DialPlates: Enabling Pursuits-based User Interfaces with Large Target Numbers

Heiko Drewes
heiko.drewes@ifi.lmu.de
LMU Munich
Munich, Germany

Mohamed Khamis
mohamed.khamis@glasgow.ac.uk
University of Glasgow
Glasgow, UK

Florian Alt
florian.alt@unibw.de
Bundeswehr University
Munich, Germany

ABSTRACT

In this paper we introduce a novel approach for smooth pursuits eye movement detection and demonstrate that it allows up to 160 targets to be distinguished. With this work we advance the well-established smooth pursuits technique, which allows gaze interaction without calibration. The approach is valuable for researchers and practitioners, since it enables novel user interfaces and applications to be created that employ a large number of targets, for example, a pursuits-based keyboard or a smart home where many different objects can be controlled using gaze. We present findings from two studies. In particular, we compare our novel detection algorithm based on linear regression with the correlation method. We quantify its accuracy for around 20 targets on a single circle and up to 160 targets on multiple circles. Finally, we implemented a pursuits-based keyboard app with 108 targets as proof-of-concept.

CCS CONCEPTS

• **Human-centered computing** → **Interaction techniques**.

KEYWORDS

Smooth pursuit detection; distinguishable pursuit targets; gaze-only text entry; linear regression.

ACM Reference Format:
Heiko Drewes, Mohamed Khamis, and Florian Alt. 2019. DialPlates: Enabling Pursuits-based User Interfaces with Large Target Numbers. In *MUM 2019: 18th International Conference on Mobile and Ubiquitous Multimedia (MUM 2019), November 26–29, 2019, Pisa, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3365610.3365626

1 INTRODUCTION

Gaze-based interfaces hold many promises: they work over distances, they are hygienic as there is nothing to touch, they keep hands free for other tasks, they are silent, and they are maintenance-free as eye trackers have no moving parts. However, gaze-based UIs usually need a time-consuming calibration, they lack high accuracy, and they are prone to the so-called Midas touch problem [14].

In 2013, Vidal et al. introduced a concept for gaze interaction based on smooth pursuit eye movements [32, 33]. In interfaces with moving targets, they compare a user's gaze and the movement of the target, hence allowing a matching pursuit movement to be detected by calculating Pearson's correlation coefficient. The strength of this approach is its independence from offset and scaling and, therefore, the eye tracker does not need calibration but can be instantly used. Another advantage is that due to being scale-independent, small interfaces, e.g., for a smartwatch [11], can be built.

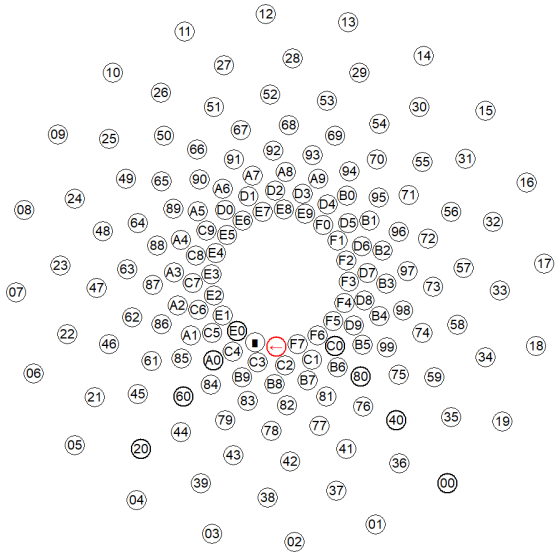

Figure 1: The 160 pursuit targets dial plate. The targets are organized in 8 circles with 20 targets each. The targets on a circle move alternately clockwise and counter-clockwise. 160 targets are the biggest number of targets reported up to now which are detectable by an algorithm and from which humans can select a target.

A typical smooth pursuits-based interface offers several targets to give users a choice. Esteves et al. [11] showed that it is possible to distinguish eight targets moving on a circle. Enhancements by using PCA (principal component analysis) enable up to nine targets to be distinguished [2]. Adding a second circle with targets moving in the opposite directions allows doubling this number to 18 targets.

The motivation behind our research is to further increase this number of distinguishable targets to hence enable novel applications not possible before. In particular, we envision that environments where targets can be displayed either on a single screen or be distributed in the environment can benefit from such a technology. This vision is supported by the availability of mobile eye trackers and displays to show the stimulus. Possible applications include:

MUM 2019, November 26–29, 2019, Pisa, Italy

ACM ISBN 978-1-4503-7624-2/19/11…$15.00
https://doi.org/10.1145/3365610.3365626

PursuitKeyboard A possible application is a keyboard, allowing for eye typing using smooth pursuits. Each character is represented by one target.

Smart Home Previous work investigated the use of smooth pursuits to control appliances in a smart home [31], e.g., controlling the volume of a music player. Our approach allows more targets which in turn allows controlling a variety of features for a large number of devices, such as toasters, coffee machines, vacuum cleaners, smart televisions, etc.

Virtual Reality Our approach is also valuable for complex virtual reality environment. Here, different virtual objects can be controlled using gaze.

Virtual Shopping In 2011, Tesco presented a virtual supermarket, where users chose products for home delivery from more than 100 items, using a QR code[1]. Pursuits allows selecting items via gaze in a more privacy-preserving manner.

To achieve this goal we first introduce a new detection method based on the slope of a regression line to which we refer as slope method. This method is sensitive to scale which makes it less universal than other detection methods described in [30]. In the context of eye tracking and the calibration issue, however, the differences in scale between a calibrated and an uncalibrated eye tracker signal are small. Typically these differences are smaller than factor 1.2. The advantage of the slope detection method is that it can detect pursuit movements on circles with different radii. A further advantage is that this method can distinguish many targets on one circle. Our first user study revealed that it is possible to distinguish pursuit movements for 20 targets moving on a circle with the new method. Together with circles of different sizes this allows to distinguish between far over hundred pursuit targets. As the the layout of the pursuit targets resembles old-style telephone interfaces we call this pursuit interfaces DialPlates.

As a proof of concept we implemented an interface with 160 pursuit targets (Figure 1) and as a possible application the PursuitKeyboard with 108 keys (Figure 2). In our second user study we confirm our theoretical assumptions and demonstrate the capabilities of the new detection method. Our research is complemented by a discussion of questions arising from the large number of targets, in particular, readability of target labels and layout.

2 BACKGROUND AND RELATED WORK

While early works on gaze-based interaction relied mostly on fixations, the research community started to move towards detecting gaze behavior, such as gaze gestures [10] and smooth pursuit [32, 33]. Smooth pursuit eye movements are naturally performed when gazing at a moving target. Interaction using smooth pursuit (aka Pursuits) is promising since it does not require calibration as it relies on relative eye movements rather than precise fixation points.

2.1 Applications of Pursuits

Pursuits has been utilized in several applications and domains. Being a calibration-free and contactless gaze-only modality, a large body of work investigated its use on public displays, where immediate usability is essential [1, 23]. For example, Vidal et al. used

[1]Tesco Virtual Stores: https://goo.gl/L4mEgU

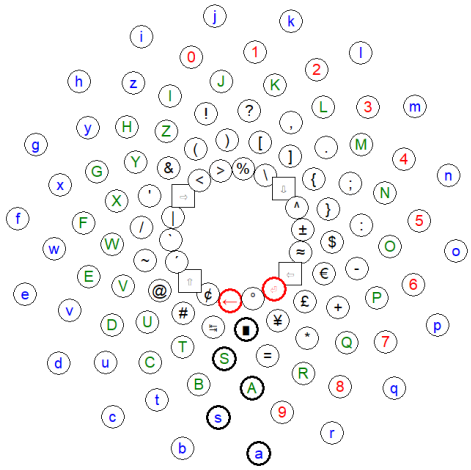

Figure 2: The PursuitKeyboard (Western variant) – 108 targets are distributed over 6 circles (18 targets each). The PursuitKeyboard is a possible application of our algorithm.

Pursuits on public displays for gaming and entertainment applications [32]. EyeVote uses Pursuits for voting on public displays [20]. Pursuits was also used in active eye tracking settings, where the tracker moved on a rail system to follow users as they pass by large displays [17]. Lutz et al. used Pursuits for entering text on public displays [22]. They worked around Pursuits' limitations by performing each letter's selection on two stages: the user first selects one of 5 groups of letters. The group then expands to allow the user to finally select the desired letter.

Further application areas of Pursuits include interaction with smart watches [11], interaction in smart homes [31], and using Pursuits for authentication [7, 27, 28]. Kangas et al. [15] and Špakov et al. [34] used Pursuits in desktop settings as a continuous signal to control an on-screen widget to, for example, adjust volume.

In addition to using it as a calibration-free gaze interaction technique, Pursuits can also be used for calibration. Pfeuffer et al. [25] introduced a method to calibrate the eye tracker as users follow on-screen moving targets. Similarly, Celebi et al. [3] used Pursuits for eye tracker calibration. Khamis et al. [19] used gradually revealing text to calibrate the eye tracker while users read-and-pursue.

Further possible applications for selection from Pursuit are AR and VR environments where the hands are used for other tasks and noise or privacy concerns does not allow speech commands. VR benefits from using Pursuits during interaction, especially when moving in VR [18], and when interacting with occluded targets [26]. Pursuits was also employed in augmented reality glasses [12].

2.2 Implementations of Pursuits

Different implementations exist to detect Pursuits. For an overview on general pursuit detection methods we refer to Velloso et al. [29] who discussed two detection methods: Euclidean distance and correlation. Herlina et al. [13] report on a comparison of both methods.

The detection methods using the Euclidean distance between the gaze estimates and target positions [15, 27, 28, 34] work very well but need a calibrated eye tracker as the method is susceptible to offsets from inaccurate detection. Methods using the standard deviation of the Euclidean distance as suggested in [34] eliminate

the offset problem and work well for linear trajectories without calibration. In the context of circular trajectories, the challenge is that the distance of one target to any other target stays constant.

The other detection method employs Pearson's product moment correlation [11, 12, 16–19, 21, 29, 31]. In contrast to the Euclidean distance, the correlation method is independent of offsets and scaling. For this reason, it works reliably without calibration [21, 29, 32, 33] and even on small interfaces such that of smart watches [11]. On the downside, the accuracy of the correlation-based detection drops significantly in the presence of more than 8 targets [11, 29, 32]. A recent suggestion for improving circular smooth pursuit detection was proposed by Velloso et al. [30]. They call their methods '2D correlation' and 'profile matching'. These methods improve pursuit detection and reduce false positives.

We learn that despite the many different application areas, interaction using Pursuits is limited in the number of distinguishable targets. This makes it difficult to apply Pursuits in scenarios, where a lot of different targets are required, such as smart homes, virtual reality, or user interfaces with many targets such as a keyboard.

3 PURSUIT DETECTION

Typical pursuit detection methods need a window size, a comparison function and a threshold. The *window size* is a time interval over which the comparison function is calculated. An eye tracker delivers data at a certain frequency and, therefore, the important value for the detection algorithm is not the time interval but the number of samples n delivered during this time interval. From a user's perspective, short time intervals are preferable, since they allow for faster selection. At the same time, shorter time intervals increase the risk of either missing the detection or getting false positives. Obviously, input devices delivering a high data rate are preferable as they enable short detection times.

The *comparison function* is a metric giving a value for how good two movements match. The detection algorithm compares every target motion with the gaze motion. Selection takes place when the value from the metric function matches a given *threshold* criteria.

3.1 Detection Using Correlation

If the eye follows a moving target, eye and target movements correlate. Hence, calculating the correlation for detection seems the obvious approach. Calculating correlation can be visualized by plotting the corresponding values, the x-coordinates of the gaze and the target, in a two-dimensional plane (Figure 3). Correlation is a measure of how close the plot is to a linear relationship. For pursuit detection, the correlation should be above a certain threshold, typically 0.8. The advantage of correlation is independence for scale and offset: Hence, this method works without calibration.

3.2 Detection Using Regression Line Analysis

If the eye tracker is perfectly calibrated and the eyes follow the target perfectly, the plot in Figure 3 is a line through the origin with slope 1.0 – in other words, gaze and target coordinates are equal. In case of a calibration error, there may be an offset between the gaze and the target coordinates' value. The slope may have a value different from 1.0. The regression line analysis approach was introduced by Drewes et al. [9] in the context of calibration.

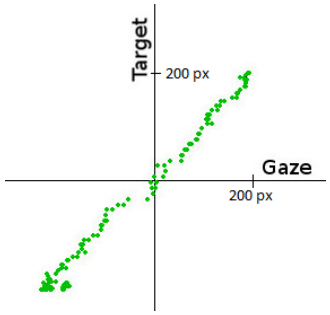

Figure 3: Real data plot for calculating correlation and for regression line analysis. Correlation is a measure of linearity and regression line analysis provides intercept and slope. In perfect calibration gaze and target coordinates are equal, meaning correlation and slope are 1.0 and the intercept is 0.

Observing data from an uncalibrated eye tracker reveals a considerable offset but the scaling error and, therefore, the error in the slope typically stays within a range from 0.8 to 1.2. Hence, it is possible to detect a smooth pursuit movement by testing how close the slope of the regression line is to 1.0. Close to 1.0 means that a threshold t_s needs to be defined with $1/t_s < \text{slope} < t_s$. From our experience, values for t_s between 1.2 and 1.3 work well. We refer to this method as the slope method.

3.3 Designing a New Pursuit Detection Algorithm

We use the following formula to calculate the correlation r:

$$r = \frac{n\sum_{i=1}^{n} gt - \sum_{i=1}^{n} g \sum_{i=1}^{n} t}{\sqrt{n\sum_{i=1}^{n} g^2 - (\sum_{i=1}^{n} g)^2}\sqrt{n\sum_{i=1}^{n} t^2 - (\sum_{i=1}^{n} t)^2}} \quad (1)$$

where g is a gaze coordinate, t the corresponding target coordinate, and n the size of the data window. The following formula calculates the slope s of the regression line:

$$s = \frac{n\sum_{i=1}^{n} gt - \sum_{i=1}^{n} g \sum_{i=1}^{n} t}{n\sum_{i=1}^{n} g^2 - (\sum_{i=1}^{n} g)^2} \quad (2)$$

which is similar to the formula for correlation. In contrast to formulas which require mean values and consequently need to sum up values over all data in the window, these formulas allow a sliding window by only subtracting an old value and adding a new value. Hence, the algorithm's run time depends on the data window size.

To avoid false positives, for a positive detection the threshold condition should not only be true for one sample but for a certain number of subsequent samples. We refer to this as minimum signal duration. This idea was already suggested by previous work [15, 34]. Reducing false positives is also possible by increasing the data window size. However, a small data window and a minimum signal duration are more efficient than a large data window with

Table 1: Parameters for correlation/slope detection methods

Parameter	Correlation Method	Slope Method
Window size	30 samples	30 samples
Smoothing	0 samples	20 samples
Minimum duration	20 samples	15 samples
Threshold	0.8	0.77 – 1.3
Skipped samples	30 samples	30 samples

the size of both, small data window and minimal signal duration. Furthermore, the gaze signal is smoothed by calculating the average over k samples. While testing our implementation, we observed that after a successful detection of a target, a false positive detection of the same target occurs sometimes. The reason is the reaction time of the user who is often still following the target after successful detection. To address this problem, we skip some samples after positive detection.

4 STUDY I: EVALUATING THE SLOPE METHOD

The purpose of the first study was to find out the maximum of distinguishable targets on a circle. To understand the strengths and weaknesses of the slope method we deliberately compared it to the classical correlation method as this method has been research intensively and well-working thresholds as well as the maximum number of distinguishable targets are known. However, we comment on other approaches, such as the rotated correlation method [30], in the theory section.

4.1 Comparing both Methods

Both detection methods depend on several parameters – the threshold, the data window size, the minimum signal duration, and the smoothing window size. A systematic approach with five different values for each parameter would have led to 625 combinations for each detection method. Note, that additionally, the target speed as well as the radius of the circle on which the targets are moving might influence the results. A systematic approach that tests all possible combinations is not feasible. Simulations do not work either as there is a feedback loop with the user's eye.

As a solution, we decided to optimize parameters for each method individually. We use the same correlation value of 0.8 and a data window size of 30 samples as Vidal et al. [32]. However, smoothing the gaze signal improves the detection with the slope method but strongly increased the false positive rate for the correlation method. The same is true for setting the minimum signal detection. Table 1 shows the parameters for both detection methods as used for the user study. We used an eye tracker which delivered 60 samples per second. All numbers in the table are samples except the thresholds which are real numbers.

4.2 Apparatus

To evaluate our Pursuits detection approach, we developed a sample application (Figure 4) in which users can enter digits (0 to 9) and letters (A to N) via Pursuits.

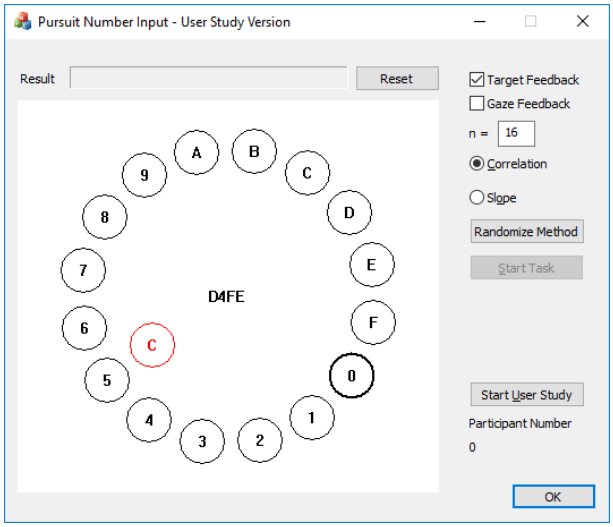

Figure 4: User interface for our study: participants had to enter a four-digit number via selection from up to 24 targets.

The application runs on an gaming laptop with integrated Tobii IS4 Base AC eye tracker (60 Hz). The display has a resolution of 1920 times 1080 pixels on 38.4 cm times 21.7 cm, which results in 0.2 mm for one pixel or 50 px per centimeter. The average distance between the participants' eyes to the display is around 50 cm +/- 5 cm, which corresponds to 0.02° per pixel or around 50 px per degree. The targets move clockwise on a circle with a radius of 130 px (2.6°), except for the 'cancel'-target which moves counter-clockwise on a circle with a radius of 80 px (1.6°). The radius of each target is 20 px (0.4°) and they move at 6.5°/s (2.5 seconds per rotation).

The interface provides visual and acoustic feedback for detection. Every target that matches the threshold condition is filled with color, whose intensity increases the longer the threshold condition stays true, and reaches its maximum once the minimum signal duration is reached. Different beeps represent correct and wrong entries.

4.3 Procedure

We invited 16 participants (3 females) with normal (7) or corrected to normal vision (9) aged between 24 and 58. After arriving at the lab, participants filled out a form with the demographic data and received a short introduction to the system (Figure 4). To test how well the methods work for spontaneous gaze interaction, we did not calibrate the eye tracker for each participant. Instead, it was calibrated only once by one of the authors. The participants' task, inspired by a PIN entry task [7], was to enter a four-digit number by following the clockwise rotating number targets using gaze. In case of entering a wrong digit, the participants had to delete it by selecting the counter-clockwise rotating 'cancel'-target. Each canceled entry was counted as an error.

Participants first completed a training task with six targets in which they entered four symbols (digits and letters), and tried to cancel an entry. These entries were excluded from the analysis. We started the study with six targets and the participants had to enter 4 symbols using each Pursuits detection method. We randomized the method with which they started. After a successful round we

increased the number of targets by two and randomized again with which method to start. Every selection task had a timeout of 90 seconds. If a participant was not able to fulfill the task in time or wished to abort, the study continued with the other method until the maximum target number of 24 was reached or the participant failed. We concluded with a semi-structured interview.

4.4 Results

Apart from the qualitative feedback and observations, we logged the *maximum number of targets* shown simultaneously from which participants could still perform successful selections. We further logged the *errors*, which correspond to the number of times users canceled their input. We also logged the *task completion time*, which denotes the time taken to enter all 4 symbols correctly. Finally we logged the average *entry time* for entering each symbol.

4.4.1 Interviews and Observations. All participants understood immediately how to operate the system and how to enter the digits, but it seemed that they were at the beginning of a steep learning curve. Many saw the user study like a computer game and were highly ambitious to reach a high score. All participants reported that the task required a lot of focusing. All participants reported that they found the slope-based method more accurate and easier. Some of them even mentioned their preference before being asked.

4.4.2 Maximum Number of Targets. We counted the maximum number of displayed targets from which participants were able to enter the four symbols (Figure 5). The slope detection approach outperformed the correlation detection method. Only one participant could select more targets with the latter.

A Wilcoxcon signed ranked test revealed that the slope detection method results in a significant increase in the number of displayed targets from which participants successfully made selections ($Z = 3.168, p < 0.01$). Using the correlation method, the maximum target number for which participants accomplished the task was between 10 and 24 ($M = 15.0$, $SD = 3.7$). Using the slope method, the maximum target number was between 8 and 24 ($M = 21.6$, $SD = 4.6$). Note, that in our implementation, the correlation method performs even better than in previous work [32].

4.4.3 Errors. Whenever participants entered a wrong digit, they had to cancel the entry by selecting the 'cancel' target. Every entry of the 'cancel' target was counted as error. The average number of errors increases in the presence of more targets (Figure 6). The increase in errors is higher for the correlation method. For example, while both methods yielded almost no errors at 6 targets across all participants, the mean number of errors at 8 targets was 1.25 and 0.13 for the correlation and slope methods respectively. Similarly, at 24 targets, participants made 22 errors on average in case of correlation, but only 3 errors on average in case of the slope method. Note, that Figure 6 displays an average over the participants who were successful in the respective conditions.

4.4.4 Task Completion Time. We measured the completion time for successfully entering 4 symbols, starting from the moment displaying the symbols, until the moment the fourth symbol was entered. This also includes cancellations. The average completion time is similar for both methods for up to 8 targets (Figure 5).

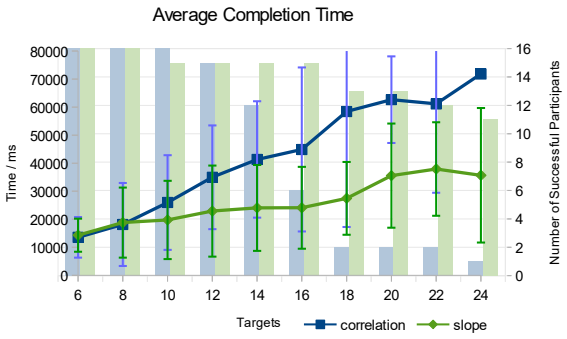

Figure 5: Completion time over number of targets. The slope method was consistently faster than the correlation method. The bars in the background indicate the number of participants who successfully completed the task.

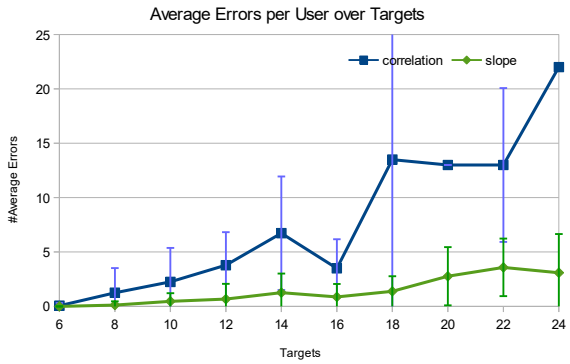

Figure 6: Errors over number of targets. User made consistently less errors with the slope method.

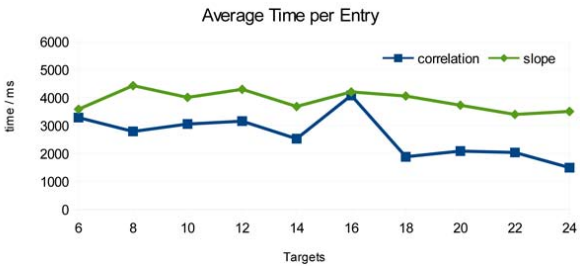

Figure 7: Time per Entry. Participants performed slightly faster on a single entry with the correlation method. It seems the correlation method is more sensitive but also detects more false positives.

For more targets, it increases strongly for the correlation method compared to the slope method. Similar to the errors, successful completion times exclude cases where participants failed to enter the 4 symbols. Completion times are longer for the correlation method. This is mainly due to the many cancellations participants had to perform.

4.4.5 Symbol Entry Time. Figure 7 shows the average time for selecting a single entry. The slight decrease in selection times is likely the results of a learning effect. Furthermore, only successful participants contributed to the entry times for larger target numbers. They can be assumed to be performing better overall.

One interesting observation is that the time per entry does not increase with the number of targets. The other interesting observation is that the times for the slope method are higher than the times for the correlation method. This is remarkable as the slope detection uses a shorter minimum signal duration. However, a Wilcoxon signed rank test showed no evidence of significant effects of detection method on entry time.

4.4.6 Robustness of the Methods. Robustness refers to less detected false positives. False positive can occur in two situations. First, the user wants to select a certain target but the detection reports another target (i.e. an error). The data show that the slope method performs better than the correlation method if there are more than 6 targets. For 8 targets, the correlation method leads to 15% false positives (cf. 12% in [32]), but only 1.6% with the slope method.

Second, a user does not intend to enter any target but the eye movements trigger a detection (i.e. an unintended selection). This can be avoided by turning off detection when no entries are expected (e.g., when another window is active on a desktop). We tested how both algorithms respond to cases, in which this is not possible. To do so, we recorded gaze activity for about 3 minutes from other tasks such as reading, surfing the internet, watching a video. Feeding the data to the detection algorithms, we found false positive rates from 0.05 to 0.42 per second. We did not find any pattern, suggesting that the issue of unintended selection should be investigated as future work.

5 TOWARDS MAXIMIZING TARGET NUMBERS

In the following section we provide the theoretical foundations for why the introduced method allows to build interfaces with a large numbers of targets.

5.1 Theoretical Foundations

The study yielded significant differences in both detection methods. Beside the experimental validation these findings can be explained theoretically. The explanation applies to all smooth pursuit interfaces with targets moving on circles. The coordinates of targets on a circle can be expressed by

$$x_i = A_i cos(\omega_i t + \phi_i) \quad \text{and} \quad y_i = A_i sin(\omega_i t + \phi_i) \tag{3}$$

Assuming a perfectly calibrated and accurate eye tracker, and a user whose gaze follows exactly a target, the gaze coordinates will be exactly these target coordinates. In the following we use this idealized assumption to understand why the slope method has the capability to distinguish between more targets than the correlation method does.

If there are n targets on the circle and the gaze follows a target exactly, the coordinates of the previous and next target are phase shifted by $\pm 2\pi/n$ against the gaze coordinates. The situation for $n = 20$ is depicted in Figure 8. The gray area in the figure indicates

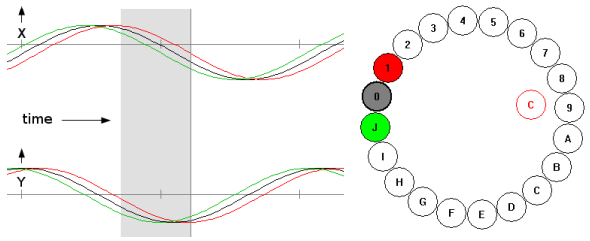

Figure 8: The right picture shows the pursuit targets. The previous (red) and the next (green) targets are phase shifted by $\pm 2\pi/20$ against the gaze (black). The left side shows the x- any y-coordinate over time for the red, black and green target. The gray area is the current data window for detection.

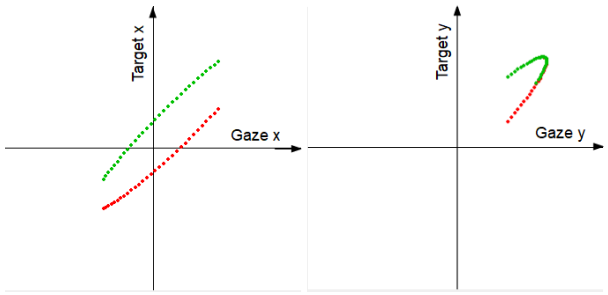

Figure 9: We assume that the eye perfectly follows the black target. We compare the gaze coordinates with the previous and next target by regression analysis. The figure shows the plot for the x-coordinate (left) and the y-coordinate (right). The red data points lie on a nearly perfect line for both coordinates and produce a positive detection signal for the red target. The green data points for the y-coordinates are not on a line and therefore do not signal a detection.

the current data window. Figure 9 shows the regression analysis for the data window in Figure 8.

All points lie on Lissajous curves, which were studied in the early 19th century. Psychology research on smooth pursuits used Lissajous curves already 20 years ago [8][2].

As all targets move with the same speed on the same circle, A_i and ω_i have the same value for each i and the Lissajous curve has the shape of an ellipsis. The phase shift affects the eccentricity; the smaller the phase shift the closer the shape is to a diagonal line. The data window size determines the fraction of the ellipsis on which the data points lie. If the data window covers a full cycle, the data points cover the ellipsis completely. In this case, the slope of the regression line and the correlation will be constant over time.

With a smaller data window (as in Figure 8), which is desirable for a short detection time, the data points fill only a part of the ellipsis (Figure 9), moving over time. At the time shown here for the x-values, the data points are on an almost straight line and the correlation and the slope are close to 1.0. At the same time, the y-values for the green target fill the ellipsis' tip and the slope and correlation are far from 1.0. As the threshold condition has

[2]This should not be confused with target movements on Lissajous curves, as used in [24].

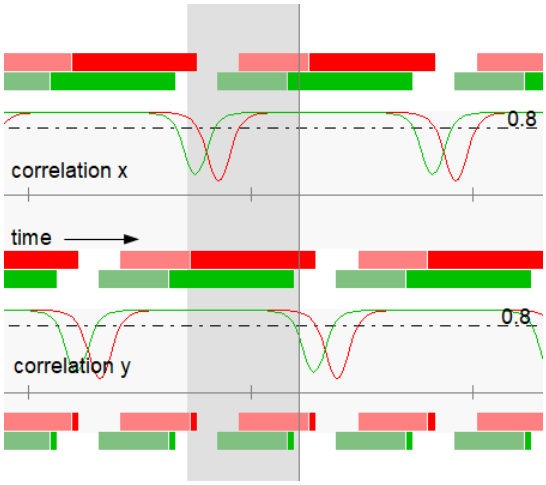

Figure 10: Correlation values for the example in Figure 8. The bars indicate a true threshold condition for x (up), y (middle) and both (down). The light color in the bars indicate the minimum signal duration. The data for the previous target are red and for the next target are green.

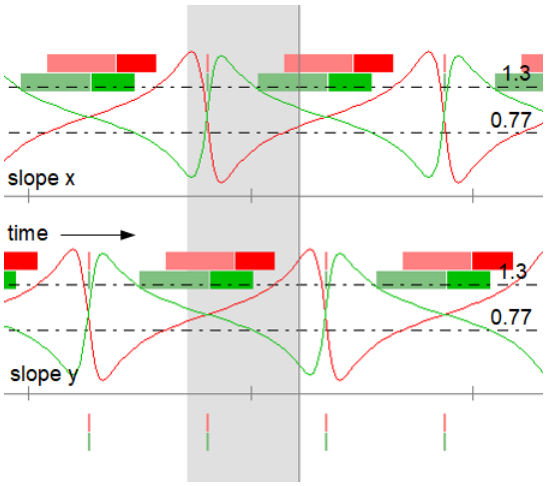

Figure 11: Slope values for the example in Figure 8. The bars have the same meaning as explained in Figure 10.

to be true for the x- and y-coordinate, this means that there is no positive detection for the green target at that moment. However, the y-values of the red target are still on a straight line and the detection algorithm reports a positive detection for the red target.

Figure 10 shows the correlation values for the given example and Figure 11 shows the values for the slope from the linear regression. The dash-dotted line indicates the thresholds and the bars indicate whether the threshold condition is true. The bars have light color before the minimum signal duration is reached. The lowest bars indicate whether both threshold conditions are true.

The correlation is close to 1.0 most of the time and satisfies the threshold condition (Figure 10). This is a reason why the strategy of choosing the target with the highest correlation does not work reliably. The correlation value drops, when the data window covers the ellipsis' tip. As the threshold condition has to be true for the x and the y coordinate, the correlation method signals detection between both drops.

If using the rotated correlation (cf. [30]), the correlation signal does not drop but stays constantly on a value close to 1. The rotated correlation needs a threshold of 0.97 to distinguish 20 targets on a circle with the synthetic perfect gaze signal. However, such a high value will not work reliably on a real gaze signal with noise.

The slope values pass the threshold interval quickly and satisfies the threshold condition for a shorter time (the bars are shorter in Figure 11). The overlap of both signals for the x- and y-coordinate is shorter. Together with the concept of minimum signal duration, the slope method does not report false positives for this example while the correlation method does. The shorter time fulfilling the threshold condition is the reason why the slope method can distinguish more targets on a circle. This also means also that the correlation method detects more easily and more quickly (but at the expense of more false positives). This could explain, why the entry time for the correlation method is slightly shorter (Figure 7).

The theoretical considerations given here provide hints for further improvements of the slope detection method. When the data window passes the the ellipse's tip the calculated slope traverses the threshold interval and there is also a moment where it may be infinite, i. e. vertical. The passing of the threshold interval happens very quickly and, therefore, this false positive detection is filtered by the minimum signal duration. However, it is possible to improve the detection by demanding that the correlation has to be above a threshold before calculating the slope. The correlation value drops when the data window passes the ellipsis' tip.

5.2 Target Detection on Circles of Different Size

The advantage of the introduced slope method is that it works without calibration but in contrast to the correlation method it allows to distinguish between targets moving on circles with different radius. Figure 12 shows a regression line analysis for three different targets on circles with different radii (50 px, 100 px. 150 px). Targets move with the same angular velocity and phase. As the eyes follow each target, the slope varies (0.5 for 50 px, 1.0 for 100 px, and 1.5 for 150 px). While the correlation detects all targets, it cannot distinguish them. In contrast, the slope method can do so, as the slope for each target is inside a certain threshold interval.

5.3 The Maximum Target Number

The correlation method can distinguish targets moving clockwise and counter-clockwise but not between circles with different radius. Literature reports 8–9 distinguishable targets [2, 11], limiting the correlation method to 16–18 targets.

As shown above, one advantage of the slope method is that it can distinguish around 20 different targets on one circle. This number can be further increased by using different radii.

Distinguishing targets on circles with different radii, means different values for A_I. The slope method works well as long as the targets move with the same angular velocity, i.e. all ω_i have the same value. In this case the corresponding Lissajous curves are

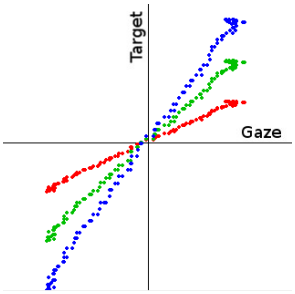

Figure 12: Regression line analysis for targets on three circles with diameter 100 px (red), 200 px (green), 300 px (blue). The plot shows recorded data from a calibrated eye tracker. The eye followed the green target and consequently the green dots lie on line with slope 1. The slope method detects the green target. As all targets create dots on a line the correlation method detects them all and is not able to distinguish.

ellipses. As targets have the same angular velocity, targets on large circles move faster than on small circles. While the angular velocity of targets on a large circle could be decreased, this creates Lissajous curves with lobes, making detection more complex.

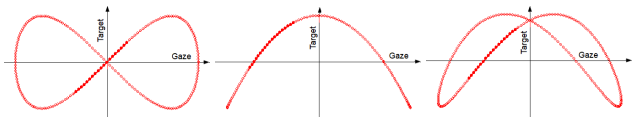

Figure 13: Lissajous curves for a target which moves on a circle with half radius and double angular velocity than the target followed by the eyes. The targets have a phase shift of 0°, 90°, and 120° (from left to right). The dots represent samples and the filled dots the samples in a data window which fulfills the threshold condition for both detection methods.

Arbitrarily selecting radius and angular velocity may lead to similar correlation and slope values, hence producing false positives. One case is depicted in Figure 13, where targets move on a circle with half radius, double angular velocity, and a phase shift of 0°, 90°, and 120° against the target followed by the eyes. Here, correlation and slope are both close to 1.0. The challenge is to find combinations of ratio of angular velocities, ratio of radii, threshold data window size, and signal duration which allow targets on different circles with different angular velocity to be reliably detected.

From previous work [11, 13, 34] and our own experience, we learn that the speed of smooth pursuits targets should be in a range from 5°/s to 20°/s. To ensure that targets move at the same angular speed, this means the maximum ratio between the smallest and largest radius can be 4. The slope threshold value t_s determines the minimum ratio of two different radii. For reliable detection, the ratio of the radii has to be larger than t_s * t_s. With t_s = 1.2 (a slope value that worked well according to our experience), the ratio of to radii should be larger than 1.44. As 1.44^4 = 4.3, this means that four circles with targets for each direction should be possible. In total, the slope method should be able to distinguish between 8 * 20 = 160 targets (cf. our implementation in Figure 1).

Note, that this is no hard value as target speed range, threshold value, and number of targets on a circle could still be varied. Increasing the number of targets leads to more false positives and a higher percentage of people who are not able to select correct targets. A lower number of targets increases robustness by decreasing the false positive rate.

6 STUDY II: EXPLORING MULTI-TARGET UIS

In the following we report on an exploration of interfaces employing large numbers of targets. In particular, we are interested in people's ability to use these interfaces.

6.1 Prototypes

We implemented two types of interfaces. Firstly, we implemented a 72-target and a 160-target interface to investigate selection speed and errors. Secondly, as a proof-of-concept, we built a pursuits-based keyboard. Suitable radii and target speeds for both interfaces were determined in pre-tests.

6.1.1 72/160-Target Interface. The prototype is depicted in Figure 1. The radius of the largest target circle was 424 px (8.5° visual angle) and the smallest 110 px (2.2°) at a distance of 50 cm from eyes to screen. The target speed was 14°/s on the largest and 3.7°/s on the smallest circle. The speed of the targets was between 9°/s – 2.3°/s.

6.1.2 PursuitKeyboard. As an application, we implemented a keyboard based on pursuits (Figure 2). Our layout is based on a standard keyboard using 102 letters. To avoid using caps-locks (which would require multiple selections), we assigned each key to a distinct target. As for the layout, we show characters in alphabetical order. We both show small and capital letters. To allow for more easily distinguishing between keys and special characters, we show capital letters in green color, small letters in blue, digits in red, and special characters in black. All targets have a round shape except for the four navigation keys which are square-shaped. We chose a geometry with 108 targets consisting of six circles with 18 targets each. The outer circle had a radius of 300 px and 90 px for the inner circle. The speed on the outer circle was 10°/s and 3°/s for the inner circle. Note, that future work could optimize the layout to maximize input times as well as on improving the visual appearance.

6.2 Study Design & Tasks

The study followed a within subject design. The independent variables were three interface layouts (72-targets, 160-targets, PursuitsKeyboard) and two interface sizes:

72-target interface Targets were arranged on 6 circles with 12 targets each (ratio of circle radii: 1.3).

160-target interface (large) Targets were arranged on 8 circles with 20 targets each (ratio of circle radii: 1.2).

160-target interface (small) We used the same layout for this condition, but the geometry was scaled by 0.5. Both 160-target UIs used the same angular speed.

For these condition, users had to select 6 targets. Note, that finding the correct target was not part of this task. We measured input speed and errors. Participants were not asked to make corrections.

PursuitKeyboard We used the keyboard depicted in Figure 2 with 108 targets.

Participants had to enter the phrase '50% Chance!'. This phrase was chosen as it has at least one character on each ring. Note that participants had to locate the correct character. Wrong entries had to be corrected by using the backspace target.

6.3 Procedure

We used the same setup as in the first study. After the participants signed the consent form they were introduced to the system and completed a sample task. Then, participants completed the aforementioned tasks in counterbalanced order. After the study, we conducted interviews with participants and asked them about their experience while using the interface.

6.4 Evaluation

Eleven users (22–39 years, 3 female) participated in our study. Five of them wore glasses. All participants were able to complete task 1 and 2There were three participants who were not able to complete task three and four but it was only one person who could not complete both. However, all participants who failed to complete a task did not abort but completed the task for more than 80%.

Table 2 summarizes the descriptive results of the study. Selections require between 6 and 8 seconds. Interestingly, both time and number of errors seem to increase for larger interfaces. Locating the correct character in the PursuitKeyboard increases both selection time (10 s) and errors. The rather high standard deviation suggests that for some people it is easier to operate interfaces with high target numbers instantly while others require more training.

Most participants stated that the task was challenging, some reported 'tired eyes', and some complained about the target speed. Participants felt that in particular in situation where movement of arms and fingers is restricted, this is a useful approach (e.g., being disabled, holding something in the hands).

Table 2: Avg. completion time per target and std. dev., the median of errors, and the percentage of successful participants for the four tasks.

	Av. Time per Target	Std. Dev.	Med. errors	% successful participants
72-targets	5.8 s	4.2 s	1	100%
160-targets (large)	7.8 s	4.8 s	1	100%
160-targets (small)	6.1 s	3.5 s	2.5	73%
PursuitKeyboard	10.0 s	3.8 s	4.5	73%

7 DISCUSSION & FUTURE WORK

In the following we summarize and discuss findings, that open interesting directions for future research.

One vs. multiple screens. Our approach is applicable both in settings where all targets are shown on one screen and in settings where targets are assigned to different screens (e.g. on home appliances). In the former case, locating the correct target from many small, co-located targets may be challenging (for example a smart watch to control several home appliances). In contrast, if targets are attached to specific devices it becomes both easier to select them as well as to relate them to a certain devices or function. Future work could investigate the learning effect for such interfaces.

Midas touch. Another challenge arises as users are required to read labels. While doing so, users may accidentally select an object (cf. the Midas touch). Visual hints, such as target color and shape, may help mitigating this challenge. This should be investigated more closely in future work.

Eye Typing. Future work could further investigate eye typing. The Pursuit keyboard is useful for disabled people who depend on gaze-only text-entry. There are existing solutions, such as 'dasher' [35]. Yet, this approach requires calibration of the eye tracker. Future work could compare 'dasher' and 'SMOOVS' [22] with our approach in terms of input speed, error rates and user acceptance.

Robustness. The robustness of Pursuits against false positives is still an open question for all pursuits detection methods. Our experimental data and theoretical considerations explain the occurrence of false positives during target selection. However, false positives triggered by other gaze activities are not yet fully understood. Our data suggests that the detection robustness depends on the type of other gaze activities. Velloso et al. [30] recently reported on increased robustness in the context of another detection method.

Speed vs. angular speed. It is yet unclear, whether speed or angular speed is more important on how humans perceive target speed on circular trajectories. This, as well as an answer to how users' ability to recognize target labels is affected by different target speeds, need further investigation.

Interplay of number of targets, input speed, error rate, and user acceptance. The interplay of targets and their properties needs further investigation. For example, it is unclear, whether a lower number of targets on more circles leads to a better user experience, compared to fewer circles with more targets. Also it is unclear how the variation of parameters such as the threshold interval or the minimum signal duration affect false positives or the ease of use.

Scenarios with mobile eye trackers. When using our approach with a mobile eye tracker, coordinates from the mobile eye tracker need to be transformed to the coordinates of the display presenting the pursuit targets. For this the corners of the display have to be detected in the mobile eye tracker's world view. For a sufficient accuracy in the transformation the mobile eye tracker should be not to far from the display.

Further Application Areas. Directions for future work also include testing the the slope method in specific application scenarios and with other eye trackers. Researchers could investigate, how quickly users adapt to such interfaces and whether the need to strongly focus on the target decreases over time. Furthermore, researchers and practitioners could apply and evaluate the slope method beyond gaze, e.g. motion matching for body movements [4–6] and mid-air gestures [2].

8 CONCLUSION

We introduced a new pursuits detection approach – the *slope method* – which, although sensitive to scaling, works without calibration and can distinguish targets on circles of different size. We demonstrated that it can distinguish up to 160 simultaneously moving targets in a desktop setting. We provided theory and a proof-of-concept, showing that selection from a high number of targets is feasible.

REFERENCES

[1] Florian Alt, Stefan Schneegaß, Albrecht Schmidt, Jörg Müller, and Nemanja Memarovic. 2012. How to Evaluate Public Displays. In *Proceedings of the 2012 International Symposium on Pervasive Displays (PerDis '12)*. ACM, New York, NY, USA, Article 17, 6 pages. https://doi.org/10.1145/2307798.2307815

[2] Marcus Carter, Eduardo Velloso, John Downs, Abigail Sellen, Kenton O'Hara, and Frank Vetere. 2016. PathSync: Multi-User Gestural Interaction with Touchless Rhythmic Path Mimicry. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3415–3427. https://doi.org/10.1145/2858036.2858284

[3] Feridun M. Celebi, Elizabeth S. Kim, Quan Wang, Carla A. Wall, and Frederick Shic. 2014. A Smooth Pursuit Calibration Technique. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. ACM, New York, NY, USA, 377–378. https://doi.org/10.1145/2578153.2583042

[4] Christopher Clarke, Alessio Bellino, Augusto Esteves, and Hans Gellersen. 2017. Remote Control by Body Movement in Synchrony with Orbiting Widgets: An Evaluation of TraceMatch. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 45 (Sept. 2017), 22 pages. https://doi.org/10.1145/3130910

[5] Christopher Clarke, Alessio Bellino, Augusto Esteves, Eduardo Velloso, and Hans Gellersen. 2016. TraceMatch: A Computer Vision Technique for User Input by Tracing of Animated Controls. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*. ACM, New York, NY, USA, 298–303. https://doi.org/10.1145/2971648.2971714

[6] Christopher Clarke and Hans Gellersen. 2017. MatchPoint: Spontaneous Spatial Coupling of Body Movement for Touchless Pointing. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 179–192. https://doi.org/10.1145/3126594.3126626

[7] Dietlind Helene Cymek, Antje Christine Venjakob, Stefan Ruff, Otto Hans-Martin Lutz, Simon Hofmann, and Matthias Roetting. 2014. Entering PIN codes by smooth pursuit eye movements. *Journal of Eye Movement Research* 7, 4 (2014). https://bop.unibe.ch/index.php/JEMR/article/view/2384

[8] Claudio de'Sperati and Paolo Viviani. 1997. The Relationship between Curvature and Velocity in Two-Dimensional Smooth Pursuit Eye Movements. *Journal of Neuroscience* 17, 10 (1997), 3932–3945. https://doi.org/10.1523/JNEUROSCI.17-10-03932.1997 arXiv:http://www.jneurosci.org/content/17/10/3932.full.pdf

[9] Heiko Drewes, Ken Pfeuffer, and Florian Alt. 2019. Time- and Space-efficient Eye Tracker Calibration. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications (ETRA '19)*. ACM, New York, NY, USA, Article 7, 8 pages. https://doi.org/10.1145/3314111.3319818

[10] Heiko Drewes and Albrecht Schmidt. 2007. Interacting with the Computer Using Gaze Gestures. In *Proceedings of the 11th IFIP TC 13 International Conference on Human-computer Interaction - Volume Part II (INTERACT'07)*. Springer-Verlag, Berlin, Heidelberg, 475–488. http://dl.acm.org/citation.cfm?id=1778331.1778385

[11] 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)*. ACM, New York, NY, USA, 457–466. https://doi.org/10.1145/2807442.2807499

[12] Augusto Esteves, David Verweij, Liza Suraiya, Rasel Islam, Youryang Lee, and Ian Oakley. 2017. SmoothMoves: Smooth Pursuits Head Movements for Augmented Reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 167–178. https://doi.org/10.1145/3126594.3126616

[13] Herlina, Sunu Wibirama, and Igi Ardiyanto. 2018. Similarity measures of object selection in interactive applications based on smooth pursuit eye movements. In *2018 International Conference on Information and Communications Technology (ICOIACT)*. 639–644. https://doi.org/10.1109/ICOIACT.2018.8350701

[14] 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

[15] Jari Kangas, Oleg Špakov, Poika Isokoski, Deepak Akkil, Jussi Rantala, and Roope Raisamo. 2016. Feedback for Smooth Pursuit Gaze Tracking Based Control. In *Proceedings of the 7th Augmented Human International Conference 2016 (AH '16)*. ACM, New York, NY, USA, Article 6, 6:1–6:8 pages. https://doi.org/10.1145/2875194.2875209

[16] Mohamed Khamis, Florian Alt, and Andreas Bulling. 2015. A Field Study on Spontaneous Gaze-based Interaction with a Public Display Using Pursuits. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*. ACM, New York, NY, USA, 863–872. https://doi.org/10.1145/2800835.2804335

[17] Mohamed Khamis, Axel Hoesl, Alexander Klimczak, Martin Reiss, Florian Alt, and Andreas Bulling. 2017. EyeScout: Active Eye Tracking for Position and Movement Independent Gaze Interaction with Large Public Displays. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 155–166. https://doi.org/10.1145/3126594.3126630

[18] Mohamed Khamis, Carl Oechsner, Florian Alt, and Andreas Bulling. 2018. VR-Pursuits: Interaction in Virtual Reality using Smooth Pursuit Eye Movements. In *Proceedings of the 2018 International Conference on Advanced Visual Interfaces (AVI '18)*. ACM, New York, NY, USA, 7.

[19] 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 *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16)*. ACM, New York, NY, USA, 274–285. https://doi.org/10.1145/2971648.2971679

[20] Mohamed Khamis, Ludwig Trotter, Markus Tessmann, Christina Dannhart, Andreas Bulling, and Florian Alt. 2016. EyeVote in the Wild: Do Users Bother Correcting System Errors on Public Displays?. In *Proceedings of the 15th International Conference on Mobile and Ubiquitous Multimedia (MUM '16)*. ACM, New York, NY, USA, 57–62. https://doi.org/10.1145/3012709.3012743

[21] Mohamed Khamis, Ludwig Trotter, Markus Tessmann, Christina Dannhart, Andreas Bulling, and Florian Alt. 2016. EyeVote in the Wild: Do Users Bother Correcting System Errors on Public Displays?. In *Proceedings of the 15th International Conference on Mobile and Ubiquitous Multimedia (MUM '16)*. ACM, New York, NY, USA, 57–62. https://doi.org/10.1145/3012709.3012743

[22] 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).

[23] Jörg Müller, Florian Alt, Daniel Michelis, and Albrecht Schmidt. 2010. Requirements and Design Space for Interactive Public Displays. In *Proceedings of the 18th ACM International Conference on Multimedia (MM '10)*. ACM, New York, NY, USA, 1285–1294. https://doi.org/10.1145/1873951.1874203

[24] Andriy Pavlovych and Carl Gutwin. 2012. Assessing Target Acquisition and Tracking Performance for Complex Moving Targets in the Presence of Latency and Jitter. In *Proceedings of Graphics Interface 2012 (GI '12)*. Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 109–116. http://dl.acm.org/citation.cfm?id=2305276.2305295

[25] Ken Pfeuffer, Melodie Vidal, Jayson Turner, Andreas Bulling, and Hans Gellersen. 2013. Pursuit Calibration: Making Gaze Calibration Less Tedious and More Flexible. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 261–270. https://doi.org/10.1145/2501988.2501998

[26] Thammathip Piumsomboon, Gun Lee, Robert W. Lindeman, and Mark Billinghurst. 2017. Exploring natural eye-gaze-based interaction for immersive virtual reality. In *2017 IEEE Symposium on 3D User Interfaces (3DUI)*. 36–39. https://doi.org/10.1109/3DUI.2017.7893315

[27] Vijay Rajanna, Adil Hamid Malla, Rahul Ashok Bhagat, and Tracy Hammond. 2018. DyGazePass: A gaze gesture-based dynamic authentication system to counter shoulder surfing and video analysis attacks. In *2018 IEEE 4th International Conference on Identity, Security, and Behavior Analysis (ISBA)*. 1–8. https://doi.org/10.1109/ISBA.2018.8311458

[28] Vijay Rajanna, Seth Polsley, Paul Taele, and Tracy Hammond. 2017. A Gaze Gesture-Based User Authentication System to Counter Shoulder-Surfing Attacks. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 1978–1986. https://doi.org/10.1145/3027063.3053070

[29] Eduardo Velloso, Marcus Carter, Joshua Newn, Augusto Esteves, Christopher Clarke, and Hans Gellersen. 2017. Motion Correlation: Selecting Objects by Matching Their Movement. *ACM Trans. Comput.-Hum. Interact.* 24, 3, Article 22 (April 2017), 35 pages. https://doi.org/10.1145/3064937

[30] Eduardo Velloso, Flavio Luiz Coutinho, Andrew Kurauchi, and Carlos H Morimoto. 2018. Circular Orbits Detection for Gaze Interaction Using 2D Correlation and Profile Matching Algorithms. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18)*. ACM, New York, NY, USA, Article 25, 9 pages. https://doi.org/10.1145/3204493.3204524

[31] Eduardo Velloso, Markus Wirth, Christian Weichel, Augusto Esteves, and Hans Gellersen. 2016. AmbiGaze: Direct Control of Ambient Devices by Gaze. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 812–817. https://doi.org/10.1145/2901790.2901867

[32] 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

[33] Mélodie Vidal, Ken Pfeuffer, Andreas Bulling, and Hans W. Gellersen. 2013. Pursuits: Eye-based Interaction with Moving Targets. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 3147–3150. https://doi.org/10.1145/2468356.2479632

[34] Oleg Špakov, Poika Isokoski, Jari Kangas, Deepak Akkil, and Päivi Majaranta. 2016. PursuitAdjuster: An Exploration into the Design Space of Smooth Pursuit –based Widgets. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA '16)*. ACM, New York, NY, USA, 287–290. https://doi.org/10.1145/2857491.2857526

[35] D. J. Ward and D. J. C. MacKay. 2002. Fast Hands-free writing by Gaze Direction. *Nature* 418, 6900 (2002), 838.