, and 2.29 to 2.38 respectively. Therefore, conclusions on input speed remain unchanged. Similarly, we reported WER for both gesture-typing and baseline, instead of WER for gesture-typing and char error rate (CER) for baseline.

Auto-correction and Auto-suggestion. The participants had difficulty entering certain words, such as 'disaster', 'company', 'environment'. Often times incorrect word shape gestures were the result of not knowing the word. In such situations, the gesture typing model was unable to do any kind of correction since the model matches shapes corresponding to words. A potential improvement would be to offer suggestions, similar to regular keyboard but integrated into the gesture paradigm.

Two-thumb Typing. We observed a few participants were using two-thumbs for typing. These participants had difficulty making gestures at the beginning. They said they had developed different muscle memory for each key, and while using AGTex for the first time, they had to re-adjust their mental model. These individuals asked for multi-touch gesture inputs.

Multi-finger Transitional Gesture. The *explore-and-lift-and-re-land* transitional gesture in AGTex was in line with blind users' current char-by-char typing behavior, where they *explore* a character first and and then *lift* their finger up to enter it. However, we envision that AGTex typing could co-exist with the char-by-char typing by incorporating a multi-finger gesture (e.g., Split-tap), which would let the user to switch between these two typing techniques seamlessly.

8 CONCLUSION

As a widely supported text entry paradigm for smartphones, gesture typing has a number of advantages over regular tap typing. However, it was designed with sighted people in mind, and is inaccessible to blind users. This paper explores the utility of gesture typing for blind smartphone users. Towards that, we developed an accessible gesture typing keyboard AGTex. Its development was shaped by a pilot study with blind participants. The study helped uncover the parameters that are critical for designing accessible gesture typing. A user study with 14 blind participants shows 35% improvements in their typing speed which was statistically significant. Furthermore, the error rate decreased by 62% – all of these are indicative of the promise and potential of gesture typing technology for non-visual text entry on smartphones.

Acknowledgement. We thank anonymous reviewers for their insightful feedback. This research was supported in part by NSF: 1806076, NEI/NIH: R01EY02662, NIDILRR: 90IF0117-01-00, and a Google Faculty Research Award (2018).

REFERENCES

[1] AFB. 2018. Screen Readers. http://www.afb.org/ProdBrowseCatResults.aspx?CatID=49. Accessed: 2018-09-10.

[2] Shiri Azenkot, Jacob O. Wobbrock, Sanjana Prasain, and Richard E. Ladner. 2012. Input finger detection for nonvisual touch screen text entry in Perkinput. In *Proceedings of Graphics Interface 2012*. Canadian Information Processing Society, 2305297, 121–129.

[3] Xiaojun Bi, Shiri Azenkot, Kurt Partridge, and Shumin Zhai. 2013. Octopus: Evaluating Touchscreen Keyboard Correction and Recognition Algorithms via. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 543–552. https://doi.org/10.1145/2470654.2470732

[4] Xiaojun Bi, Ciprian Chelba, Tom Ouyang, Kurt Partridge, and Shumin Zhai. 2012. Bimanual gesture keyboard. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. ACM, 2380136, 137–146. https://doi.org/10.1145/2380116.2380136

[5] Barry Brown, Moira McGregor, and Donald McMillan. 2014. 100 days of iPhone use: understanding the details of mobile device use. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices and services*. ACM, 2628377, 223–232. https://doi.org/10.1145/2628363.2628377

[6] W. Buxton, R. Foulds, M. Rosen, L. Scadden, and F. Shein. 1986. Human interface design and the handicapped user. *SIGCHI Bull.* 17, 4 (1986), 291–297. https://doi.org/10.1145/22339.22386

[7] Joao Guerreiro, Andre Rodrigues, Kyle Montague, Tiago Guerreiro, Hugo Nicolau, and Daniel Goncalves. 2015. TabLETS Get Physical: Non-Visual Text Entry on Tablet Devices. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2702373, 39–42. https://doi.org/10.1145/2702123.2702373

[8] Tiago Joao Vieira Guerreiro, Paulo Lagoa, Hugo Nicolau, Daniel Goncalves, and Joaquim A. Jorge. 2008. From Tapping to Touching: Making Touch Screens Accessible to Blind Users. *IEEE MultiMedia* 15, 4 (2008), 48–50. https://doi.org/10.1109/MMUL.2008.88

[9] Shaun K. Kane, Jeffrey P. Bigham, and Jacob O. Wobbrock. 2008. Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques. In *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility*. ACM, 1414487, 73–80. https://doi.org/10.1145/1414471.1414487

[10] Shaun K. Kane, Jacob O. Wobbrock, and Richard E. Ladner. 2011. Usable gestures for blind people: understanding preference and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1979001, 413–422. https://doi.org/10.1145/1978942.1979001

[11] Per Kristensson. 2007. *Discrete and Continuous Shape Writing for Text Entry and Control.* Doctoral dissertation. LinkÃűping University.

[12] Per-Ola Kristensson and Shumin Zhai. 2004. SHARK: a large vocabulary shorthand writing system for pen-based computers. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*. ACM, 1029640, 43–52. https://doi.org/10.1145/1029632.1029640

[13] Per Ola Kristensson and Shumin Zhai. 2007. Command Strokes with and Without Preview: Using Pen Gestures on Keyboard for Command Selection. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, USA, 1137–1146. https://doi.org/10.1145/1240624.1240797

[14] Anders Markussen, Mikkel Rønne Jakobsen, and Kasper Hornbæk. 2014. Vulture: A Mid-air Word-gesture Keyboard. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1073–1082. https://doi.org/10.1145/2556288.2556964

[15] David McGookin, Stephen Brewster, and WeiWei Jiang. 2008. Investigating touchscreen accessibility for people with visual impairments. In *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*. ACM, 1463193, 298–307. https://doi.org/10.1145/1463160.1463193

[16] Hugo Nicolau, Kyle Montague, Tiago Guerreiro, Andre Rodrigues, and Vicki L. Hanson. 2015. Typing Performance of Blind Users: An Analysis of Touch Behaviors, Learning Effect, and In-Situ Usage. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, 2809861, 273–280. https://doi.org/10.1145/2700648.2809861

[17] Joao Oliveira, Tiago Guerreiro, Hugo Nicolau, Joaquim Jorge, and Daniel Goncalves. 2011. Blind People and Mobile Touch-based Text-entry: Acknowledging the Need for Different Flavors. In *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '11)*. ACM, New York, NY, USA, 179–186. https://doi.org/10.1145/2049536.2049569

[18] Joao Oliveira, Tiago Guerreiro, Hugo Nicolau, Joaquim Jorge, and Daniel Goncalves. 2011. BrailleType: Unleashing Braille over Touch Screen Mobile Phones. Springer Berlin Heidelberg, 100–107.

[19] Jaime Sanchez and Fernando Aguayo. 2007. Mobile Messenger for the Blind. In *ECRIM'06*. Springer Berlin Heidelberg, 369–385.

[20] Jaime Sanchez and Eduardo Maureira. Year. Subway Mobility Assistance Tools for Blind Users. In *ECRIM'06*. Springer Berlin Heidelberg, 386–404.

[21] Caleb Southern, James Clawson, Brian Frey, Gregory Abowd, and Mario Romero. 2012. An evaluation of BrailleTouch: mobile touchscreen text entry for the visually impaired. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*. ACM, 2371623, 317–326. https://doi.org/10.1145/2371574.2371623

[22] TalkBack. [n. d.]. TalkBack: An Open Source Screenreader For Android. http://google-opensource.blogspot.com/2009/10/talkback-open-source-screenreader-for.html

[23] Hussain Tinwala and I. Scott MacKenzie. 2010. Eyes-free text entry with error correction on touchscreen mobile devices. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*. ACM, 1868972, 511–520. https://doi.org/10.1145/1868914.1868972

[24] Gregg C. Vanderheiden. 1996. Use of Audio-Haptic Interface Techniques to Allow Nonvisual Access to Touchscreen Appliances. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 40, 24 (1996), 1266–1266. https://doi.org/10.1177/154193129604002430

[25] VoiceOver. [n. d.]. Screen reader from Apple. https://www.apple.com/accessibility/iphone/vision/

[26] Hui-Shyong Yeo, Xiao-Shen Phang, Steven J. Castellucci, Per Ola Kristensson, and Aaron Quigley. 2017. Investigating Tilt-based Gesture Keyboard Entry for Single-Handed Text Entry on Large Devices. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 3025520, 4194–4202. https://doi.org/10.1145/3025453.3025520

[27] Georgios Yfantidis and Grigori Evreinov. 2006. Adaptive blind interaction technique for touchscreens. *Universal Access in the Information Society* 4, 4 (2006), 328–337. https://doi.org/10.1007/s10209-004-0109-7

[28] Xin Yi, Chun Yu, Weinan Shi, Xiaojun Bi, and Yuanchun Shi. 2017. Word Clarity as a Metric in Sampling Keyboard Test Sets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 3025701, 4216–4228. https://doi.org/10.1145/3025453.3025701

[29] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. 2017. Tap, Dwell or Gesture?: Exploring Head-Based Text Entry Techniques for HMDs. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 3025964, 4479–4488. https://doi.org/10.1145/3025453.3025964

[30] Shumin Zhai and Per-Ola Kristensson. 2003. Shorthand Writing on Stylus Keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 97–104. https://doi.org/10.1145/642611.642630

[31] Shumin Zhai and Per Ola Kristensson. 2012. The Word-gesture Keyboard: Reimagining Keyboard Interaction. *Commun. ACM* 55, 9 (Sept. 2012), 91–101. https://doi.org/10.1145/2330667.2330689

[32] Shumin Zhai and Per Ola Kristensson. 2007. *Introduction to Shape Writing*. 139–158. https://doi.org/10.1016/B978-012373591-1/50007-3

[33] Shengdong Zhao, Pierre Dragicevic, Mark Chignell, Ravin Balakrishnan, and Patrick Baudisch. 2007. Earpod: eyes-free menu selection using touch input and reactive audio feedback. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1240836, 1395–1404. https://doi.org/10.1145/1240624.1240836