GazeWheels: Comparing Dwell-time Feedback and Methods for Gaze Input

MISAHAEL FERNANDEZ, FLORIAN MATHIS, and MOHAMED KHAMIS, University of Glasgow, UK

We present an evaluation and comparison of GazeWheels: techniques for dwell time gaze input and feedback. In GazeWheel, visual feedback is shown to the user in the form of a wheel that is filled. When completely filled, a selection is made where the user is gazing. We compare three methods for responding to the user when gazing away from the target: Resetting GazeWheel, Pause-and-Resume GazeWheel, and Infinite GazeWheel. We also compare the position of the GazeWheel; Co-located Feedback: shown on the target being gazed at, and Remote Feedback: shown at the top of the interface. To this end, we report on results of a user study (N=19) that investigates the benefits and drawbacks of each method at different dwell times: 500ms, 800ms, and 1000ms. Results show that Infinite GazeWheel and Pause-and-Resume GazeWheel are more error prone but significantly faster than Resetting GazeWheel when using 800-1000 ms dwell time, even when including the time for correcting errors.

CCS Concepts: • **Human-centered computing → Interaction techniques**.

Additional Key Words and Phrases: Gaze-based input, Dwell Time, Visual Feedback

ACM Reference Format:
Misahael Fernandez, Florian Mathis, and Mohamed Khamis. 2020. GazeWheels: Comparing Dwell-time Feedback and Methods for Gaze Input. To appear in *NordiCHI '20: Nordic Conference on Human-Computer Interaction, October 25–29, 2020, Tallinn, Estonia.* ACM, New York, NY, USA, 8 pages.

1 INTRODUCTION AND BACKGROUND

Gaze interface designers have traditionally used dwell time to improve the usability of gaze input. The need for dwell time selection arose due to the Midas Touch problem [8, 9] which occurs when a user inadvertently selects a target via gaze while examining it. This is a general problem with gaze interfaces because the eyes are perceptual organs, and humans are not acquainted with controlling them to provide input [18]. Gaze interaction techniques that minimize this problem were proposed, such as Pursuits [6, 26] and gaze gestures [5, 11, 12]. Pursuits relies on smooth pursuit eye movements performed when following on-screen moving targets [26], while gaze gestures require users to perform coarse eye movements from one position to another [5], thereby both methods reduce chances of unintentional selections. That being said, gaze interfaces that require dwell time will pertain to exist as none of the newly proposed techniques allow users to "point and select" the same way as done using a mouse or a touchscreen. This underlines the importance of improving the usability and performance of dwell-time based gaze interfaces. Studies on feedback methods for dwell time interaction revealed effects on input speed, accuracy, gaze behavior and subjective experiences [19]. Several works investigated improving gaze input speed by, for example, adjusting the dwell duration dynamically [3, 16, 17, 21, 22, 27]. Dwell time was also compared with other gaze-based methods, such as gestures and taps [28],

In this work, we present multiple techniques for dwell-time gaze interfaces that utilize a GazeWheel—a visual feedback widget in the shape of a wheel that is filled as the user is temporally closer to the selection event. In particular, we experiment with different ways the GazeWheel reacts to the user's gaze when looking away from the target: Resetting GazeWheel, Pause-and-Resume GazeWheel, and Infinite GazeWheel. In Resetting GazeWheel, the GazeWheel resets when the user looks away from any target. In Pause-and-Resume GazeWheel, the GazeWheel is paused when the user looks away, and resumes at the same state when the user looks at the same or a new target. In Infinite GazeWheel,

Manuscript submitted to ACM

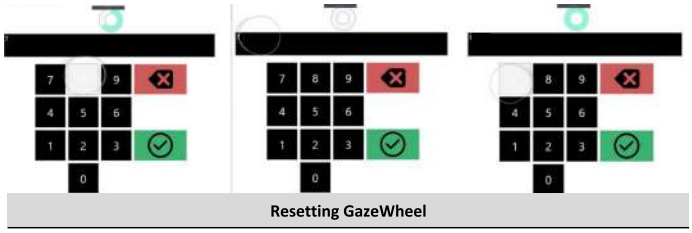


Fig. 1. Resetting GazeWheel (baseline): When the user gaze at a target, the GazeWheel starts to gradually fill (left). Gazing away from the target, causes the GazeWheel to *reset* (middle). When the user gazes at another (or the same) target, the GazeWheel *starts from the empty state* and progresses until it is fully filled to make the selection.

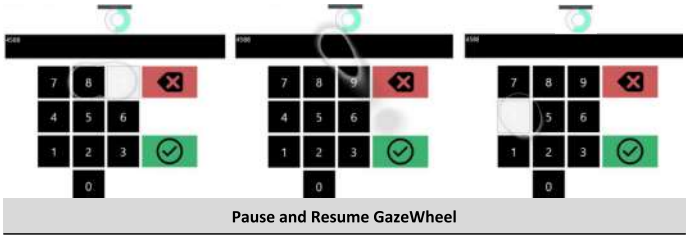


Fig. 2. Pause-and-Resume GazeWheel: when the user gazes at a target, the GazeWheel starts to gradually fill (left). Gazing away from the target causes the GazeWheel to *pause at the current state* (middle). As the user gazes at another (or the same) target, the GazeWheel *resumes* and progresses until it is fully filled to make the selection.

the GazeWheel progresses endlessly regardless the user is looking at any of the targets or not. In all three methods, a selection is made when the GazeWheel is completely filled. We further compare displaying the GazeWheel on the target (Co-located Feedback) or remotely at the top of the interface (Remote Feedback), and using different dwell time durations that are motivated by prior work [1, 6]: 500 ms, 800 ms, and 1000 ms. We compare the aforementioned conditions in a within subjects user study with 19 participants who used the techniques on a sample gaze-based PIN pad. We found that although Infinite and Pause-and-Resume GazeWheel are more error prone than Resetting GazeWheel, they are significantly faster when using a dwell time of 800-1000 ms. Assuming that users make less errors over time due to training, we found that when excluding error correction time, Infinite and Pause-and-Resume GazeWheel are significantly faster than Resetting GazeWheel at all dwell times. The main **contribution** of this work is a user-centered comparison of multiple designs of GazeWheel in terms of selection time and error rates.

2 GAZEWHEELS

In all three methods, a target is selected when the GazeWheel is completely filled while the user is gazing at the target. Some design decisions are inspired by Majaranta et al.'s guidelines for feedback on dwell time interaction [19].

1. Resetting GazeWheel: GazeWheel starts filling when the user gazes at any target, and it is reset when looking away from the target (Figure 1). This is the traditional implementation of gaze entry via dwell time and hence treated as the baseline in our experiment.

2. Pause-and-Resume GazeWheel: GazeWheel starts filling when the user gazes at a target. When the user looks away, the GazeWheel is paused (i.e., its state is saved) and resumes at the last state when the user looks at the same or another target. It is reset only after being completely filled (Figure 2).

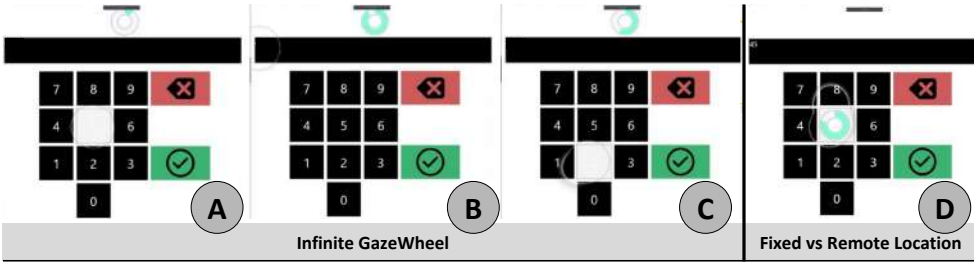


Fig. 3. Infinite GazeWheel: The GazeWheel starts to gradually fill as soon as the user makes visual contact with any target (A). The GazeWheel keeps progressing even when the user gazes away from the target. At the moment the GazeWheel is completely filled then (1) If the user is gazing at a target, a selection is made (2) If no target is gazed at, no selection takes place and the GazeWheel resets.In addition to the three feedback methods, we compared showing GazeWheel in a fixed remote location (A, B, C in this figure and in Figures 1 and 2) to being co-located: on the target itself (D in this figure). **Note:** *in all figures, we show a bubble to indicate user's gaze for illustration only. In our study, no indicator of user's gaze was shown to avoid confusing users.*

3. Infinite GazeWheel: GazeWheel starts filling automatically once the interface is loaded independent of whether or not the user gazes at a target. It does not reset or pause when gazing away, instead, it infinitely resets when completely filled. A target is selected only when the GazeWheel is completely filled while gazing at said target (Figure 3).
We also experimented with the location of the GazeWheel, and the required dwell duration:

Remote vs Co-located Feedback: We compared displaying the feedback on a remote fixed location at the top of the targets to displaying it co-located with target on its top (Figure 3-D).

Dwell durations: Increasing the dwell time reduces input errors, but also reduces input speed. On the other hand, decreasing dwell time increases selection speed but could result in unintended inputs. We chose three dwell time durations to experiment with based on prior work: 500 ms [4, 7], 800 ms [4, 7], and 1000 ms [6] .

3 IMPLEMENTATION AND STUDY SETUP

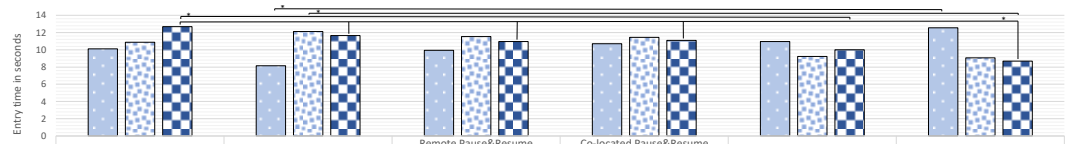
Our implementation is in C# and uses the Windows Presentation Foundation (WPF) framework and Tobii's Core SDK [25] to capture the gaze points and adapt the interface accordingly. The interface shows multiple WPF frames, one at a time. We show a welcome screen in which we log the participant's ID and demographic data (age, gender, familiarity with eye tracking). A second frame allows the experimenter to choose which of the 6 conditions to load (3 GazeWheel Methods × 2 Feedback Locations) and one of the dwell durations. A frame was implemented for each condition. Each of those frames shows a PIN pad. We chose PIN entry as a task for the participants as password entry is a popular application of eye gaze [1, 2, 10, 13, 20]. Experiment data (e.g., entry time, errors) were stored anonymously in a local CSV file. We used a Tobii 4C eye tracker (90Hz) [24] and ran our prototype on a Lenovo IdeaPad 530S.

4 USER STUDY

Our study was designed as a repeated measures experiment. All participants went through all conditions. Counterbalancing was done using a Latin Square. Participants volunteered in our study which was advertised through mailing lists and word of mouth. We recruited 19 participants: 1 females and 9 males, aged 19-36 years (M=24.8, SD=4.05). Out of those, 11 were wearing eyeglasses or contact lenses. The study complied with our university's ethics procedure.

4.1 Study Procedure

The experimenter started by explaining the purpose of the study and asked the participant to sign a consent form. After that, we connected the eye tracker and calibrated it for the participant. The participant was then provided with a



	Remote Resetting GazeWheel	Co-located Resetting GazeWheel	Remote Pause&Resume GazeWheel	Co-located Pause&Resume GazeWheel	Remote Infinite GazeWheel	Co-located Infinite GazeWheel
500ms	10.10	8.14	9.94	10.70	10.95	12.55
800ms	10.87	12.12	11.53	11.44	9.22	9.06
1000ms	12.68	11.65	10.95	11.09	10.00	8.69

Fig. 4. Overall mean entry times including the time taken for correcting errors. Significance of $p < .05$ is denoted by *.

training sheet in which the GazeWheel methods and the participant's task were described. If there were no questions, the experimenter provided the participant with a sheet of 60 PINs to enter via gaze using the given interface. The PINs were all of length four, to ensure comparability to prior work [1, 14], and were generated randomly. After providing their demographics, participant underwent a training phase in which they entered one PIN using each of the six conditions. This data was for training only and was not used in the analysis. Once the training phase was over, participants went through six blocks according to the Latin Square: one block per condition. In each block, the participant entered nine PINs using one of the six conditions, each three followed one of the three dwell durations. Participants were allowed to use the "delete" button. The PINs were listed on a sheet of paper handed earlier to the participant, who had to read the PIN and then enter it without examining the sheet again. After all six blocks, we ran a semi-structured interview.

4.2 Limitations

A limitation in our study is that participants had to remember the password from the time they read it until they enter it. This may have had an impact on reported feedback and perceived difficulty. However, in a real-world application, users would need to enter fewer PINs and hence spend less time interacting with the application compared to the study. This means reported difficulty might be overrated. However, we expect that relative differences between the conditions to remain similar. Moreover, we used PIN entries as a use case in our study.

5 RESULTS

We collected 1026 entries (54 per participant). For each entry, we measured entry time, and error rate.

5.1 Overall Entry Time

Overall entry time was measured from the moment the user started entering the first digit of a PIN, until the moment the last digit was entered. Results are summarized in Figure 4. It can be seen that Resetting GazeWheel with Co-located Feedback at 500 ms is associated with the lowest entry time across all conditions (8.14 s), followed immediately by Infinite GazeWheel with Co-located Feedback at 1000 ms (8.69 s), and Infinite GazeWheel with Co-located Feedback at 800 ms (9.06 s). We ran a three-way repeated measures ANOVA and follow-up two-way ANOVAs when interaction effects were found. Greenhouse-Geisser correction was used when there was a violation of the sphericity assumption. We found a statistically significant two-way interaction between the GazeWheel methods and dwell duration on entry time $F_{(6.727,369.980)} = 4.077$, $p < .05$. We did not find any other statistically significant interaction effects. Further analysis was conducted to investigate the impact of the GazeWheel methods on entry time. Individual ANOVAs for each dwell time condition and post hoc t-tests with Bonferroni correction revealed significant differences in all three dwell conditions ($p < 0.05$). For a dwell duration of 500ms, co-located Resetting GazeWheel *(M=8.14,SD=2.84)* is significantly faster than co-located Infinite GazeWheel *(M=12.55,SD=10.47)*. For a dwell duration of 800ms, we found that collocated Infinite

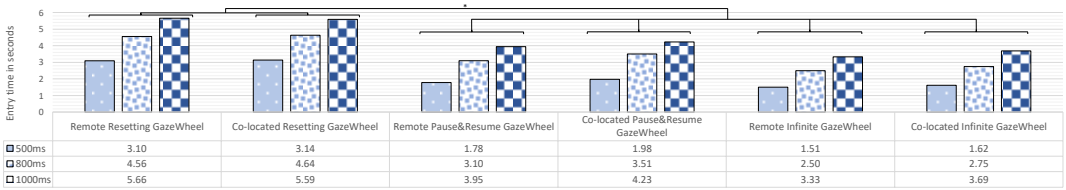


Fig. 5. When excluding error correction time, entry time using Infinite GazeWheel is the fastest, followed by Pause-and-Resume GazeWheel. Remote Feedback makes users slightly faster than Co-located Feedback. Significance of $p < .05$ is denoted by *.

GazeWheel *(M=9.06,SD=5.70)* is significantly faster than co-located Resetting GazeWheel *(M=12.12,SD=6.76)* ($p < 0.05$). For a dwell duration of 1000ms, we found that co-located Infinite GazeWheel *(M=8.69, SD=4.03)* is significantly faster than remote Pause-and-Resume GazeWheel *(M=10.95, SD=5.35)*, co-located Pause-and-Resume GazeWheel *(M=11.09, SD=4.49)*, remote Resetting GazeWheel *(M=12.68, SD=4.89)*, and co-located Resetting GazeWheel *(M=11.65, SD=4.70)*. Moreover, we found that remote Infinite GazeWheel *(M=10.00, SD=4.41)* is significantly faster than remote Resetting GazeWheel *(M=12.68, SD=4.89)*.

5.2 Successful Entry Time

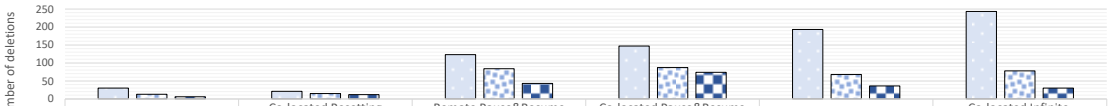
Since participants were able to undo previous entries using the delete button, we also measured the entry time for correctly entered PINs only. We found that participants provided the fastest input across all dwell durations of 500 ms (Figure 5). Successful selections are generally faster when using Remote Feedback rather than Co-located Feedback, and fastest when using Infinite GazeWheel, followed by Pause-and-Resume GazeWheel and then Resetting GazeWheel. We also applied a three-way repeated measures ANOVA on the successful entry times and followed with two-way ANOVAs on each simple two-way interaction. We were particularly interested in significant differences between the GazeWheel methods on each dwell time level. We found for all dwell durations that remote Infinite GazeWheel, co-located Infinite GazeWheel, remote Pause-and-Resume GazeWheel and co-located Pause-and-Resume GazeWheel are significantly faster than remote Resetting GazeWheel and co-located Resetting GazeWheel ($p < 0.05$).

5.3 Number of Errors

The interface did not allow submitting any incorrect entries, and participants had to enter all PINs. Thus, the number of deletions is an indicator of the number of errors. As summarized in Figure 6, Resetting GazeWheel is the least error prone condition, while Infinite GazeWheel is the highest error prone. In most cases, the Co-located Feedback variations are more error prone compared to Remote Feedback ones.

5.4 Qualitative feedback

Participants had mixed feedback about the Pause-and-Resume GazeWheel Remote Feedback. Participants perceived it as easy (6), hard (4) and uncomfortable (1). Infinite GazeWheel Remote Feedback was similarly perceived, but one participant noted that they would have preferred shorter dwell durations (e.g., 350 ms or 400 ms). Resetting GazeWheel was perceived to be easy (5), comfortable (1) and reliable (3), but also slow (4). As for the Co-located Feedback versions, Pause-and-Resume GazeWheel was found hard (3) and tiring (1). Participants had mixed opinions about whether it is fast (1) or slow (2). Similarly, Infinite GazeWheel received mixed feedback, with some participants finding it easy (8) but hard for others (5). Resetting GazeWheel was found again to be easy (6), comfortable (1), but slow (7).



	Remote Resetting GazeWheel	Co-located Resetting GazeWheel	Remote Pause&Resume GazeWheel	Co-located Pause&Resume GazeWheel	Remote Infinite GazeWheel	Co-located Infinite GazeWheel
500ms	30	21	123	147	193	243
800ms	13	15	84	87	68	78
1000ms	6	12	43	74	36	30

Fig. 6. Deletions count per condition across all participants. Errors are more common in Infinite and Pause-and-Resume GazeWheel.

6 DISCUSSION, LIMITATIONS AND FUTURE WORK

Interestingly, although Infinite GazeWheel and Pause-and-Resume GazeWheel elicit more errors compared to Resetting GazeWheel, overall completion times (which include the time taken to fix errors) vary only slightly. For example, entry time using Infinite GazeWheel is between 9.06 s and 12.55 s, while for Resetting GazeWheel it is between 8.14 s and 12.68 s. This suggests that Infinite GazeWheel and Pause-and-Resume GazeWheel can significantly improve entry time despite the fact that they are error prone. However, we note that while the low error tolerance can be frustrating for some users, learning effects are expected to make users faster over time. Based on the mixed feedback on Infinite GazeWheel and Pause-and-Resume GazeWheel, it can be argued that people perceive these implementations differently. Therefore, real-world deployments of this work should allow users to choose the implementation they prefer. Future work should also investigate additional dwell durations. Participants noted they would have preferred an even faster Infinite GazeWheel. We did not capture any significant influences of the GazeWheel Feedback's location on users' performance or perception. However, there is a slight tendency for less error prone entries when using the Remote Feedback versions of GazeWheel. Promising directions for future work include investigating the use of GazeWheel for accessibility. Hereof, some participants noted that the concept of GazeWheel sounds promising for people with disabilities or when users' hands are occupied. Future work should also investigate different modalities for feedback; in particular, vibrotactile [23] and auditory [15] feedback are generally promising for gaze interfaces.

We evaluated GazeWheels on a PIN interface only. This is motivated by the growing importance of gaze input in security [10]. However, the results are valid for interfaces that feature selectable options, such as supermarket self-checkout machines, ticket machines and interfaces deployed in public where hygienic interactions are of concern.

7 CONCLUSION

In this work, we introduced three GazeWheel methods: Infinite GazeWheel, Pause-and-Resume GazeWheel and Resetting GazeWheel and evaluated their use for Co-located Feedback and Remote Feedback at 500 ms, 800 ms, and 1000 ms. Using PIN entry as a use case, we reported on results from a user study with 19 participants which show that Infinite GazeWheel and Pause-and-Resume GazeWheel are more error prone than Resetting GazeWheel but faster to use despite the high error rate. Feedback from participants is mixed in many cases, suggesting that GazeWheels are promising for certain user groups who do not find it overwhelming, and that more work is needed to better understand continuous feedback mechanisms like GazeWheel.

ACKNOWLEDGMENTS

This work was supported by the Royal Society of Edinburgh (RSE award number 65040) and the University of Edinburgh and the University of Glasgow jointly funded PhD studentships.

REFERENCES

[1] Yasmeen Abdrabou, Mohamed Khamis, Rana Mohamed Eisa, Sherif Ismail, and Amrl Elmougy. 2019. Just Gaze and Wave: Exploring the Use of Gaze and Gestures for Shoulder-Surfing Resilient Authentication. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. ACM, New York, NY, USA, Article Article 29, 10 pages. https://doi.org/10.1145/3314111.3319837

[2] Darrell S. Best and Andrew T. Duchowski. 2016. A Rotary Dial for Gaze-based PIN Entry. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. ACM, New York, NY, USA, 69–76. https://doi.org/10.1145/2857491.2857527

[3] Zhaokang Chen and Bertram E. Shi. 2019. Using Variable Dwell Time to Accelerate Gaze-Based Web Browsing with Two-Step Selection. *International Journal of Human–Computer Interaction* 35, 3 (2019), 240–255. https://doi.org/10.1080/10447318.2018.1452351 arXiv:https://doi.org/10.1080/10447318.2018.1452351

[4] Alexander De Luca, Martin Denzel, and Heinrich Hussmann. 2009. Look into My Eyes!: Can You Guess My Password?. In *Proceedings of the 5th Symposium on Usable Privacy and Security (SOUPS '09)*. ACM, New York, NY, USA, Article 7, 12 pages. https://doi.org/10.1145/1572532.1572542

[5] Heiko Drewes and Albrecht Schmidt. 2007. *Interacting with the Computer Using Gaze Gestures*. Springer Berlin Heidelberg, Berlin, Heidelberg, 475–488. https://doi.org/10.1007/978-3-540-74800-7_43

[6] 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 (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 457–466. https://doi.org/10.1145/2807442.2807499

[7] Alain Forget, Sonia Chiasson, and Robert Biddle. 2010. Shoulder-Surfing Resistance with Eye-Gaze Entry in Cued-Recall Graphical Passwords. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. Association for Computing Machinery, New York, NY, USA, 1107–1110. https://doi.org/10.1145/1753326.1753491

[8] Robert J. K. Jacob. 1990. What You Look at is What You Get: Eye Movement-based Interaction Techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '90)*. ACM, New York, NY, USA, 11–18. https://doi.org/10.1145/97243.97246

[9] Robert J. K. Jacob. 1991. The Use of Eye Movements in Human-computer Interaction Techniques: What You Look at is What You Get. *ACM Trans. Inf. Syst.* 9, 2 (April 1991), 152–169. https://doi.org/10.1145/123078.128728

[10] Christina Katsini, Yasmeen Abdrabou, George Raptis, Mohamed Khamis, and Florian Alt. 2020. The Role of Eye Gaze in Security and Privacy Applications: Survey and Future HCI Research Directions.. In *Proceedings of the 38th Annual ACM Conference on Human Factors in Computing Systems (CHI '20)*. ACM, New York, NY, USA, 21. https://doi.org/10.1145/3313831.3376840

[11] Mohamed Khamis, Florian Alt, Mariam Hassib, Emanuel von Zezschwitz, Regina Hasholzner, and Andreas Bulling. 2016. GazeTouchPass: Multimodal Authentication Using Gaze and Touch on Mobile Devices. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 2156–2164. https://doi.org/10.1145/2851581.2892314

[12] Mohamed Khamis, Mariam Hassib, Emanuel von Zezschwitz, Andreas Bulling, and Florian Alt. 2017. GazeTouchPIN: Protecting Sensitive Data on Mobile Devices Using Secure Multimodal Authentication. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction (ICMI '17)*. Association for Computing Machinery, New York, NY, USA, 446–450. https://doi.org/10.1145/3136755.3136809

[13] Mohamed Khamis, Carl Oechsner, Florian Alt, and Andreas Bulling. 2018. VRpursuits: Interaction in Virtual Reality Using Smooth Pursuit Eye Movements. In *Proceedings of the 2018 International Conference on Advanced Visual Interfaces (AVI '18)*. Association for Computing Machinery, New York, NY, USA, Article Article 18, 8 pages. https://doi.org/10.1145/3206505.3206522

[14] Mohamed Khamis, Ludwig Trotter, Ville Mäkelä, Emanuel von Zezschwitz, Jens Le, Andreas Bulling, and Florian Alt. 2018. CueAuth: Comparing Touch, Mid-Air Gestures, and Gaze for Cue-based Authentication on Situated Displays. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4, Article 174 (Dec. 2018), 21 pages. https://doi.org/10.1145/3287052

[15] Anne Köpsel, Päivi Majaranta, Poika Isokoski, and Anke Huckauf. 2016. Effects of auditory, haptic and visual feedback on performing gestures by gaze or by hand. *Behaviour & Information Technology* 35, 12 (2016), 1044–1062. https://doi.org/10.1080/0144929X.2016.1194477 arXiv:https://doi.org/10.1080/0144929X.2016.1194477

[16] 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 (CHI '09)*. Association for Computing Machinery, New York, NY, USA, 357–360. https://doi.org/10.1145/1518701.1518758

[17] Päivi Majaranta, Anne Aula, and Kari-Jouko Räihä. 2004. Effects of Feedback on Eye Typing with a Short Dwell Time. In *Proceedings of the 2004 Symposium on Eye Tracking Research & Applications (ETRA '04)*. Association for Computing Machinery, New York, NY, USA, 139–146. https://doi.org/10.1145/968363.968390

[18] Päivi Majaranta and Andreas Bulling. 2014. *Eye Tracking and Eye-Based Human–Computer Interaction*. Springer London, 39–65. https://doi.org/10.1007/978-1-4471-6392-3_3

[19] Päivi Majaranta, Scott MacKenzie, Anne Aula, and Kari-Jouko Räihä. 2006. Effects of Feedback and Dwell Time on Eye Typing Speed and Accuracy. *Univers. Access Inf. Soc.* 5, 2 (July 2006), 199–208. https://doi.org/10.1007/s10209-006-0034-z

[20] Florian Mathis, John Williamson, Kami Vaniea, and Mohamed Khamis. 2020. RubikAuth: Fast and Secure Authentication in Virtual Reality. In *Proceedings of the 38th Annual ACM Conference on Human Factors in Computing Systems (CHI EA '20)*. ACM, New York, NY, USA. https://doi.org/10.1145/3334480.3382827

[21] 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 (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 2558–2570. https://doi.org/10.1145/3025453.3025517

[22] 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 (CHI '12)*. Association for Computing Machinery, New York, NY, USA, 3001–3010. https://doi.org/10.1145/2207676.2208711

[23] Jussi Rantala, Päivi Majaranta, Jari Kangas, Poika Isokoski, Deepak Akkil, Oleg Špakov, and Roope Raisamo. 2020. Gaze Interaction With Vibrotactile Feedback: Review and Design Guidelines. *Human–Computer Interaction* 35, 1 (2020), 1–39. https://doi.org/10.1080/07370024.2017.1306444 arXiv:https://doi.org/10.1080/07370024.2017.1306444

[24] Tobii. 2020. Tobii 4C eye tracker. https://gaming.tobii.com/product/tobii-eye-tracker-4c/. Accessed 03 February 2020.

[25] Tobii. 2020. Tobii SDK. https://developer.tobii.com/consumer-eye-trackers/core-sdk/getting-started/. Accessed 03 February 2020.

[26] Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: Spontaneous Interaction with Displays Based on Smooth Pursuit Eye Movement and Moving Targets. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*. ACM, New York, NY, USA, 439–448. https://doi.org/10.1145/2493432.2493477

[27] Oleg Špakov and Darius Miniotas. 2004. On-Line Adjustment of Dwell Time for Target Selection by Gaze. In *Proceedings of the Third Nordic Conference on Human-Computer Interaction (NordiCHI '04)*. Association for Computing Machinery, New York, NY, USA, 203–206. https://doi.org/10.1145/1028014.1028045

[28] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. 2017. Tap, Dwell or Gesture? Exploring Head-Based Text Entry Techniques for HMDs. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 4479–4488. https://doi.org/10.1145/3025453.3025964