EyeSayCorrect: Eye Gaze and Voice Based Hands-free Text Correction for Mobile Devices

Maozheng Zhao
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
mazhao@cs.stonybrook.edu

Henry Huang
Tappan Zee High School
Orangeburg, NY, USA
1henry.huang@gmail.com

Zhi Li
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
zhili3@cs.stonybrook.edu

Rui Liu
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
ruiliu1@cs.stonybrook.edu

Wenzhe Cui
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
wecui@cs.stonybrook.edu

Kajal Toshniwal
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
kajaltoshniwal16@gmail.com

Ananya Goel
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
angoel@cs.stonybrook.edu

Andrew Wang
Ward Melville High School
East Setauket, NY, USA
andrewwxng@gmail.com

Xia Zhao
Stony Brook Medicine
Stony Brook, NY, USA
xia.zhao@stonybrookmedicine.edu

Sina Rashidian
Verily Life Sciences
Cambridge, MA, USA
srashidian@cs.stonybrook.edu

Furqan Baig
University of Illinois at
Urbana-Champaign
Urbana, IL, USA
fbaig@cs.stonybrook.edu

Khiem Phi
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
kphi@cs.stonybrook.edu

Shumin Zhai
Mountain View, California, USA
zhai@acm.org

I.V. Ramakrishnan
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
ram@cs.stonybrook.edu

Fusheng Wang
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
fushwang@cs.stonybrook.edu

Xiaojun Bi
Department of Computer Science,
Stony Brook University
Stony Brook, NY, USA
xiaojun@cs.stonybrook.edu

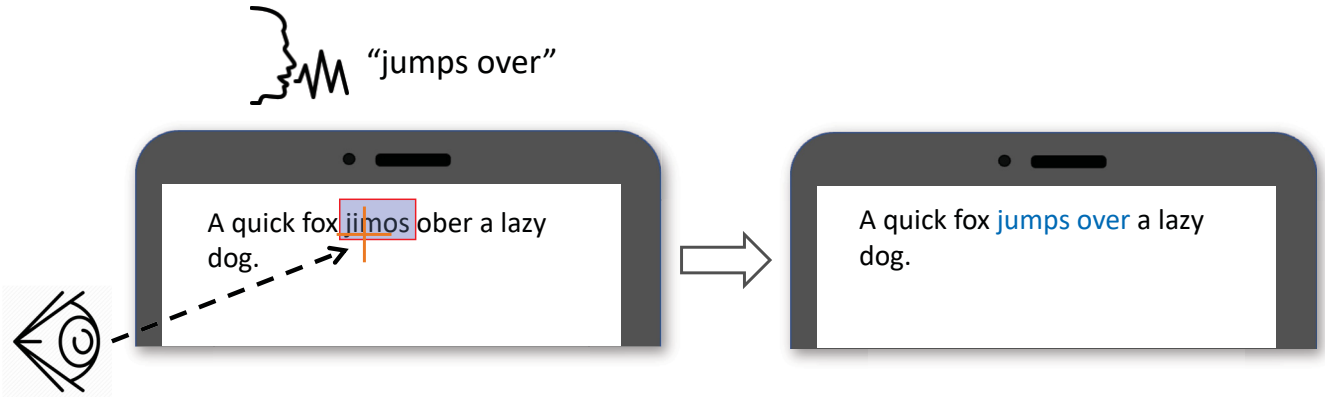

Figure 1: Demonstration of EyeSayCorrect. To correct errors in the text, the user first utilized the gaze location on the screen to select a word, and then spoke the new content for correction. The text was corrected after the user finished speaking.

ABSTRACT

Text correction on mobile devices usually requires precise and repetitive manual control. In this paper, we present EyeSayCorrect, an eye gaze and voice based *hands-free* text correction method for mobile devices. To correct text with EyeSayCorrect, the user first utilizes the gaze location on the screen to select a word, then speaks the new phrase. EyeSayCorrect would then infer the user's correction intention based on the inputs and the text context. We used a Bayesian approach for determining the selected word given an eye-gaze trajectory. Given each sampling point in an eye-gaze trajectory, the posterior probability of selecting a word is calculated and accumulated. The target word would be selected when its accumulated interest is larger than a threshold. The misspelt words have higher priors. Our user studies showed that using priors for misspelt words reduced the task completion time up to 23.79% and the text selection time up to 40.35%, and EyeSayCorrect is a feasible *hands-free* text correction method on mobile devices.

CCS CONCEPTS

• **Human-centered computing** → **Text input**; **Mobile devices**; **Interaction techniques**.

KEYWORDS

multimodal interaction; eye gaze; text correction; voice input.

ACM Reference Format:

Maozheng Zhao, Henry Huang, Zhi Li, Rui Liu, Wenzhe Cui, Kajal Toshniwal, Ananya Goel, Andrew Wang, Xia Zhao, Sina Rashidian, Furqan Baig, Khiem Phi, Shumin Zhai, I.V. Ramakrishnan, Fusheng Wang, and Xiaojun Bi. 2022. EyeSayCorrect: Eye Gaze and Voice Based Hands-free Text Correction for Mobile Devices. In *27th International Conference on Intelligent User Interfaces (IUI '22), March 22–25, 2022, Helsinki, Finland.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3490099.3511103

IUI '22, March 22–25, 2022, Helsinki, Finland

ACM ISBN 978-1-4503-9144-3/22/03…$15.00
https://doi.org/10.1145/3490099.3511103

1 INTRODUCTION

Correcting text is a core activity we perform daily on mobile devices. Despite that it's essential for searching, emailing, messaging, and social networking applications, it's difficult to perform. The requirement of precise and repetitive manual control remains a hindrance. For example, the default cursor-based text correction technique requires precisely positioning the cursor after the erroneous text, repeatedly clicking the backspace buttons, typing the new text, and re-positioning the cursor back to the original position.

Besides finger touch, eye gaze and voice are alternative input modalities we could leverage for text correction. However, eye gaze may not be precise and stable enough for text operation where text is usually small. It's difficult to use eye gaze to precisely select a text span. Voice input is prone to speech recognition errors, especially in noisy environments. And it's unsuitable for specifying the location of the error.

In this paper, we designed and implemented EyeSayCorrect, an eye gaze and voice based error-tolerant multimodal text correction system for mobile devices. To correct errors, the user first selects a word close to the error location by the gaze location on the screen. Then, the user speaks the correct word or phrase. EyeSayCorrect would correct the error based on the word selected by eye gaze, voice input, and the text context. We utilized the advantages of each modality and let the two modalities supplement each other. We used a text correction algorithm from [91] which can process ambiguous input by considering the context of text using a language model, editing distance, and word embedding distance. For selecting text by eye gaze, we proposed a Bayesian based two-dimensional target selection method which can accumulate a target word's interest even when the gaze location is outside of the target. Given each sampling point in an eye-gaze trajectory, the posterior probability of selecting a word is calculated and accumulated. The target word would be selected when its accumulated interest is larger than a

threshold. To facilitate selecting misspelled words for correction, we assigned higher priors for misspelled targets words. Our eye gaze and voice based multimodal text correction system offers users a *hands-free* approach for correcting words or phrases and can accommodate ambiguous and noisy input signals.

Our user studies showed that using priors for misspelt words significantly reduced the task completion time, especially the text selection time by eye gaze, and EyeSayCorrect is a feasible *hands-free* text correction method on mobile devices.

2 RELATED WORK

As background of the current work, we review previous techniques for text correction, multi-modal interactions, gaze tracking technologies and gaze based target selection.

2.1 Techniques for Text Correction on Mobile Devices

Text correction is a crucial part of text entry on mobile devices [39, 60]. The cursor-based method is well explored in previous research. The cursor can be moved by touch [2], arrow keys [83], gestures [22, 31, 88], combining taps and gestures [23]. A number of assistive methods for text selection are also explored, such as gesture based methods [22, 31], Gaze'N'Touch [66] which combines eye gaze and taps to select text.

Cursor-based methods on mobile devices often have challenges due to small screen sizes and the fat finger problem [9, 28, 82]. Intelligent auto-correction were introduced to address those problems. However, auto-correction only corrects the word currently being entered [10, 25, 81]. It cannot correct text that has already been entered. Grammar checking methods such as Gboard [46] and Grammarly [32] support correcting entered text by providing possible correction suggestions in a menu. Recent methods "Type, then Correct" technique [87] and "JustCorrect" [14] added more intelligence to the post hoc text correction for reducing cursor operations.

There are also voice-based text correction methods [27, 55]. However, those methods were cumbersome for indicating text locations.

2.2 Multimodal Interaction Technologies on Mobile Devices

Multimodal interaction has benefits such as being natural, more error tolerant [40, 57, 58], and flexible [16, 59]. Previous research explored combining multiple modalities to reduce text entry ambiguity, such as using handwriting to correct speech recognition errors[79], combining gesture typing and speech [56, 73], using touch to indicate speaking word boundaries for better speech recognition performance [72], using unistrokes and key landings [34] to speedup text input, combining eye gaze and keyboard to for text editing on desktops [75]. Recent soft keyboards (e.g., Gboard [46]) supports voice and touch input, but the two modalities are often used separately. Multimodal method were also applied to disambiguation interfaces [43, 49, 68].

iOS's voice control [30] and Android Voice Access [26] supports combining voice and touch to edit text. However, they are not error-tolerant since they still require precise text selection and precise text content for error correction. VT [91] is able to accommodate ambiguous input from touch and voice, it does not require precise text selection or precise text content for error correction. It infers a user's correction intention from touch input, voice input and text context by utilizing language model, Word2Vec distance, and Levenshtein distance.

EYEditor [24] presented a smartglass-based text-editing that allows selecting text with a hand controller and inputting new words with voice. Our proposed EyeSayCorrect used eye gaze to select a word and it endowed users with more freedom about which word to select and which phrase to speak. Talk-and-Gaze [70] explored combining voice and eye gaze modalities for text correction. Gaze dwelling was used to select a word. To correct a word, users used voice to either speak the index number of a suggested word or spell the the new word letter by letter. EyeSayCorrect used a Bayesian method for eye-gaze based word selection which was more tolerant for noisy gaze locations. And users can directly speak the new phrase to correct the text.

2.3 Eye Tracking Technologies

Eye tracking technology has become increasingly available and advanced nowadays. Technologies for eye tracking include eye attached tracking using special contact lens, skin potential measurement with electrodes, and optical tracking without direct contact to the eye, i.e. infrared light based or image feature based [92]. There are many commercial products such as Tobii [80], EyeGaze Edge [19], SMI REDn [77], Eyelink 1000 plus [20] and PRC Accent 1400 [64]. Many eye-gaze related studies were conducted on top of them [18, 33, 41, 69, 71, 74]. On the one hand, existing eye trackers can provide high resolution, high sampling rate and high eye tracking quality for users. On the other hand, they come at a price and with a learning curve.

Besides these dedicated eye tracking devices, researchers are also devoting to enabling eye tracking for general-purpose embedded cameras on daily devices such as Android smartphones, iPhone and iPad. For example, Wood et al. proposed EyeTab [86], a model-based gaze estimation on unmodified tablet computers. Papoutsaki et al. proposed WebGazer [61], an online eye tracker that uses common webcams in laptops and mobile devices to infer gaze position in real-time. Huang et al. proposed an in-situ gaze estimation method on the glint of the screen on the user's cornea, using only the image captured by front-facing camera on smartphones [29]. There are also many Deep Learning techniques proposed to support gaze-tracking [38, 85, 89]. Li et al. took advantage of Apple's ARKit to enable eye tracking on an iPad Pro with TrueDepth camera [44]. Our work used the same method for gaze tracking as [44].

2.4 Gaze Based Target Selection

Gaze-based target selection is a core activity for a number of applications such as gaze-based text input [65], gaming [36] or smart device control [67]. Dwell-based method (Dwell) [35, 37, 90] is the most widely used target selection method. However, dwelling gaze on a target for a specific period often results in eye fatigue [62]. There were works trying to reduce the dwelling time such as letting users adjust the dwell time manually [48], using Fitts' law to reduce the dwell time [35], adjusting the dwell time based on the probability of each letter during text entry [53, 63].

In addition to dwell-based methods, specific UI designs that can facilitate gaze base target selection were explored [47, 51, 76]. Actions such as blinking [12] and gaze gesture [15] have also been used for target selection. Multimodal input such as a keyboard input [42], hand-held touchscreen input [78], EMG input [50] and the head movement [71] were used to replace the dwelling action.

Bayes' theory were well applied to accommodate uncertainty for touch based target selection [11, 84, 93], it's also applied for gaze-based interactions [8, 54], gaze-to-object mapping problem [8]. BayesGaze [44] applied it for target selection. Our work improved BayesGaze for text target selection. BayesGaze was designed for menu button targets, it assumed that the distribution of targets been selected follows a Zipf' distribution [1, 13, 17, 45, 52, 93], and updated priors after each selection. For our text correction problem, the targets are words. Most words may never been selected again once been corrected. So the BayesGaze' prior model which requires multiple updates to be effective is not suitable for our problem. We propose to assign higher prior values for misspelt words since they are more likely to be the targets in text correction problem. BayesGaze only accumulates interest from gaze sampling points inside the target area. Since text targets are much smaller than menu buttons, it's much more difficult to keep gaze location inside a word target. We extend the area for accumulating interest to a larger ellipse area whose semi axes are proportional to the width and height of the targets, so that a target can continue accumulate interest even when the gaze location sways out of the target area. This could better accommodate noisy gaze trajectories for small targets. In [44], BayesGaze is only applied to 1D targets. In our work, the text targets are two-dimensional. We used a 2D Gaussian likelihood function for computing the posterior likelihood. BayesGaze used raw accumulated interest for target selection while we normalized the accumulated interest to the range of 0 to 1. The advantage of normalizing the accumulated interest is that we can visualize the normalized accumulated interests as visual feed backs for users. The visual feed backs are important for users to understand the current state of selecting a target.

3 EYESAYCORRECT

3.1 Workflow Design

In this paper, we propose EyeSayCorrect, in which users can select a word by eye gaze and speak a new phrase. EyeSayCorrect would generate correction suggestions based on the selected word, the speech input, and the text context. The top suggestion would directly replace the original text. The second and third top suggestions are shown for selection. The workflow of using EyeSayCorrect to correct text is shown in Figure 2. In Figure 2, (1) shows the text before a user starts to correct it. (2) shows that the location of a user's gaze location (the red cross). A word's background intensity represents the normalized accumulated interest for that word. (3) shows a word was selected. If a word is selected, there would be a red bounding box around it. The speech recognition would start when the red bounding box appeared. (4) means that the user speaks the new content for correction. (5) shows the default correction result after the user spoke the new content. The new word or phrase in the text would be shown in blue color for 3 seconds. This serves as visual feedback showing which part of the text was changed. (6) shows the alternative suggestions for users to select. The user can skip selecting suggestions if the default correction result was correct.

If a word is selected, all the words stop accumulating interest until the selection is canceled or the text is changed. The selection will be canceled if no speech is recognized after a waiting period of five seconds. If there is partial speech recognition results in five seconds, the selection will remain until the speech finishes. Once the speech finishes, the text will be changed by EyeSayCorrect based on the speech recognition transcripts and the selected word. We adopted this two-step design, selecting the word first and then speaking the new content, instead of the one-step design, gazing at the word while speaking. Because based on our observation when the user is speaking, the gaze trajectory usually sways in a large range. It's difficult for users' gaze to remain on a specific word while speaking. Once the user stops speaking, the correction results and suggestions would show. Users can select the suggestions if necessary. If the default correction result and suggestions are not users' correction intention. Users can click the "Undo" button to undo the last editing.

The implementation of eye gaze tracking on iPad Pro is described in Section 3.2. For word selection, EyeSayCorrect used a Bayesian based method that accumulates each sampled gaze location weighted by a spatial Gaussian model. It's described in Section 3.3 to Section 3.6. For phrase correction, EyeSayCorrect supports correcting a phrase by voice without precisely selecting the whole phrase. It's described in Section 3.7.

3.2 Eye gaze tracking on iPad Pro

In this paper, we followed [44] to use an 11-inch iPad Pro for the user study. The gaze tracking is implemented based on Apple's ARKit [4]. ARKit is Apple's augmented reality software development kit that enables face and eye-tracking on iPhone/iPad with Apple's TrueDepth [6] camera system. With ARFaceAnchor [3] in ARKit, we used the *leftEyeTransform* and *rightEyeTransform* to locate the position and orientation of the face's left and right eye and perform the hit testing *hitTestWithSegment* to compute the gaze location on the screen. We also used an Outlier Correction Filter with a triangle kernel [21] to smooth the detected gaze trajectory. The sampling rate of gaze trajectory was 60Hz and sampled gaze locations on the screen were processed in real-time. The parameters of the filter were set as the same as in [44].

3.3 Bayesian based word selection

Selecting a word among text can be considered as a gaze-based target selection problem, each word is a target for selection. Given a gaze trajectory $\mathbb{S} = \{s_1, s_2, \ldots, s_K\}$ where s_i is a sampling point along the gaze trajectory at time i, and a set of candidate targets denoted by $\mathbb{T} = \{t_1, t_2, \ldots, t_N\}$, where each t_j is a word target in the text. The goal of the problem is to decide which target in t_j is the intended target. Each target t_j accumulates "interest" from each sampling point s_i along the gaze trajectory until one of them reaches a threshold (denoted by θ) for being selected.

For each target, if the gaze sampling point s_i is inside an ellipse area around the word, the word would accumulate interest from that gaze sampling point, otherwise the total interest of that target

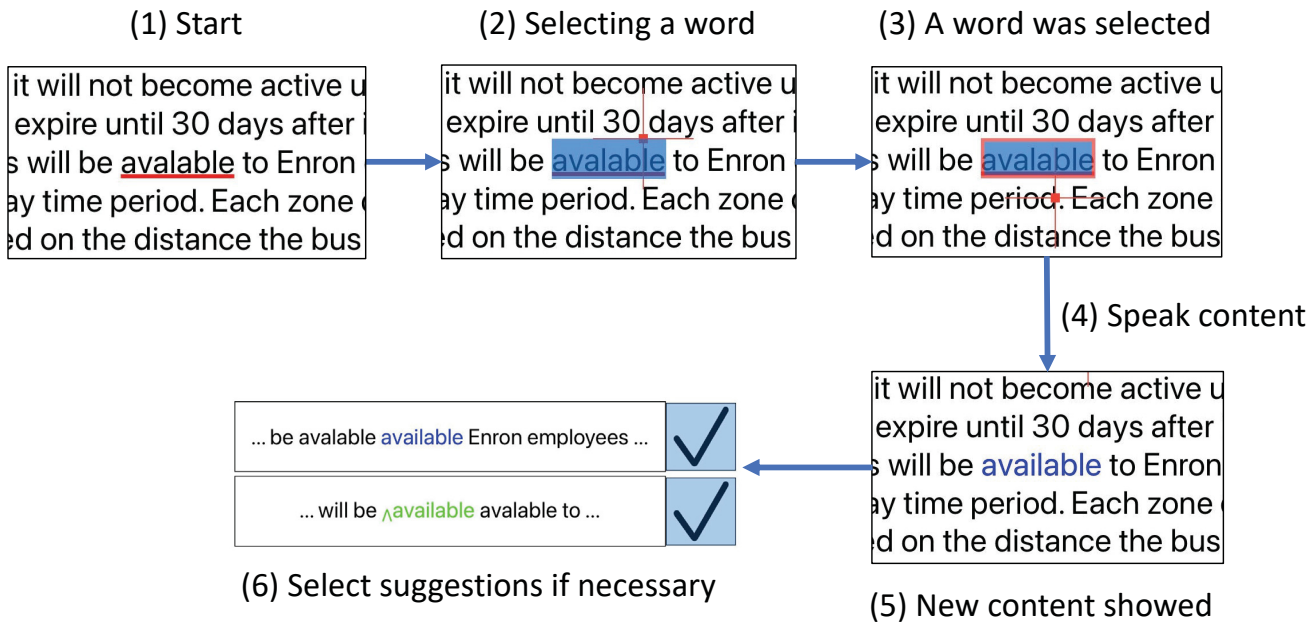

Figure 2: Workflow of EyeSayCorrect. 1) shows the text to be corrected. 2) shows the user's gaze location (the red cross). A word's background intensity represents the normalized accumulated interest for that word. 3) shows a word is selected. If a word is selected, it would have a red bounding box. The speech recognition would start when the red bounding box appeared. 4) represents the process of speaking new content for correction. 5) shows the default suggestion that automatically replaced the erroneous word. 6) shows two alternative suggestions for users to select by eye gaze if necessary.

is set to 0. The ellipse area is decided by the width w_j and height h_j of the 2-dimensional target t_j, the ellipse's two semi axes are $\alpha * w_j$, and $\beta * h_j$. Assuming we observe a gaze sampling point s_i as (s_{i_x}, s_{i_y}), the accumulated interest of selecting a target is defined as the posterior probability of selecting a target given a sampling point based on Bayes' theorem weighted by the sampling interval.

$$I_i(t_j) = \begin{cases} 0, \text{ if } \frac{(s_{i_x}-\mu_{j_x})^2}{(\alpha * w_j)^2} + \frac{(s_{i_y}-\mu_{j_y})^2}{(\beta * h_j)^2} > 1 \\ I_{i-1}(t_j) + \Delta\tau \cdot P(t_j|s_i), \text{ otherwise} \end{cases} \quad (1)$$

where $P(t_j|s_i)$ is the posterior probability which can be estimated according to Bayes' theorem. (μ_{j_x}, μ_{j_y}) is the center coordinate of target t_j. α and β are parameters that define the shape of the ellipse area around a target for interest accumulation. We set α and β much larger than 1, this means the ellipse area was significantly larger than the target itself. This was critical for our text target selection task. Because text targets were usually small, the gaze trajectory may often sway around a word instead of staying inside a word. Setting α and β larger than 1 can accumulate interest for a target when the gaze sampling point is outside the word target.

Assuming there are N target candidates,

$$P(t_j|s_i) = \frac{P(s_i|t_j)P(t_j)}{P(s_i)} = \frac{P(s_i|t_j)P(t_j)}{\sum_{k=1}^{N} P(s_i|t_k)P(t_k)}, \quad (2)$$

where $P(t_j)$ is the prior probability of target t_j being the intended target without observing the current gaze input trajectory, and $P(s_i|t_j)$ is the probability of s_i if the intended target is t_j (the likelihood).

3.4 Dual-Gaussian Likelihood Model for gaze target selection

A dual-Gaussian likelihood model is used to calculate $P(s_i|t_j)$:

$$P(s_i|t_j) = \frac{1}{2\pi\sigma_{j_x}\sigma_{j_y}} \exp\left[-\frac{z}{2(1-\rho_j^2)}\right], \quad (3)$$

where

$$z \equiv \frac{(s_{i_x}-\mu_{j_x})^2}{\sigma_{j_x}^2} - \frac{2\rho_j(s_{i_x}-\mu_{j_x})(s_{i_y}-\mu_{j_y})}{\sigma_{j_x}\sigma_{j_y}} + \frac{(s_{i_y}-\mu_{j_y})^2}{\sigma_{j_y}^2}. \quad (4)$$

σ_{j_x} and σ_{j_y} are the standard deviations of the dual-Gaussian model for target t_j, and ρ_j is the correlation coefficient between x and y.

3.5 Priors for misspelt word

In the text correction task, the misspelt words usually have higher probabilities to be a selection target for text correction. We set the prior value $P(t_j)$ for misspelt words higher than that of correctly spelt words.

$$P(t_j) = \begin{cases} \gamma, \text{if the word target is misspelt} \\ 1, \text{otherwise} \end{cases} \quad (5)$$

where γ is a constant larger than 1. In our implementation, γ was set to be 2 for EyeSayCorrect with priors. We used the *UITextChecker* in Swift to detect the misspelt words in the text.

3.6 Softmax normalization

The accumulated interests for all targets are normalized by softmax.

$$I'_i(t_j) = \frac{e^{I_i(t_j)}}{\sum_{k=1}^{N} e^{I_i(t_k)}} \tag{6}$$

If a target's normalized accumulated interest $I'_i(t_j)$ is larger than a threshold θ, the target is selected. In our implementation, the θ value was selected by experience. It's a trade-off between the target selection speed and the chance of incorrect selections. The advantage of normalizing the accumulated interest is that the normalized interest values are in the range of 0 to 1 which are easy to be visualized as visual feedback. We set the intensity of words' background color to be linear with the normalized accumulated interest. The eye gazed based Bayesian target selection algorithm for 2D word targets is summarized in Algorithm 1. The parameter settings of the algorithm are described in Section 4.1.

Algorithm 1 Bayesian word selection algorithm for eye gaze

Require: Target set: $\mathbb{T} = \{t_1, t_2, \ldots, t_N\}$, the center of a target t_j is (μ_{j_x}, μ_{j_y}). Gaze trajectory: $\mathbb{S} = \{s_1, s_2, \ldots, s_K\}$ where a gaze sampling point s_i is (s_{i_x}, s_{i_y}), Threshold: θ

for s_i **in** S **do**
  for t_j **in** $\mathbb{T}$ **do**
    if $\frac{(s_{i_x}-\mu_{j_x})^2}{(\alpha * w_j)^2} + \frac{(s_{i_y}-\mu_{j_y})^2}{(\beta * h_j)^2} > 1$ **then**
      $I_i(t_j) = 0$
    else
      Obtain prior probability $P(t_j)$ using Equation (5) and compute likelihood $P(s_i|t_j)$ using Equation (3) and Equation (4);
      Compute accumulated interest $I_i(t_j)$ from Equation (1);
    end if
    Compute normalized accumulated interest I'_i using Equation (6);
    if $I'_i(t_j) > \theta$ **then**
      return t_j
    end if
  end for
end for

3.7 Voice based text correction

Once a word is selected, the user can speak the new phrase. We used Apple's Speech framework [5] for speech recognition on live audio. The speech recognition model will recognize the dictation into multiple possible transcripts, each with a confidence value. With the locations of the start and end letters of the selected word and the list of possible transcripts and their confidence values, we used the text correction algorithm (Algorithm 2) in [91] to generate possible correction candidates in sentence level. We implemented the algorithm in Swift language for iOS while it's originally proposed for Android smartphones in [91]. This algorithm used a language model, editing distance, and word embedding distance to decide the location of the new phrase in the sentence and the number of neighboring words in the sentence to be substituted. The top correction candidate sentence directly substitutes the original sentence in the text. The second and third top candidates are shown as alternative suggestions for users to select.

This algorithm offers two degrees of freedom for users:

- First, it does not require users to select the exact range of the phrase to be corrected. Users can select any word inside the range that will be replaced by the new phrase or the word directly outside that range. For example, to correct the sentence "When do yoybgade too be there" to "When do you have to be there," the user will speak "you have to." In the erroneous sentence, the user can select "yoybgade" or "too", which are words inside the phrase to be replaced, or the user can select "do" or "be," which are words directly outside the range that will be replaced by the new phrase.
- Second, the algorithm does not require users to speak the exact phrase to be corrected. Users can include context words before and after the phrase to be corrected. For example, to correct the sentence "It waspada very nice" to "It was very nice," users can select "waspada." Then, the user can speak "was," or "was" with any context words before and after "was" in the target sentence such as "It was", "was very" and "It was very nice".

4 EXPERIMENT 1: COMPARING EYESAYCORRECT WITHOUT AND WITH PRIORS

4.1 Parameter settings

We set the parameters in Equation (1) to Equation (5) as follows. The prior values γ for misspelled words were set to 2 for methods with priors and 1 for methods without priors. The σ_{j_x} and σ_{j_y} were set to be proportional to the width w_j and height h_j of the target t_j, specifically $\sigma_{j_x} = \delta * w_j$ and $\sigma_{j_y} = \delta * h_j$. δ was set to be 0.7. The correlation coefficient ρ_j is set to be 0. The parameters controlling the ellipse area for accumulating interest values were $\alpha = 3.5$, $\beta = 3.5$. The target selection threshold θ for normalized accumulated interest $I'_i(t_j)$ was $\theta = 0.999999999$. θ is extremely close to 1 because the softmax output $I'_i(t_j)$ can easily be that large when $I_i(t_j)$ is outstandingly larger than other targets' accumulated interest. The θ value was selected by experience, it's a trade-off between the target selection speed and the chance of incorrect selections.

4.2 Participants

We recruited 12 participants (2 females) from 21 to 33 years old (Mean = 26.50, Std = 3.26). The self-reported median familiarity (1: not familiar, 5: very familiar) with the eye gaze tracking interface and voice input interface was 3 and 3.5.

4.3 Apparatus

An iPad Pro device (Model launched: Oct, 2018, OS: IOS 14.5, Chipset: Apple A12X Bionic, RAM: 4GB, Internal storage: 64GB) with an 11

inch display (IPS LCD with 1668 × 2388 pixel resolution) was used for the experiment.

4.4 Design

The study was a within-subjects design. The independent variable was the text font size and the priors for misspelled words. The text font size has two levels:

- Small font size (14 points). This is close to the default footnote font size (13 points) according to the typography guidelines [7] for Apple developers.
- Large font size (28 points). This is the title font size according to the typography guidelines [7] for Apple developers.

The misspelled prior has two levels:

- With priors for misspelled words. The misspelled words are underlined with red lines, and each misspelled word has a prior value that is 2 times of correctly spelled words, namely $\gamma = 2$.
- Without priors for misspelled words. The misspelled words are underlined with red lines, and each misspelled word has the same prior value as the correctly spelled words, namely $\gamma = 1$.

In the experiment, the conditions of the two independent variables were counterbalanced across 12 users.

4.5 Tasks

Participants corrected errors in texts using eye gaze and voice. We used sentences with errors from Palin et al.'s mobile typing dataset [60] which had erroneously entered text and their correct versions by 37,370 users on mobile phones. There were 20 testing sentences for this task, 2 with omission errors, 18 with substitution errors. The edit distances between correct and erroneous sentence pairs range from 1 to 6. The edit distance reflects the difficulty of correcting an erroneous sentence. It is the minimum number of operations required to transform one sentence into the other.

Each erroneous and correct sentence pair from Palin et al.'s mobile typing dataset [60] are single sentences. In order to simulate a more realistic correction environment where there are usually sentences before and after the erroneous sentence, we randomly pasted the correct sentences before and after the erroneous sentence a few times. Those pasted sentences serve as distractions during word selection. In each trial, there were at most N sentences in the text for editing. One of them had errors. The rest were the correct versions of that sentence. N ranged from 1 to 7. It was random for each trial, each user and each configuration. The numbers of the correct sentences pasted before and after the erroneous sentence were also random for each trial, each user and each configuration under the constraint that there were N sentences in total. For each participant, the sentences for the 4 experiment conditions were generated from the same set of sentence pairs. The order of the erroneous and correct sentence pairs was randomized for different users and configurations.

In total, the experiment included 12 users × 2 font sizes × 2 prior conditions × 20 trials = 960 trials.

Figure 3: A user is correcting text on an iPad Pro using EyeSayCorrect.

4.6 Procedure

Figure 4 shows the procedure of the experiment. In each trial, a task presentation page was first displayed to show the correction task to participants as shown in Figure 4 (a). The text for editing and correction was shown on the page. The differences between the two pieces of texts were highlighted in yellow. The task was to edit the highlighted text in the text to edit to the same as the highlighted text in the target text. In the text to edit, the misspelled words detected by the spell checker would have red underlines. Those words would have higher priors during eye gaze word selection when prior was used. A user needed to clicking the "Start" button to start editing.

On the editing page as shown in Figure 4 (b), the eye gaze location of participants on the screen was shown with a red cross in real time. Participants would use their eye gaze location to select a word or button. Once the accumulated information for a word was higher than the threshold, the word would be selected. A red bounding box would show around the word to indicate it was selected. Once a word was selected, the speech recognition started. Participants could speak the new phrase when the red box around the word appeared. Once a word was selected, all the words would stop accumulate information for 5 seconds. If the participant started to speak in the 5 seconds, the words would keep not accumulating information until the participant stopped speaking and the new phrase is replaced or inserted into the text. The words would restart to accumulate information when the new phrase changed the text. There would be two other correction suggestions shown below the text. Users could select one of them by staring at the tick button at the end of that suggestion. If a word was selected, but the user did not speak in 5 seconds, the red box would disappear and all the words would restart to accumulate information.

Users can click the "Undo" button to undo the last editing. If the user forgot the editing task, he/she can choose the "Back" button to go back to the task presentation page to restart the trial. If the user successfully corrected the text for editing to the target text, a "Succeed!" label would appear, the text editing would get locked

Erroneous sentences	Correct sentences
1. What so you thinl	1. What do you think
2. Itu waspada very nice	2. It was very nice
3. The jets will rry to cintrol the ball and the clock against the Rams	3. The Jets will try to control the ball and the clock against the Rams
4. The enforcement has responsibility for the safety of the public	4. The law enforcement has responsibility for the safety of the public

Table 1: Examples of erroneous and correct sentences used in the experiment. The first 3 sentences contain substitution errors. The fourth sentence contains an omission error. The different words between the erroneous sentences and correct sentences are underlined.

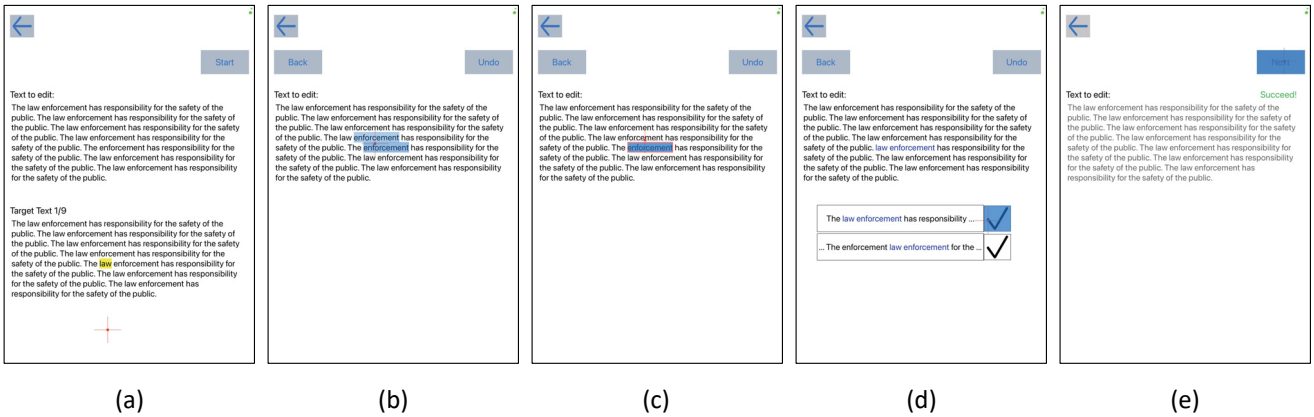

Figure 4: Procedure of experiment. (a) The page to present the correction task. The difference between Text to edit and Target text was highlighted in yellow. In this example, the user needed to insert the word 'law' into the text. (b) A user was selecting a word by eye gaze. The word's background intensity was promotional to the normalized accumulated interest for that word. (c) A word was selected. If a word was selected, it would have a red bounding box. The speech recognition started when the red bounding box appeared. (d) The default corrected text and two alternative suggestions after user finished speaking. If the default suggestion was not correct, the user could look at the tick buttons to select from alternative suggestions. (e) The page when a user successfully corrected the text, with a "Succeed" label shown. The user could select "Next" to start the next trial.

and the "Next" would appear. The user needs to choose the "Next" button to go to the next trial.

Before the experiment, participants completed a warm up session to get familiar with the interface and the procedure. They completed at least 5 trials for each of the 4 conditions. In the experiment, the participants were instructed to complete the editing of each trial (from the "Start" button was clicked to the "Succeed!" label was shown) as fast as possible. A demonstration of a participant using EyeSayCorrect on an iPad is shown in Figure 3.

4.7 Failing rate

The failing rate is the percentage of trials that were not successfully finished among all the trials. If a user cannot successfully correct an error after trying 5 times, that trial was failed. The user can skip that trial. For the large font size, the failing rate without priors and with priors were 0.83% and 0.42%. For the small font size, the failing rate without priors and with priors were both 0.00%. The failures are caused by incorrect speech recognition results or incorrect correction results and suggestions. Users have no problem to select words by eye gaze.

4.8 Task-completion time

For each trial, the task-completion time is defined as the time from the moment the "Start" button in the task presentation page was clicked to the moment the "Succeed!" label on the editing page was shown. This metric measures users' operation time to correct the errors.

The average task-completion time for all the trials in each configuration was shown in Figure 5. For large font size, the mean ± 95% CI of the task-completion time without priors and with priors were 12.82 ± 1.23 and 11.63 ± 1.07. Using priors for misspelled words reduced the task-completion time for large font size by 9.26%. A paired-samples t-test indicated that the difference was not statistically significant ($t_{11} = 1.42, p = 0.18$). For small font size, the mean ± 95% CI of the task-completion time without priors and with priors were 15.18 ± 1.96 and 11.57 ± 1.14. Using priors for

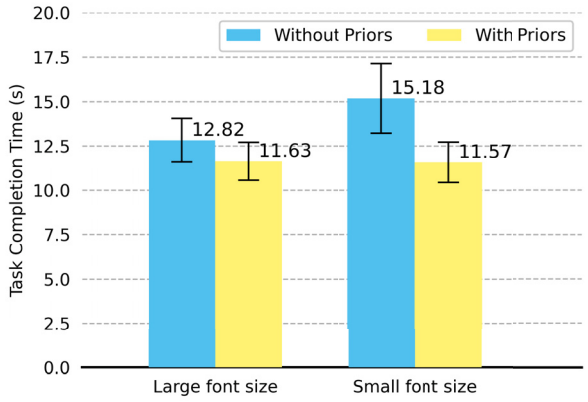

Figure 5: The mean (95% CI) of task-completion time for two font sizes and two methods.

misspelled words reduced the task-completion time for small font size by 23.79%. A paired-samples t-test indicated that the difference was statistically significant ($t_{11} = 4.07, p = 0.001$).

4.9 Text-selecting time

To investigate the effectiveness of priors for reducing the word selecting time by eye gaze, we extracted the text-selecting time from task-completion time. For each trial, the text-selecting time is the total time for selecting text by eye gaze. The text-selecting time was shown in Figure 6.

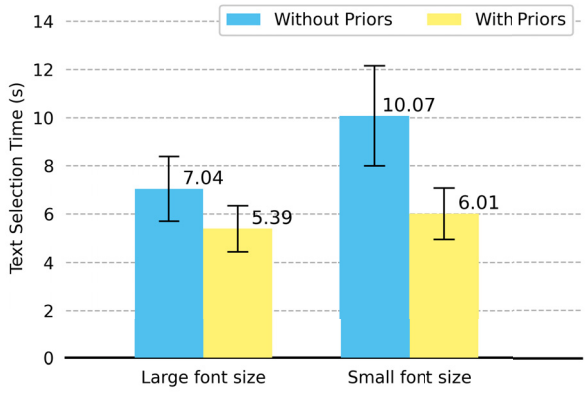

Figure 6: The mean (95% CI) of text-selecting time for two font sizes and two methods.

For large font size, the mean ± 95% CI of text-selecting time without priors and with priors were 7.04 ± 1.34 and 5.39 ± 0.95. Using priors for misspelled words reduced the text-selecting time for large font size by 23.49%. A paired-samples t-test indicated that the difference was statistically significant ($t_{11} = 2.35, p = 0.03$). For small font size, the mean ± 95% CI of text-selecting time without priors and with priors were 10.07 ± 2.08 and 6.01 ± 1.06. Using priors for misspelled words reduced the text-selecting time for small font size by 40.35%. A paired-samples t-test indicated that the difference was statistically significant ($t_{11} = 3.93, p = 0.002$).

4.10 Subjective feedback

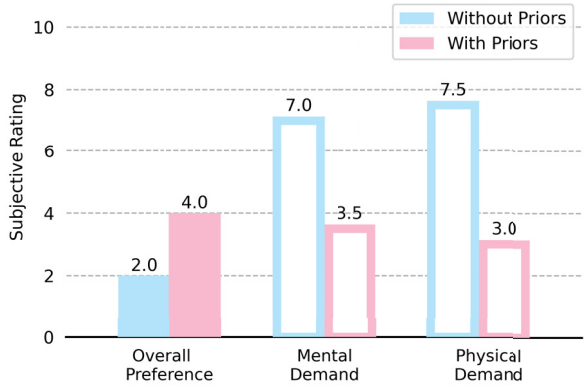

Figure 7: Medians of subjective ratings for EyeSayCorrect without and with priors. For overall preference (1: least preferred, 5: most preferred), a higher score means the method is more preferred. For mental demand and physical demand (1:least demanding, 10:most demanding), a lower rating means lower demand. EyeSayCorrect with priors received favorable ratings in all categories.

At the end of the experiment, we asked users to rate each method on a scale of 1 to 5 (1: least preferred, 5: most preferred) for conditions without and with priors for misspelled words as shown in Figure 7. The median ratings for the conditions without and with misspelled words were 2 and 4. A Wilcoxon Signed-Ranks Test indicated that the subjective ratings of condition with priors were significantly higher than that of conditions without priors ($Z = 3.06, p = 0.002$).

The participants were also asked to provide a numerical rating (1: least demanding, 10: most demanding) on mental and physical demand for conditions without and with priors for misspelled words. Mental demand describes how much mental effort is required. Physical demand describes how much physical effort is required. As shown in Figure 7, the medians of the mental demand for methods without priors and with priors were 7 and 3.5. Wilcoxon Signed-Ranks Tests indicated that the subjective mental demand of the method with priors was significantly lower than that of the method without priors ($Z = 2.80, p = 0.005$). The medians of the physical demand for methods without priors and with priors were 7.5 and 3. Wilcoxon Signed-Ranks Tests indicated that the subjective physical demand of the method with priors was significantly lower than that of the method without priors ($Z = 2.80, p = 0.005$).

5 EXPERIMENT 2: COMPARING EYESAYCORRECT WITH THE TOUCH-ONLY METHOD

Without using hands or touch, text correction on mobile devices is an extremely difficult task if not impossible. To better understand the performance of the EyeSayCorrect method comparing to the touch based method. We carried out an experiment with the touch-only method which is the common practice for text correction on mobile devices. Although the touch-only method is a unimodal method and is vastly different from the proposed *multi-modal hands-free* EyeSayCorrect method, it would serve as a benchmark for understanding the performance of EyeSayCorrect.

5.1 Experiment Design

The study was a within-subjects design. The independent variable was the correction method and the text font size. The correction method has two levels:

- Touch-only method. To select text, users can use following touch operations. 1. Tapping a word to move the cursor to the end of that word. 2. Dragging the cursor to move it at letter level. 2. Double clicking on a word to select the word. 3. First double click a word, then drag the handle at the beginning and end of the selected word to expand the selection range. To change the text, users type on the default iOS soft keyboard with QWERTY layout.
- EyeSayCorrect with priors for misspelled words. The workflow of EyeSayCorrect method is described in Section 3.1. The parameters of EyeSayCorrect method in this study are the same as the one with priors ($\gamma = 2$) in Section 4.

The text font size has two levels small (14 points) and large (28 points) which are the same as Section 4. In the experiment, the conditions of the two independent variables were counterbalanced across users.

5.2 Participants, apparatus, tasks and procedure

We recruited 12 participants (4 females, 8 males) from 23 to 33 years old (Mean = 26.00, Std = 2.86). The self-reported median familiarity (1: not familiar, 5: very familiar) with the eye gaze tracking interface, voice input interface, touch based interface was 2, 3, and 5. Four participants used single-hand typing for the touch-only method. Eight participants used two-hands typing for the touch-only method.

The same iPad Pro device as in Section 4 and the same set of tasks as in Section 4 was used for this experiment. The procedure of the experiment is the same as Section 4, the only difference is that the condition of EyeSayCorrect without priors is replaced by the touch-only method.

In total, the experiment included 12 users × 2 font sizes × 2 methods × 20 trials = 960 trials.

5.3 Failing rate

For the large font size and the small font size, the failing rates for both methods were 0% in this study.

5.4 Task-completion time

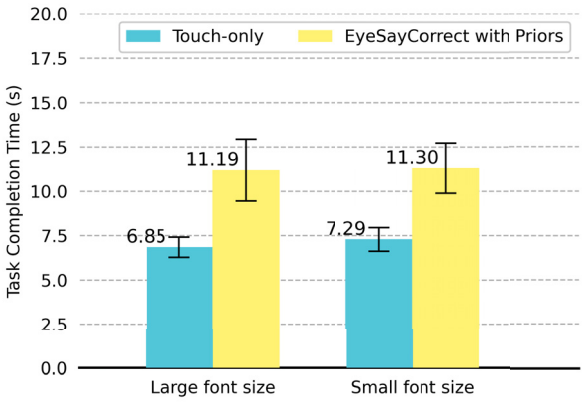

Figure 8: The mean (95% CI) of task-completion time for two font sizes and two methods.

The average task-completion time for all the trials in each configuration was shown in Figure 8. For large font size, the mean ± 95% CI of the task-completion times of touch-only method and EyeSayCorrect with priors were 6.85 ± 0.56 and 11.19 ± 1.74. The touch-only method's task-completion time is 61.23% of EyeSayCorrect method's task-completion time. A paired-samples t-test indicated that the difference was statistically significant ($t_{11} = -5.02, p = 0.00039$). For small font size, the mean ± 95% CI of the task-completion times of touch-only method and EyeSayCorrect with priors were 7.29 ± 0.66 and 11.30 ± 1.42. The touch-only method's task-completion time is 64.49% of EyeSayCorrect method's task-completion time. A paired-samples t-test indicated that the difference was statistically significant ($t_{11} = -4.13, p = 0.0016$).

5.5 Subjective feedback

At the end of the experiment, the participants rated their overall preference for each method on a scale of 1 to 5 (1: least preferred, 5: most preferred) as shown in Figure 9. The median ratings for the touch-only method and EyeSayCorrect were 4 and 3.5. A Wilcoxon Signed-Ranks Test indicated that the difference was not significant ($Z = -0.1019, p = 0.9203$).

The mental and physical demands for the two methods were also rated by participants (1: least demanding, 10: most demanding). The medians of the mental demand for the touch-only method and EyeSayCorrect were 5 and 5. Wilcoxon Signed-Ranks Tests indicated that the difference was not significant ($Z = -0.2353, p = 0.8103$). The medians of the physical demand for the touch-only method and EyeSayCorrect were 5.5 and 6.5 . Wilcoxon Signed-Ranks Tests indicated that the difference was not significant ($Z = -0.7113, p = 0.4777$).

6 GENERAL DISCUSSION

Our first experiment showed that the priors for misspelled word was effective for reducing the task-completion time and text-selecting

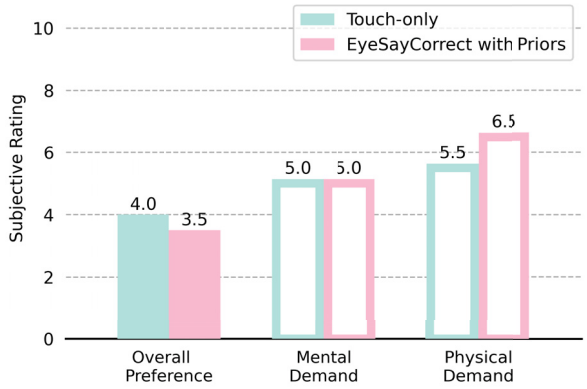

Figure 9: Medians of subjective ratings for the touch-only method and EyeSayCorrect with priors. For overall preference (1: least preferred, 5: most preferred), a higher score means the method is more preferred. For mental demand and physical demand (1:least demanding, 10:most demanding), a lower rating means lower demand.

time for text correction tasks. Using higher priors for misspelled words reduced the task-completion time by 9.26% for large font size and 23.79% for small font size, and it reduced the text-selecting time by 23.49% for large font size and 40.35% for small font size. However, this does not prove that the $\gamma = 2$ is the best value for the prior. In future work, the optimal γ value could be searched by simulation or user studies.

EyeSayCorrect cannot correct everything. In our first experiment, the failure rate was very low but not zero. The possible reasons are as follows. First, the speech recognition model may not be able to correctly transcribe users' speaking due to various reasons such as environmental noise, user's accent and model's bias. Second, the correction candidates generated by the algorithm may not be able to correct all the errors. However, users have no problem selecting their intended words by eye gaze, none of the failures were caused by inability of selecting the intended word.

The second experiment comparing EyeSayCorrect with the touch-only method showed that although hands-free text correction is extremely difficult if not impossible, EyeSayCorrect can still reach around 65% performance of the touch-only method, which makes the difficult task feasible. EyeSayCorrect does not require acquaintance with QWERTY layout which needs long time of practice to memorize. Anyone who can gaze and speak can use EyeSayCorrect. In addition, a user can correct text without knowing the exact spelling of a word such as "Wednesday".

EyeSayCorrect offers a *hands-free* approach for text correction. This is potentially useful for situations where users are not able to use hands, especially for people who have motor impairment such as quadriplegic patients and amyotrophic lateral sclerosis (ALS) patients.

EyeSayCorrect has the common limitation of voice based interface. Users may not be willing to speak out the correcting content when they are in public due to privacy concerns. In this situation, users could choose other appropriate input modalities, such as a touch based soft keyboard.

7 CONCLUSION

We proposed EyeSayCorrect, an eye gaze and voice based hands-free text correction system for mobile devices. To correct text, the user first select a word using eye gaze by a Baysian based target selection model. Then the user speaks the new phrase. EyeSayCorrect would infer the user's correction intention based on the voice inputs and the text context. Our user studies showed that using priors for misspelt words significantly reduced the task completion time and text selection time. And EyeSayCorrect made *hands-free* text correction feasible on mobile devices using gaze and voice.

ACKNOWLEDGMENTS

This work was supported in part by grants from ALS Association 20-MALS-538, NIH award R01EY030085 and NSF awards 1805076, 1936027, 2113485, 1815514. This work was done as part of the Ph.D. dissertation of Maozheng Zhao, a Stony Brook Ph.D. student supervised by Dr. Xiaojun Bi.

REFERENCES

[1] Caroline Appert and Shumin Zhai. 2009. Using strokes as command shortcuts: cognitive benefits and toolkit support. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2289–2298.
[2] Apple. 2018. About the keyboards settings on your iPhone, iPad, and iPod touch. https://support.apple.com/en-us/HT202178. [Online; accessed 22-August-2019].
[3] Apple. 2021. ARFaceAnchor. https://developer.apple.com/documentation/arkit/arfaceanchor?language=objc. [Online; Accessed: 2021-10-03].
[4] Apple. 2021. ARKit. https://developer.apple.com/documentation/arkit. [Online; Accessed: 2021-10-03].
[5] Apple. 2021. Speech Framework. https://developer.apple.com/documentation/speech. [Online; Accessed: 2021-10-03].
[6] Apple. 2021. TrueDepth Camera. https://support.apple.com/en-us/HT208108. [Online; Accessed: 2021-10-03].
[7] Apple. 2021. Typography. https://developer.apple.com/design/human-interface-guidelines/ios/visual-design/typography/. [Online; Accessed: 2021-10-03].
[8] Matthias Bernhard, Efstathios Stavrakis, Michael Hecher, and Michael Wimmer. 2014. Gaze-to-object mapping during visual search in 3d virtual environments. *ACM Transactions on Applied Perception (TAP)* 11, 3 (2014), 1–17.
[9] Xiaojun Bi, Yang Li, and Shumin Zhai. 2013. FFitts Law: Modeling Finger Touch with Fitts' Law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 1363–1372. https://doi.org/10.1145/2470654.2466180
[10] Xiaojun Bi, Tom Ouyang, and Shumin Zhai. 2014. Both Complete and Correct?: Multi-objective Optimization of Touchscreen Keyboard. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. ACM, New York, NY, USA, 2297–2306. https://doi.org/10.1145/2556288.2557414
[11] Daniel Buschek and Florian Alt. 2015. TouchML: A machine learning toolkit for modelling spatial touch targeting behaviour. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*. 110–114.
[12] Ishan Chatterjee, Robert Xiao, and Chris Harrison. 2015. Gaze+ gesture: Expressive, precise and targeted free-space interactions. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*. 131–138.
[13] Andy Cockburn, Carl Gutwin, and Saul Greenberg. 2007. A predictive model of menu performance. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 627–636.
[14] Wenzhe Cui, Suwen Zhu, Mingrui Ray Zhang, H. Andrew Schwartz, Jacob O. Wobbrock, and Xiaojun Bi. 2020. JustCorrect: Intelligent Post Hoc Text Correction Techniques on Smartphones. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 487–499. https://doi.org/10.1145/3379337.3415857
[15] Alexander De Luca, Roman Weiss, and Heiko Drewes. 2007. Evaluation of eye-gaze interaction methods for security enhanced PIN-entry. In *Proceedings of the 19th australasian conference on computer-human interaction: Entertaining user interfaces*. 199–202.

[16] A. D. N. Edwards. 2002. *Multimodal Interaction and People with Disabilities*. Springer Netherlands, Dordrecht. https://doi.org/10.1007/978-94-017-2367-1_5

[17] Stephen R Ellis and Robert J Hitchcock. 1986. The emergence of Zipf's law: Spontaneous encoding optimization by users of a command language. *IEEE transactions on systems, man, and cybernetics* 16, 3 (1986), 423–427.

[18] Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Gaze interaction for smart watches using smooth pursuit eye movements. In *Proceedings of the 28th annual ACM symposium on user interface software & technology*. 457–466.

[19] eyegaze edge. 2021. eye-tracker. https://eyegaze.com/products/eyegaze-edge/. [Online; Accessed: 2021-10-03].

[20] eyelink. 2021. eye-tracker. https://www.sr-research.com/eyelink-1000-plus/. [Online; Accessed: 2021-10-03].

[21] Anna Maria Feit, Shane Williams, Arturo Toledo, Ann Paradiso, Harish Kulkarni, Shaun Kane, and Meredith Ringel Morris. 2017. Toward everyday gaze input: Accuracy and precision of eye tracking and implications for design. In *Proceedings of the 2017 Chi conference on human factors in computing systems*. 1118–1130.

[22] Vittorio Fuccella, Poika Isokoski, and Benoit Martin. 2013. Gestures and Widgets: Performance in Text Editing on Multi-touch Capable Mobile Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. ACM, New York, NY, USA, 2785–2794. https://doi.org/10.1145/2470654.2481385

[23] Vittorio Fuccella and Benoît Martin. 2017. TouchTap: A Gestural Technique to Edit Text on Multi-Touch Capable Mobile Devices. In *Proceedings of the 12th Biannual Conference on Italian SIGCHI Chapter* (Cagliari, Italy) *(CHItaly '17)*. Association for Computing Machinery, New York, NY, USA, Article 21, 6 pages. https://doi.org/10.1145/3125571.3125579

[24] Debjyoti Ghosh, Pin Sym Foong, Shengdong Zhao, Can Liu, Nuwan Janaka, and Vinitha Erusu. 2020. Eyeditor: Towards on-the-go heads-up text editing using voice and manual input. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[25] Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. 2002. Language Modeling for Soft Keyboards. In *Proceedings of the 7th International Conference on Intelligent User Interfaces* (San Francisco, California, USA) *(IUI '02)*. ACM, New York, NY, USA, 194–195. https://doi.org/10.1145/502716.502753

[26] Google. 2021. Get started with Voice Access. https://support.google.com/accessibility/android/answer/6151848?hl=en. [Online; Accessed: 2021-07-18].

[27] google.com. 2021. Type with your voice. https://support.google.com/docs/answer/4492226?hl=en#zippy=%2Cselect-text. [Online; accessed 6-April-2021].

[28] Christian Holz and Patrick Baudisch. 2011. Understanding Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 2501–2510. https://doi.org/10.1145/1978942.1979308

[29] Michael Xuelin Huang, Jiajia Li, Grace Ngai, and Hong Va Leong. 2017. Screenglint: Practical, in-situ gaze estimation on smartphones. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2546–2557.

[30] iMore.com. 2021. Everything you can do with Voice Control on iPhone and iPad. https://www.imore.com/everything-you-can-do-voice-control-iphone-and-ipad. [Online; Accessed: 2021-07-18].

[31] ExIdeas Inc. 2018. MessagEase - The Smartest Touch Screen keyboard. https://www.exideas.com/ME/index.php. [Online; accessed 22-August-2019].

[32] Grammarly Inc. 2020. Grammarly Keyboard. https://en.wikipedia.org/wiki/Grammarly [Online; accessed May-2020].

[33] Jalal Ismaili et al. 2017. Mobile learning as alternative to assistive technology devices for special needs students. *Education and Information Technologies* 22, 3 (2017), 883–899.

[34] Poika Isokoski, Benoît Martin, Paul Gandouly, and Thomas Stephanov. 2010. Motor Efficiency of Text Entry in a Combination of a Soft Keyboard and Unistrokes. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries* (Reykjavik, Iceland) *(NordiCHI '10)*. ACM, New York, NY, USA, 683–686. https://doi.org/10.1145/1868914.1869004

[35] Toshiya Isomoto, Toshiyuki Ando, Buntarou Shizuki, and Shin Takahashi. 2018. Dwell Time Reduction Technique Using Fitts' Law for Gaze-Based Target Acquisition. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications* (Warsaw, Poland) *(ETRA '18)*. Association for Computing Machinery, New York, NY, USA, Article 26, 7 pages. https://doi.org/10.1145/3204493.3204532

[36] Howell Istance, Aulikki Hyrskykari, Lauri Immonen, Santtu Mansikkamaa, and Stephen Vickers. 2010. Designing gaze gestures for gaming: an investigation of performance. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*. 323–330.

[37] Robert JK Jacob. 1991. The use of eye movements in human-computer interaction techniques: what you look at is what you get. *ACM Transactions on Information Systems (TOIS)* 9, 2 (1991), 152–169.

[38] Liu Jigang, Bu Sung Lee Francis, and Deepu Rajan. 2019. Free-head appearance-based eye gaze estimation on mobile devices. In *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, 232–237.

[39] Andreas Komninos, Mark Dunlop, Kyriakos Katsaris, and John Garofalakis. 2018. A Glimpse of Mobile Text Entry Errors and Corrective Behaviour in the Wild. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct* (Barcelona, Spain) *(MobileHCI '18)*. ACM, New York, NY, USA, 221–228. https://doi.org/10.1145/3236112.3236143

[40] N. Krahnstoever, S. Kettebekov, M. Yeasin, and R. Sharma. 2002. A Real-Time Framework for Natural Multimodal Interaction with Large Screen Displays. In *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces (ICMI '02)*. IEEE Computer Society, USA, 349. https://doi.org/10.1109/ICMI.2002.1167020

[41] Chandan Kumar, Ramin Hedeshy, I Scott MacKenzie, and Steffen Staab. 2020. TAGSwipe: Touch Assisted Gaze Swipe for Text Entry. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.

[42] Manu Kumar, Andreas Paepcke, and Terry Winograd. 2007. EyePoint: practical pointing and selection using gaze and keyboard. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 421–430.

[43] Toby Jia-Jun Li, Jingya Chen, Haijun Xia, Tom M. Mitchell, and Brad A. Myers. 2020. Multi-Modal Repairs of Conversational Breakdowns in Task-Oriented Dialogs. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 1094–1107. https://doi.org/10.1145/3379337.3415820

[44] Zhi Li, Maozheng Zhao, Yifan Wang, Sina Rashidian, Furqan Baig, Rui Liu, Wanyu Liu, Michel Beaudouin-Lafon, Brooke Ellison, Fusheng Wang, et al. 2021. BayesGaze: A Bayesian Approach to Eye-Gaze Based Target Selection. In *Graphics Interface 2021*.

[45] Wanyu Liu, Gilles Bailly, and Andrew Howes. 2017. Effects of frequency distribution on linear menu performance. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 1307–1312.

[46] Google LLC. 2020. Gboard. https://en.wikipedia.org/wiki/Gboard [Online; accessed May-2020].

[47] Christof Lutteroth, Moiz Penkar, and Gerald Weber. 2015. Gaze vs. mouse: A fast and accurate gaze-only click alternative. In *Proceedings of the 28th annual ACM symposium on user interface software & technology*. 385–394.

[48] Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. 2009. Fast gaze typing with an adjustable dwell time. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 357–360.

[49] Jennifer Mankoff, Gregory D Abowd, and Scott E Hudson. 2000. OOPS: a toolkit supporting mediation techniques for resolving ambiguity in recognition-based interfaces. *Computers & Graphics* 24, 6 (2000), 819–834. https://doi.org/10.1016/S0097-8493(00)00085-6 Calligraphic Interfaces: towards a new generation of interactive systems.

[50] Julio C Mateo, Javier San Agustin, and John Paulin Hansen. 2008. Gaze beats mouse: hands-free selection by combining gaze and emg. In *CHI'08 extended abstracts on Human factors in computing systems*. 3039–3044.

[51] Darius Miniotas, Oleg Špakov, and I Scott MacKenzie. 2004. Eye gaze interaction with expanding targets. In *CHI'04 extended abstracts on Human factors in computing systems*. 1255–1258.

[52] Alistair Morrison, Xiaoyu Xiong, Matthew Higgs, Marek Bell, and Matthew Chalmers. 2018. A Large-Scale Study of iPhone App Launch Behaviour. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–13.

[53] Martez E Mott, Shane Williams, Jacob O Wobbrock, and Meredith Ringel Morris. 2017. Improving dwell-based gaze typing with dynamic, cascading dwell times. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2558–2570.

[54] Aanand Nayyar, Utkarsh Dwivedi, Karan Ahuja, Nitendra Rajput, Seema Nagar, and Kuntal Dey. 2017. OptiDwell: intelligent adjustment of dwell click time. In *Proceedings of the 22nd international conference on intelligent user interfaces*. 193–204.

[55] nuance.com. 2021. Dragon Speech Recognition - Get More Done by Voice: Dragon. https://www.nuance.com/dragon.html. [Online; accessed 6-April-2021].

[56] Per Ola Kristensson and Keith Vertanen. 2011. Asynchronous Multimodal Text Entry Using Speech and Gesture Keyboards.. In *Proceedings of the International Conference on Spoken Language Processing* (Florence, Italy). 581–584.

[57] Sharon Oviatt and Philip Cohen. 2000. Perceptual User Interfaces: Multimodal Interfaces That Process What Comes Naturally. *Commun. ACM* 43, 3 (March 2000), 45–53. https://doi.org/10.1145/330534.330538

[58] Sharon Oviatt, Phil Cohen, Lizhong Wu, John Vergo, Lisbeth Duncan, Bernhard Suhm, Josh Bers, Thomas Holzman, Terry Winograd, James Landay, Jim Larson, and David Ferro. 2000. Designing the User Interface for Multimodal Speech and Pen-Based Gesture Applications: State-of-the-Art Systems and Future Research Directions. *Hum.-Comput. Interact.* 15, 4 (Dec. 2000), 263–322. https://doi.org/10.1207/S15327051HCI1504_1

[59] Sharon Oviatt and Philip R. Cohen. 2015. *The Paradigm Shift to Multimodality in Contemporary Computer Interfaces*. Morgan & Claypool Publishers.

[60] Kseniia Palin, Anna Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How do People Type on Mobile Devices? Observations from a Study with

37,000 Volunteers.. In *Proceedings of 21st International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI'19)*. ACM.

[61] Alexandra Papoutsaki, Patsorn Sangkloy, James Laskey, Nediyana Daskalova, Jeff Huang, and James Hays. 2016. WebGazer: Scalable Webcam Eye Tracking Using User Interactions. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*. AAAI, 3839–3845.

[62] Mohsen Parisay, Charalambos Poullis, and Marta Kersten-Oertel. 2020. FELiX: Fixation-based Eye Fatigue Load Index A Multi-factor Measure for Gaze-based Interactions. In *2020 13th International Conference on Human System Interaction (HSI)*. IEEE, 74–81.

[63] Jimin Pi and Bertram E Shi. 2017. Probabilistic adjustment of dwell time for eye typing. In *2017 10th International Conference on Human System Interactions (HSI)*. IEEE, 251–257.

[64] prc accent. 2021. eye-tracker. https://www.prentrom.com/products/devices. [Online; Accessed: 2021-10-03].

[65] Kari-Jouko Räihä and Saila Ovaska. 2012. An exploratory study of eye typing fundamentals: dwell time, text entry rate, errors, and workload. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 3001–3010.

[66] Radiah Rivu, Yasmeen Abdrabou, Ken Pfeuffer, Mariam Hassib, and Florian Alt. 2020. Gaze'N'Touch: Enhancing Text Selection on Mobile Devices Using Gaze. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3334480.3382802

[67] David Rozado, T Moreno, J San Agustin, FB Rodriguez, and Pablo Varona. 2015. Controlling a smartphone using gaze gestures as the input mechanism. *Human–Computer Interaction* 30, 1 (2015), 34–63.

[68] Ritam Jyoti Sarmah, Yunpeng Ding, Di Wang, Cheuk Yin Phipson Lee, Toby Jia-Jun Li, and Xiang 'Anthony' Chen. 2020. Geno: A Developer Tool for Authoring Multimodal Interaction on Existing Web Applications. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 1169–1181. https://doi.org/10.1145/3379337.3415848

[69] Immo Schuetz, T Scott Murdison, Kevin J MacKenzie, and Marina Zannoli. 2019. An Explanation of Fitts' Law-like Performance in Gaze-Based Selection Tasks Using a Psychophysics Approach. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.

[70] Korok Sengupta, Sabin Bhattarai, Sayan Sarcar, I Scott MacKenzie, and Steffen Staab. 2020. Leveraging Error Correction in Voice-based Text Entry by Talk-and-Gaze. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–11.

[71] Ludwig Sidenmark and Hans Gellersen. 2019. Eye&head: Synergetic eye and head movement for gaze pointing and selection. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 1161–1174.

[72] Khe Chai Sim. 2010. Haptic Voice Recognition: Augmenting speech modality with touch events for efficient speech recognition. In *2010 IEEE Spoken Language Technology Workshop*. 73–78. https://doi.org/10.1109/SLT.2010.5700825

[73] Khe Chai Sim. 2012. Speak-as-you-swipe (SAYS): A Multimodal Interface Combining Speech and Gesture Keyboard Synchronously for Continuous Mobile Text Entry. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction* (Santa Monica, California, USA) *(ICMI '12)*. ACM, New York, NY, USA, 555–560. https://doi.org/10.1145/2388676.2388793

[74] Adalberto L Simeone, Andreas Bulling, Jason Alexander, and Hans Gellersen. 2016. Three-point interaction: Combining bi-manual direct touch with gaze. In *Proceedings of the international working conference on advanced visual interfaces*. 168–175.

[75] Shyamli Sindhwani, Christof Lutteroth, and Gerald Weber. 2019. ReType: Quick Text Editing with Keyboard and Gaze. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. ACM, New York, NY, USA, Article 203, 13 pages. https://doi.org/10.1145/3290605.3300433

[76] Henrik Skovsgaard, Julio C Mateo, John M Flach, and John Paulin Hansen. 2010. Small-target selection with gaze alone. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*. 145–148.

[77] SMI-REDn. 2021. eye-tracker. https://imotions.com/hardware/smi-redn-scientific/. [Online; Accessed: 2021-10-03].

[78] Sophie Stellmach and Raimund Dachselt. 2012. Look & touch: gaze-supported target acquisition. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 2981–2990.

[79] Bernhard Suhm, Brad Myers, and Alex Waibel. 2001. Multimodal error correction for speech user interfaces. *ACM transactions on computer-human interaction (TOCHI)* 8, 1 (2001), 60–98.

[80] tobii. 2021. eye-tracker. https://gaming.tobii.com/product/eye-tracker-5/, https://www.tobiipro.com/, https://www.tobiidynavox.com/. [Online; Accessed: 2021-10-03].

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

[82] Daniel Vogel and Patrick Baudisch. 2007. Shift: A Technique for Operating Pen-Based Interfaces Using Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 657–666. https://doi.org/10.1145/1240624.1240727

[83] Klaus Weidner. 2018. Hackers Keyboard. http://code.google.com/p/hackerskeyboard/ [Online; accessed 22-August-2019].

[84] Daryl Weir, Simon Rogers, Roderick Murray-Smith, and Markus Löchtefeld. 2012. A user-specific machine learning approach for improving touch accuracy on mobile devices. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. 465–476.

[85] Erroll Wood, Tadas Baltrusaitis, Xucong Zhang, Yusuke Sugano, Peter Robinson, and Andreas Bulling. 2015. Rendering of eyes for eye-shape registration and gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision*. 3756–3764.

[86] Erroll Wood and Andreas Bulling. 2014. Eyetab: Model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*. 207–210.

[87] Mingrui Ray Zhang, He Wen, and Jacob O. Wobbrock. 2019. Type, Then Correct: Intelligent Text Correction Techniques for Mobile Text Entry Using Neural Networks. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 843–855. https://doi.org/10.1145/3332165.3347924

[88] Mingrui Ray Zhang and O. Jacob Wobbrock. 2020. Gedit: Keyboard gestures for mobile text editing. In *Proceedings of Graphics Interface (GI '20)* (Toronto, Ontario) *(GI '20)*. Canadian Information Processing Society, Toronto, Ontario, 97–104.

[89] Xucong Zhang, Michael Xuelin Huang, Yusuke Sugano, and Andreas Bulling. 2018. Training person-specific gaze estimators from user interactions with multiple devices. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 1–12.

[90] Xinyong Zhang, Xiangshi Ren, and Hongbin Zha. 2010. Modeling dwell-based eye pointing target acquisition. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2083–2092.

[91] Maozheng Zhao, Wenzhe Cui, I.V. Ramakrishnan, Shumin Zhai, and Xiaojun Bi. 2021. Voice and Touch Based Error-tolerant Multimodal Text Editing and Correction for Smartphones. In *The 34th Annual ACM Symposium on UserInterface Software and Technology (UIST '21), October 10–14, 2021, VirtualEvent, USA*. https://doi.org/10.1145/3472749.3474742

[92] Xiaolong Zhou, Haibin Cai, Zhanpeng Shao, Hui Yu, and Honghai Liu. 2016. 3D eye model-based gaze estimation from a depth sensor. In *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. 369–374. https://doi.org/10.1109/ROBIO.2016.7866350

[93] Suwen Zhu, Yoonsang Kim, Jingjie Zheng, Jennifer Yi Luo, Ryan Qin, Liuping Wang, Xiangmin Fan, Feng Tian, and Xiaojun Bi. 2020. Using Bayes' Theorem for Command Input: Principle, Models, and Applications. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–15.