EyeTell: Tablet-based Calibration-free Eye-typing using Smooth-pursuit movements

Bafna, Tanya; Bækgaard, Per; Paulin Hansen, John Paulin

Published in:
Proceedings of the ETRA '21 Short Papers

Link to article, DOI:
10.1145/3448018.3458015

Publication date:
2021

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Bafna, T., Bækgaard, P., & Paulin Hansen, J. P. (2021). EyeTell: Tablet-based Calibration-free Eye-typing using Smooth-pursuit movements. In *Proceedings of the ETRA '21 Short Papers* Article 24 ACM. https://doi.org/10.1145/3448018.3458015

EyeTell: Tablet-based Calibration-free Eye-typing using Smooth-pursuit movements

Tanya Bafna
Department of Technology,
Management and Economics
Kongens Lyngby, Denmark
taba@dtu.dk

Per Bækgaard
Department of Applied Mathematics
and Computer Science
Kongens Lyngby, Denmark
pgba@dtu.dk

John Paulin Hansen
Department of Technology,
Management and Economics
Kongens Lyngby, Denmark
jpha@dtu.dk

ABSTRACT

Gaze tracking technology, with the increasingly robust and lightweight equipment, can have tremendous applications. To use the technology during short interactions, such as in public displays or hospitals to communicate non-verbally after a surgery, the application needs to be intuitive without requiring a calibration. Gaze gestures such as smooth-pursuit eye movements can be detected without calibration. We report the working performance of a calibration-free eye-typing application using only the front-facing camera of a tablet. In a user study with 29 participants, we obtained an average typing speed of 1.27 WPM after four trials and a maximum typing speed of 1.95 WPM.

CCS CONCEPTS

• **Human-centered computing** → *Keyboards*; **Gestural input**; *Text input*; Auditory feedback; **Accessibility technologies**; *Tablet computers.*

KEYWORDS

text entry, eye-tracking, smooth-pursuit, calibration-free, tablet, mobile

ACM Reference Format:
Tanya Bafna, Per Bækgaard, and John Paulin Hansen. 2021. EyeTell: Tablet-based Calibration-free Eye-typing using Smooth-pursuit movements. In *ETRA '21: 2021 Symposium on Eye Tracking Research and Applications (ETRA '21 Short Papers), May 25–27, 2021, Virtual Event, Germany.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3448018.3458015

1 INTRODUCTION

Eye-tracking technology has found uses in marketing, medicine, gaming and usability. Tracking of eye-movements also allows people with severe disability to communicate, via on-screen keyboards and eye-trackers [Ball et al. 2010]. Eye movements have been harnessed for applications in eye-typing [Lenglet et al. 2019; Ward et al. 2000], adaptive speed control of videos [Abibouraguimane et al. 2019], password authentication [Cymek et al. 2014], expertise development [Castner et al. 2017], and other tasks that allow for hands-free uses of interfaces. This is advantageous in everyday situations, where an ongoing task requires interactions with hands, such as surgery [Abibouraguimane et al. 2019] or hands-free applications provide a hygienic option of interaction in the public spaces [Khamis et al. 2016]. Hygienic considerations for public displays have become crucial since the outbreak of the COVID-19 pandemic. Use of eye interactive interfaces in these situations could be improved if they could be accessed without requiring calibration of the eye-tracker. We report the working and performance of EyeTell, a calibration-free eye-typing application, designed for an iPad, based on smooth-pursuit eye movements.

Text entry applications with gaze tracking can be divided into two major categories – using dwell-time selection and using gaze gestures. Dwell-time selection consists of fixating on a key for a certain amount of time, named dwell-time. This technique can have several disadvantages as the time required to select each character can be about 700 ms [Hansen et al. 2001] and the application requires a high accuracy to correctly identify the key fixated on, from at least 26 keys on an English keyboard [Wobbrock and Myers 2006]. Selection using gaze gestures includes applications that require generating gestures using the eyes that resemble the letter in a keyboard called EyeWrite [Wobbrock et al. 2008], steering of the gaze towards the intended letter from a line of letters in a keyboard known as Dasher [Ward et al. 2000], or generating eye-movements, known as the smooth-pursuit eye-movements, that are similar to the moving target letter, in a keyboard called SMOOVS [Lutz et al. 2015]. EyeTell uses the concept of smooth-pursuit eye movements in an eye-typing application.

Smooth-pursuit eye movements have been used for text entry with various applications – answering multiple-choice questions on public displays by following the trajectory of the moving textual answers [Khamis et al. 2016], entering the pin-code on an on-screen pin-pad with each number having a unique trajectory [Cymek et al. 2014]. Lutz et al. have developed an application that allows entry of individual letters from the alphabet, by dividing a letter entry into two phases of interaction – selection of a cluster of letters, which contains the intended letter and the selection of which allows access to the individual letters, followed by the selection of the intended letter [Lutz et al. 2015]. In order to reduce the time required for calibration, they have used a one-point calibration for the keyboard.

The most common method of evaluation of a text entry application is typing speed, provided as the number of words typed per minute (WPM). Some of the highest typing speed has been obtained using dwell-time selection keyboards – 19.89 WPM using adjustable dwell-time after ten 15-minute typing sessions [Majaranta et al. 2009]. Among the gesture-based keyboards, the highest typing

ETRA '21 Short Papers, May 25–27, 2021, Virtual Event, Germany
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8345-5/21/05…$15.00
https://doi.org/10.1145/3448018.3458015

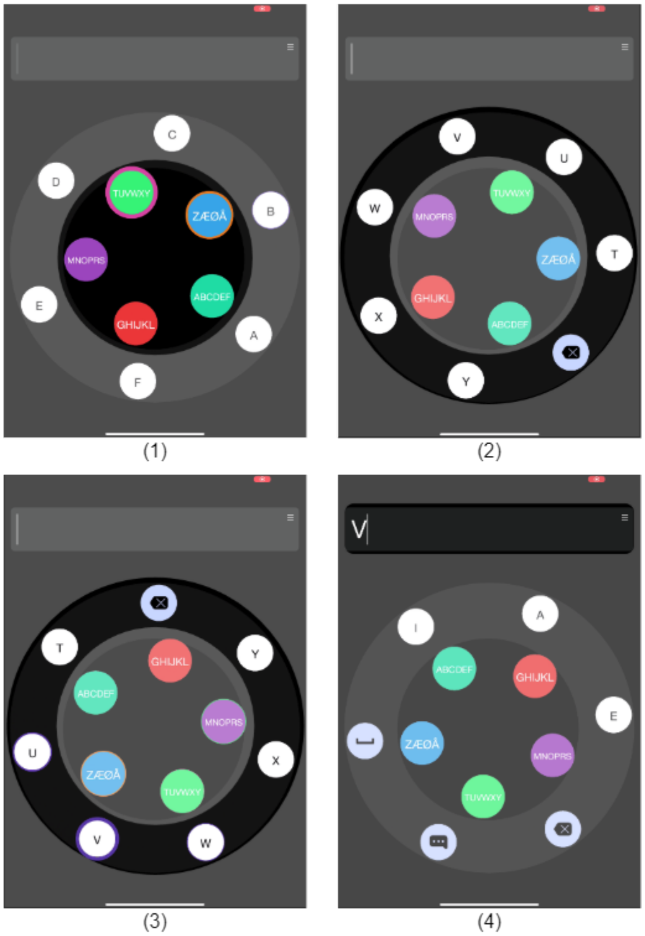


Figure 1: Selection of a letter using the EyeTell application. Events from the highlighted sections are described further. (1) The pink border around the target TUVWXY (green-colored target) and the orange border around the target ZÆØÅ (blue-colored target) in the inner circle indicate that these have the highest probability of being selected. (2) The target TUVWXY is ultimately selected, which results in the outer circle generating a target for each of the individual letters, and a target to delete one character. (3) Letter V from the outer circle is in the process of being selected, denoted by the purple border around the target. (4) When the letter V is selected, it appears in the space above the keyboard. At this point, the outer circle consists of three most-likely letters to follow the letter selected, and targets to select a space, read the typed text aloud and delete a character.

speed of 34 WPM was obtained for Dasher by an expert user [Ward et al. 2000]. Keyboards using smooth-pursuit eye movements have had a much lower typing speed, with 3.34 WPM by SMOOVS [Lutz et al. 2015], which was improved to 4.5 WPM using a language model [Zeng and Roetting 2018].

Calibration-free text entry is one of the potential outcomes of using smooth-pursuit eye movements. The smooth-pursuit keyboard SMOOVS uses a one-point calibration to get sufficient accuracy, and is thus almost calibration free. Other works have used a third-person calibration, where the eye-tracking system is calibrated by one user, and can be used directly by other users, "calibration free" [Abdrabou et al. 2019; Cymek et al. 2014; Drewes et al. 2019]. The EyeTell application examined in this paper, is the first smooth-pursuit based eye-typing application developed using only an iPad and the existing front-facing camera and it does not require any kind of calibration before use. The novel concept used here is the use of open-source services offered by Apple to implement smooth-pursuit based text entry. We envision the application finding use in public spaces for hands-free short text input, specially in the times of COVID-19, and for people with disabilities using the iPad, attached to a mobility device, during traveling or whenever in need of giving an instant short message, for instance at the bathroom or laying in bed. A text-entry application on a small screen could also find other uses, such as in public displays for pin entry, which would preferably be smaller than stationary computer screens, to provide privacy to the user.

We will examine the performance, including the strengths and limitations, of an eye-tracking system developed using only a front camera of an iPad.

2 EYETELL

In this section, we describe the calibration-free smooth-pursuit eye movements based application – EyeTell, developed in a start-up known as Obital[1]. It is designed for eye-typing on a mobile device such as a tablet. The application requires no calibration, and works by estimating the optical axis provided by the ARKit from Apple in a normalized world space[2]. After coordinate transformations to offset the device and the head position, the gaze point relative to the camera position is derived from estimates of the position and orientation information of each eye. The normalized distance between the trajectory of the gaze point and the path of every target shown on the screen is computed and subtracted from 1, generating an estimate of the confidence of the gaze following the respective target. A target which has a confidence value greater than 0.9 for at least 2 s is selected.

The application consists of two concentric circles of targets moving with an optimized angular speed of 48°/s, which was empirically determined. The inner circles contains five equiangular targets moving in a counter-clockwise circular motion. Each target in the inner circle represents a multi-letter cluster – ABCDEF, GHIJKL, MNOPRS, TUVWXY and ZÆØÅ. The targets in the outer circle perform a clockwise circular motion. The number of targets present in the outer circle depends on the previous selection:

(1) The outer circle has five targets: When the previous selection was the multi-letter target ZÆØÅ from the inner circle, it results in five targets in the outer circle – the four single-letters from ZÆØÅ and a target for character deletion.
(2) The outer circle consists of seven targets: When the previous selection was any other multi-letter target from the inner circle, it generates seven targets in the outer circle – the six

[1] Now acquired by GN Group – https://www.linkedin.com/company/obital
[2] https://developer.apple.com/augmented-reality/arkit/

single-letters from the multi-letter target and a target for character deletion, as observed in Figure 1(2).

(3) The outer circle has six targets: When the previous selection was a target from the outer circle, it results in the following six targets in the outer circle – three single-letter targets, a target to generate space, a target allowing the typed text to be spoken aloud and a character deletion target, as observed in Figure 1(4). The three single-letter targets were the most likely letters to be selected after the current selection. They could be replaced by words, generated by the word prediction model, however, for the experiment, word prediction was deactivated.

Visual feedback in form of a border around the target with the highest correlation is used to convey to the user, the target most likely to be selected. The border of the target in the inner circle has a different and contrasting color than the target itself, as observed in Figure 1(1). The border is purple-colored for the targets in the outer circle, as seen in Figure 1(3). On selection of a target, audio feedback in form of a click sound is provided to the user.

To enable use as a communication medium for people with disabilities, a read-aloud key uses the text-to-speech service to read the typed text aloud. Selection of this key also stores a log file, on the tablet, with the typing events of selections using the application.

3 METHODS

3.1 Participants

Thirty-six danish-speaking participants volunteered for the experiment (18 males; 18 females, age range: 18 – 47 years). The participants, composed of university students and employees, reported of no history of photosensitive epileptic seizures, but one participant reported having dyslexia. Due to data loss (five participants), not having completing the experiment (one participant) or dyslexia (one participant), data from 29 participants (15 males; 14 females) is reported.

3.2 Experiment Design

Each participant performed two experimental sessions. A session consisted of two trials, with the goal of observing short interactions with the application. The first session had an additional practice trial before the two trials. Each trial involved memorizing four words and typing them using EyeTell. The trial was classified as easy or difficult, depending on whether the words were easy (more common words) or difficult (less frequently used words). The four words in the easy and difficult trials consisted of a total of 38 characters. Each session was composed of an easy trial and a difficult trial, and each participant performed two sessions. The order of the easy and difficult trials was balanced over the two sessions for each participant. Two trials were used to minimize the interaction time with the application, and estimate the ease of use in various situations (easy, difficult).

At the end of each session, the participants answered questions from the NASA-Task Load Index (NASA-TLX), on a scale of 1 to 7.

The gaze and typing data was logged at a frequency of 30 Hz by the application, and saved locally on the iPad, when the *read-aloud* key was selected. The log file contained the estimated gaze points, the path followed by every target and the event of key selection. Due to the General Data Protection Regulations on data collected by a company, the data on key selected was restricted to the type of target selected – cluster of letters or a letter, space key or the delete key.

To record additional gaze data during the experiment, wearable eye-tracking glasses (Tobii Pro Glasses 2; sampling frequency: 100 Hz) were used to obtain the gaze points, in addition to the gaze points estimated by the application. The gaze data was accessed using a licensed version of the Tobii Pro Lab software, which also allowed video mapping of the camera video on a representative camera snapshot. It included gaze points mapped on the scene camera snapshot, given in pixels. The participants remained in approximately the same position for the trial, and a mapping step was used to compensate for the slight head movement.

The data from EyeTell was upsampled to 100 Hz using linear interpolation. To examine the performance of the application, the gaze points data obtained from the Tobii eye-tracking glasses were compared to the EyeTell target path data, logged by the application. The target path was obtained in arbitrary units (53 arbitrary units equaled 1 cm). The two data sets, recorded in different coordinate system, were made comparable by scaling into centimeters using size measurements from the iPad, computed using a physical scale.

3.3 Data Analysis

To examine the performance of the EyeTell application, we have computed a series of performance metrics – typing speed, error rate, search time, selection time, trial duration and the information transfer rate.

Selection of the targets was designed so as to require 2 s of gaze points correlating with the target path, but we decided to compute the actual *selection time* recorded, to evaluate the application. This was the time that the gaze points, obtained using the head-mounted eye-tracker, followed the target path, before the selection occurred.

The layout of EyeTell is one of the first with two concentric circles with targets moving in opposite directions. While this could serve an advantage with increased number of possible selections, it could also confuse the user. To evaluate the application, the *search time* was computed, which was the time required by the participant to find the next intended target. *Trial duration* was computed as the time required for a trial, in minutes.

Typing speed was calculated as the number of words typed per minute, where one word is defined as five characters, including space. This metric has the units words per minute (WPM).

We classify errors into the errors made by the system, defined as *false detection errors* and the errors corrected by the user, defined as *corrected error rate*. The false detection errors count the number of times the application selected a target not intended by the participants, by detecting a mismatch between the selected target path and the gaze points. The false detection error rate was defined as the ratio of the number of unintended selections to the total number of selections. The corrected error rate is computed as the number of backspace selections, compared to the total number of selections. Note that this metric could include corrections made by the participants in response to a false detection by the application, when the false detection resulted in a character selection. However, if a false detection occurred for a target in the inner circle, which consisted of the cluster of letters, it would not require deletion, and the error could be corrected by simply selecting the correct cluster of letters.

We also computed the *information transfer rate (ITR)* [Obermaier et al. 2001; Sadeghi and Maleki 2019; Wolpaw et al. 2002] in bits/minute as an approximate evaluation metric of the information conveyed through the system. The *ITR* can, for our purposes and with known limitations [Speier et al. 2013], naively be defined as:

$$B = log_2 N + P \times log_2 P + (1 - P) \times log_2 \left(\frac{1 - P}{N - 1} \right)$$

$$ITR = B \times \left(\frac{S_{total}}{T} \right)$$

where S_{total} is the total number of selections in the trial, T is the total trial time in minutes, B is the information transferred per selection in bits, N is the number of possible targets per selection (here $N = 30$, comprised of 28 letters, a space key and a delete key), and P is the selection accuracy. We define an upper bound on P as:

$$P \leq \frac{S_{total} - S_{delete} - S_{cerrors}}{S_{total}}$$

$$S_{cerrors} = \sum_{words} \left| C_{target} - C_{typed} \right|$$

where S_{total} again is the total number of selections, including deletions, S_{delete} is the number of deletions[3], and $S_{cerrors}$ is an estimate of the remaining uncorrected errors in what was typed by the participant, calculated by summing the absolute difference between the number of resulting, C_{typed}, and corresponding target, C_{target}, characters in each word[4].

Finally, the *subjective assessment* on the ease of use of the application was provided by the participants using the NASA-TLX questionnaire.

4 RESULTS

We report on the performance of the application and the participants during their short interaction times of two trials, repeated twice for each participant, in terms of the mean and standard error (SE).

The total time needed to select a target is depicted in Figure 2. The mean search time, or the time required to find the intended target, for all the participants was 5.22 s (SE=0.08), with the minimum time required to find the target being 3.78 s for a participant. The mean selection time, was 3.45 s (SE=0.07), and the minimum average selection time computed for a participant was 1.99 s, which was also the minimum time required by the application to allow a selection. The average trial duration was 7.53 minutes (SE=0.32).

The average typing speed obtained by the application was 1.27 WPM (SE=0.03), which reached a maximum of 1.95 WPM for one participant. The typing speed over the trials of the two sessions is depicted in Figure 3a.

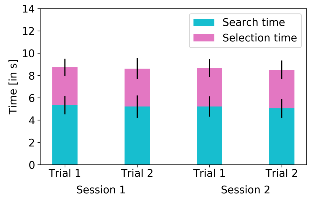
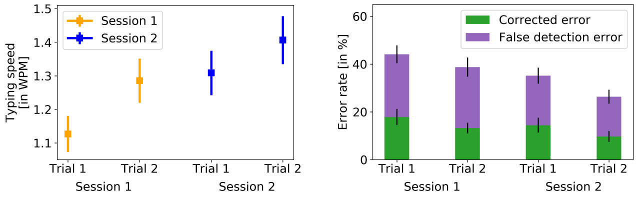


Figure 2: The time required to find the target, or search time (blue) and select the target, or selection time (pink). The error bars depict the standard error.



(a) Typing speed

(b) Error rate

Figure 3: Plots of (a) typing speed and (b) the error rate, with the false detection error by the application and the errors corrected by the participants. The error bars depict the standard error.

The error rate was classified into the false detection errors, caused by the application recognizing a selection other than the intended selection by the participant, and the corrected errors, which were corrected by the participants. The mean false detection error rate was 23.2% (SE=1.3), computed over the trials and sessions for all the participants. The minimum false detection error rate obtained for a participant was 5.0%, indicating that the false detection errors, although made by the application, could depend on the participant. Approximately 75% of the false detection error occurred in the outer circle, while the rest occurred in the inner circle. The corrected error rate had a mean of 14.1% (SE=0.9) for the participants, calculated for both the trials and sessions. The corrected errors, as seen in Figure 3b, were less than the false detection errors. This is because the false detection errors generated on targets from the inner circle of EyeTell did not require a delete key selection to be corrected, but could be corrected by selecting the correct target of cluster of letters.

The average calculated ITR of the keyboard was 33.6 bits/min (SE=1.4), and a maximum of 58.1 bits/min was achieved for one participant; these are upper bounds on the true ITR. Since the location of the targets was not stationary, the time used to complete a target selection included the selection time as well as the search time.

The subjective evaluation of the ease of using the keyboard was done using the NASA-TLX scores at the end of each session, which is depicted in Figure 4. The average NASA-TLX score, from a total score of 42, was 17.8 (SE=0.6) at the end of session 1, which reduced to 15.2 (SE=0.6) after session 2.

[3] Whenever the user selects the delete target, we assume an error has been made. Either the previous character 'X' was wrong, and is being replaced by a new character 'Y' as in <'X', DEL, 'Y'>, or alternatively the user selects the delete target incorrrectly, as in <'X', DEL, 'X>. In both cases, two correct selections have been made and one incorrect.

[4] We cannot account for swapped or incorrectly typed characters since the interface does not log the typed character; P therefore becomes an upper bound on the selection accuracy, and similarly will ITR become an upper bound on the information transfer rate.

5 DISCUSSION

This paper shows the working of an application – EyeTell, using smooth-pursuit eye movements for eye-typing on a tablet with only the front-facing camera, in a controlled environment. The application achieves reasonable accuracy without requiring the process of calibration or the eye-tracker equipment.

The average typing speed achieved using EyeTell (1.28 WPM) was lower than that using other smooth-pursuit based eye-typing applications (3.34 WPM using SMOOVS [Lutz et al. 2015]). It is highly likely that a longitudinal experiment, with multiple sessions over days [Majaranta et al. 2009; Mott et al. 2017] and the possibility of increased practice would lead to an improved typing speed as well as reduced error rate, the signs of which are observed in the Figure 3. However, the experiment was designed to test short interactions with the application and the ease of use.

The false detection error rate of the application was $\sim 20\%$ in the first trial of the first session, which revealed the average accuracy of the application to be 80%, which further improved with increased practice (see Figure 3b). This is comparable to the total errors made by the application TextPursuits in selecting targets of short text [Khamis et al. 2016]. The improvement in the application accuracy, however, could indicate that selecting and following one target among multiple moving targets might be slightly different from the natural smooth-pursuit eye movements of following a bird in the sky.

The ITR of 33 bits/min achieved by the application was higher than the reported maximum ITR obtained by brain-computer interfaces of 25 bits/min by Wolpaw et al. [2002]. For people with disabilities, with even slight control of eye movements, this application could provide easy and intuitive technique of communication.

The layout used by EyeTell, with targets moving along two concentric circles, is novel, even for applications using smooth-pursuit eye movements. The increased number of targets on the screen (10-12 compared to six in SMOOVS [Lutz et al. 2015]) may generate slight complexity in the application. However, the search time did not improve over the trials (see Figure 2), and the difference in the NASA-TLX scores between the two sessions was marginal (see Figure 4), which might suggest that the increased number of possible targets do not have an effect on the application performance or the ease of use. However, it is a limitation of our study that a direct comparison of user acceptance and task load was not performed. Moreover, the speed of the targets was constant at $48°/s$ in the experiments, and only one participant complained about it being fast. We note here that the participant who found the targets moving too fast was slightly older than most of the participants and belonged to the age group between 38–47 years.

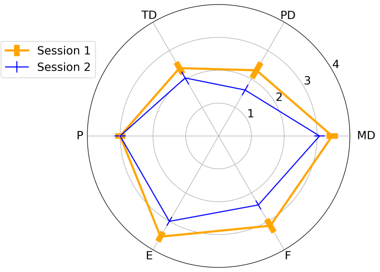


Figure 4: The NASA-TLX sub-scores (MD: mental demand, PD: physical demand, TD: temporal demand, P: performance, E: effort and F: frustration) provided by the participants, for session 1 (orange) and session 2 (blue). The error bars depict the standard error.

An overall investigation includes the strengths and limitations of the gaze tracking system. A strength of the system is that gaze tracking on an 11-inch iPad, using its own camera, for text entry was found to be feasible. The application was developed on an affordable consumer device, using open-source technical services offered by the device (ARKit). Moreover, the small size of the application could serve towards privacy concerns in public use of such devices, such as typing a private pincode. On the other hand, the small size of the iPad could also be a weakness for the application as it stands, and the reason for the low robustness and the high false selections by the application ($\sim$ 20%). Finally, the method of detecting the target using normalized error in the distance between the target and gaze points might restrict the maximum number of targets available for selection. Further improvement of the application with customization towards the special use, such as improved accuracy for assistive devices or higher number of available targets for a keyboard, may be required.

In spite of the low typing speed, there may be a number of cases where smooth-pursuit typing with standard devices may serve as a supplementary assistive device. It affords adhoc short messaging that may be a back-up solution when special gaze tracking systems fails. Also, some users may feel it to be less socially stigmatizing using a device such as an iPad for communication than special equipment. The experiment presented in this paper uses eye-typing, but in real life, an application with three options of 'yes', 'no' and 'ask again' could be applied in hospitals for non-verbal communication after surgery, or a more playful application could be designed using similar technology to train children with disability to control their eye movements for communication. Moreover, hands-free interaction could be useful in other scenarios in the public (e.g.: rating the shopping experience with shopping bags in the hand) or private (e.g.: attend to notifications while cooking).

ACKNOWLEDGMENTS

This work was partially funded by the Bevica foundation and the Horizon 2020 Framework Programme of the European Union under the ReHyb Project, Grant Agreement Number 871767.

REFERENCES

Yasmeen Abdrabou, Mariam Mostafa, Mohamed Khamis, and Amr Elmougy. 2019. Calibration-free text entry using smooth pursuit eye movements. In *In Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. 1–5. https://doi.org/10.1145/3314111.3319838

Irshad Abibouraguimane, Kakeru Hagihara, Keita Higuchi, Yoichi Sato, Tetsu Hayashida, and Maki Sugimoto. 2019. CoSummary: Adaptive Fast-Forwarding for Surgical Videos by Detecting Collaborative Scenes Using Hand Regions and Gaze Positions. In *In Proceedings of the 24th International Conference on Intelligent User Interfaces*. 580–590. https://doi.org/10.1145/3301275

Laura J. Ball, Amy S. Nordness, Susan K. Fager, Katie Kersch, Brianae Mohr, Gary L. Pattee, David R. Beukelman, Gary L. Pattee, David R. Beukelman, Gary L. Pattee, and David R. Beukelman. 2010. Eye-Gaze Access to AAC Technology for People with Amyotrophic Lateral Sclerosis. *Journal of Medical Speech-Language Pathology* 18, 3

(2010), 11–23. https://doi.org/10.1001/jamaoto.2013.35 arXiv:0604217 [arXiv:cond-mat]

Nora Castner, Enkelejda Kasneci, Thomas Kübler, Katharina Scheiter, Juliane Richter, Thérése Eder, Fabian Hüttig, and Constanze Keutel. 2017. Scanpath comparison in medical image reading skills of dental students Distinguishing stages of expertise development. In *In Proceedings of the 10th ACM Symposium on Eye Tracking Research & Applications*. 1–9. https://doi.org/10.1145/3204493.3204550

Dietlind Helene Cymek, Stefan Ruff, Simon Hofmann, Antje Christine Venjakob, Otto Hans Martin Lutz, and Matthias Roetting. 2014. Entering PIN codes by smooth pursuit eye movements. *Journal of Eye Movement Research* 7, 4 (2014), 1–11. https://doi.org/10.16910/jemr.7.4.1

Heiko Drewes, Mohamed Khamis, and Florian Alt. 2019. Dialplates: Enabling pursuits-based user interfaces with large target numbers. In *ACM International Conference Proceeding Series*. 1–10. https://doi.org/10.1145/3365610.3365626

John Paulin Hansen, Dan Witzner Hansen, and Anders Sewerin Johansen. 2001. Bringing Gaze-based Interaction Back to Basics. In *Universal Access In HCI*. 325–328.

Mohamed Khamis, Ozan Saltuk, Alina Hang, Katharina Stolz, Andreas Bulling, and Florian Alt. 2016. TextPursuits: Using text for pursuits-based interaction and calibration on public displays. In *In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 274–285. https://doi.org/10.1145/2971648.2971679

Timothée Lenglet, Jonathan Mirault, Marie Veyrat-Masson, Aurélie Funkiewiez, Maria Del Mar Amador, Gaelle Bruneteau, Nadine Le Forestier, Pierre Francois Pradat, Francois Salachas, Yannick Vacher, Lucette Lacomblez, and Jean Lorenceau. 2019. Cursive eye-writing with smooth-pursuit eye-movement is possible in subjects with amyotrophic lateral sclerosis. *Frontiers in Neuroscience* 13, MAY (2019), 1–11. https://doi.org/10.3389/fnins.2019.00538

Otto Hans Martin Lutz, Antje Christine Venjakob, and Stefan Ruff. 2015. SMOOVS: Towards calibration-free text entry by gaze using smooth pursuit movements. *Journal of Eye Movement Research* 8, 1 (2015). https://doi.org/10.16910/jemr.8.1.2

Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. 2009. Fast gaze typing with an adjustable dwell time. In *In Proceedings of the 27th international conference on Human factors in computing systems*. 357. https://doi.org/10.1145/1518701.1518758

Martez E Mott, Shane Williams, Jacob O. Wobbrock, and Meredith Ringel Morris. 2017. Improving Dwell-Based Gaze Typing with Dynamic, Cascading Dwell Times. In *In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2558–2570. https://doi.org/10.1145/3025453.3025517

Bernhard Obermaier, Christa Neuper, Christoph Guger, and Gert Pfurtscheller. 2001. Information transfer rate in a five-classes brain-computer interface. *IEEE Transactions on neural systems and rehabilitation engineering* 9, 3 (2001), 283–288. https://doi.org/10.1109/7333.948456

Sahar Sadeghi and Ali Maleki. 2019. Accurate estimation of information transfer rate based on symbol occurrence probability in brain-computer interfaces. *Biomedical Signal Processing and Control* 54 (2019), 101607. https://doi.org/10.1016/j.bspc.2019.101607

William Speier, Corey Arnold, and Nader Pouratian. 2013. Evaluating true BCI communication rate through mutual information and language models. *PLoS One* 8, 10 (2013), e78432. https://doi.org/10.1371/journal.pone.0078432

David J. Ward, Alan F. Blackwell, and David J. C. MacKay. 2000. Dasher—a data entry interface using continuous gestures and language models. In *In Proceedings of the 13th annual ACM symposium on User interface software and technology*. 129–137. https://doi.org/10.1145/354401.354427

Jacob O. Wobbrock and Brad A. Myers. 2006. Analyzing the input stream for character-level errors in unconstrained text entry evaluations. *ACM Transactions on Computer-Human Interaction* 13, 4 (2006), 458–489. https://doi.org/10.1145/1188816.1188819

Jacob O. Wobbrock, Michael W Sawyer, and Andrew T. Duchowski. 2008. Longitudinal Evaluation of Discrete Consecutive Gaze Gestures for Text Entry. In *In Proceedings of the 2008 symposium on Eye tracking research & applications*. 11–19. https://doi.org/10.1145/1344471.1344475

Jonathan R Wolpaw, Niels Birbaumer, Dennis J McFarland, Gert Pfurtscheller, and Theresa M Vaughan. 2002. Brain–computer interfaces for communication and control. *Clinical neurophysiology* 113, 6 (2002), 767–791. https://doi.org/10.1016/s1388-2457(02)00057-3

Zhe Zeng and Matthias Roetting. 2018. A text entry interface using smooth pursuit movements and language model. In *In Proceedings of the 10th ACM Symposium on Eye Tracking Research & Applications*. 1–2. https://doi.org/10.1145/3204493.3207413