Toward Everyday Gaze Input: Accuracy and Precision of Eye Tracking and Implications for Design

Anna Maria Feit[1,2] **Shane Williams**[2] **Arturo Toledo**[2,3] **Ann Paradiso**[2]
Harish Kulkarni[2] **Shaun Kane**[2,4] **Meredith Ringel Morris**[2]
[1]Aalto University [2]Microsoft Research [3]Toledo Design [4]University of Colorado

ABSTRACT
For eye tracking to become a ubiquitous part of our everyday interaction with computers, we first need to understand its limitations outside rigorously controlled labs, and develop robust applications that can be used by a broad range of users and in various environments. Toward this end, we collected eye tracking data from 80 people in a calibration-style task, using two different trackers in two lighting conditions. We found that accuracy and precision can vary between users and targets more than six-fold, and report on differences between lighting, trackers, and screen regions. We show how such data can be used to determine appropriate target sizes and to optimize the parameters of commonly used filters. We conclude with design recommendations and examples how our findings and methodology can inform the design of error-aware adaptive applications.

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation : Misc.

Author Keywords
Eye tracking; gaze filters; sensor noise; adaptive interfaces.

INTRODUCTION
Eye tracking has become an easily available and relatively cheap modality to enhance interaction with a computer. Remote (as opposed to head-mounted) video-based eye trackers are available for under $200 [43, 45]. They are unobtrusive and can be mounted on a display or built into laptops [28]. As a result, there is increasing interest in using information about eye gaze not only as a research instrument, but to enhance our everyday interaction with computers. Gaze-enabled applications and interaction techniques range from explicit gaze input, such as pointing [22, 49] or gaze gestures [8, 16, 33] to attentive applications that use gaze to make inferences about the user's intentions [13, 29, 38] and improve input with other modalities [37].

Surprisingly, little work has been done to understand the requirements of such applications as they integrate into our everyday computer use. There is no common standard or framework for designers and developers to rely on to develop easy-to-use and robust gaze-enabled applications. As when designing applications for other input modalities, such as touch or mouse, they have to face basic questions such as:
Which region of the screen is easiest to interact with? How accurate can we expect the user's input to be? What is the minimum usable size of an interactive element?

Prior work has found that tracking quality deviates from the numbers obtained in manufacturers' labs [5, 11, 30, 31] and that tracking *accuracy* (the offset from the true gaze point) and *precision* (the spread of the gaze points) vary widely across different tracking conditions and users. However, few efforts have been made to draw any formal conclusions that could inform the design of gaze-enabled applications; if so, they concentrate on specific applications or interaction techniques (e.g. [18, 32]) Research studies typically try to keep variation as small as possible by carefully controlling tracking conditions, user positioning, frequently recalibrating the eye tracker, and excluding participants that do not track well. While this is reasonable for research, it is not feasible in practice.

To illustrate the extent of this problem in everyday use of gaze input, we conducted an informal survey among five expert users of gaze applications (people with ALS who rely on gaze-based AAC for communication). We found that they have to recalibrate their eye tracker between three and ten times per day, even though devices are mounted to their wheelchairs and their movement in front of the screen is limited due to their motor disabilities. Despite this frequent recalibration, they reported that they typically fail to interact with an application several times per day as the application cannot account for differences in tracking quality.

To establish eye gaze as part of our everyday interaction with computers, we need to understand the characteristics and limitations of eye tracking in practice and derive standards for the design of gaze-enabled applications that take into account that accuracy and precision can vary widely across different tracking conditions. Toward this end, we collected calibration-style eye tracking data from 80 participants in a controlled but practical setup, using two different trackers in two lighting conditions. In contrast to many eye tracking studies, we did not exclude any participant due to insufficient tracking quality and only calibrated once at the beginning. Our findings offer several contributions for the design and development of gaze-enabled applications:

1. We report **accuracy and precision ranges** overall and for different parts of the screen that characterize the large variation across different tracking conditions.

CHI 2017, May 06–11, 2017, Denver, CO, USA
 ISBN 978-1-4503-4655-9/17/05âĂę$15.00
DOI: http://dx.doi.org/10.1145/3025453.3025599

2. We provide a formal way to derive **appropriate target sizes** from measures of accuracy and precision. Based on our data we give recommendations for the minimum size of gaze-aware regions to allow robust interaction.
3. We present an approach to find optimal parameters for any filter that minimizes target size and signal delay. We compare five commonly used **filtering techniques** and provide optimized parameters.
4. We show our vision of **error-aware adaptive applications** and show how our results and methodology can be utilized in the design and operating principle of such applications.

Our methods can be re-applied to future hardware versions or in other contexts to derive updated measurements as technology and use scenarios evolve.

BACKGROUND AND RELATED WORK

There are various technologies that can be used to capture eye gaze (see e.g. [11] for an overview). We focus on remote video-based eye tracking, in which an eye-tracking camera is attached to or integrated in a device's screen.

Computer Interaction by Eye Gaze

Eye trackers deliver information about where on the screen the user is looking. Applications can use this data for *explicit eye input* that requires conscious control by the user, or can use it as an implicit source of information, often in combination with other input modalities (*attentive user interfaces*) [26]. In the following, we use the term *gaze-enabled* to refer to any application that reacts to the changing gaze of the user. In this paper, we concentrate on estimation of gaze points by remote, video-based eye trackers. They capture the gaze location of the user in screen coordinates at regular intervals, depending on the recording frequency (commonly 30–60 Hz). Optionally, the system can apply filters to smooth the signal, and algorithms to detect events. The application then processes the signal and responds accordinglyNote that each processing step must happen in real-time and cannot rely on information from future gaze points without delaying the application.

The most important gaze events for interactive applications are *fixations* and *saccades*. A fixation denotes a period of time (from less than 100 ms up to several seconds [11, 25]) for which the gaze location is static within a small region. A saccade is the rapid and ballistic movement between fixations; they take about 30–80 ms [11] and can be 2° or larger [26]. While there are other gaze events — including smooth pursuit, microsaccades or glissades [11] — fixations and saccades are essential to describe how the attention of the user changes and can be recognized most reliably [11]. Table 1 overviews interactions performed in various gaze-enabled applications, ranging from explicit eye input to attentive user interfaces, with several in-between these two ends of the continuum (this list is not exhaustive, but represents exemplars of common interaction styles). The second column shows the gaze events on which the interaction relies. Independent of the conscious level of control, the applications mainly use the same gaze events to extract information about the user's intention: single gaze points, fixations, and saccades, with a few exceptions using smooth pursuit. This work concentrates on applications that use saccades and fixations for computer interaction.

	Explicit or implicit interaction by eye gaze	Gaze event	Ref #'s
●	Gaze pointing	Fixation or gaze point	[7, 22, 49]
●	Gaze gestures	Sequence of fixations	[8, 16, 23, 33, 47]
●	Dwell-based selection	Fixation over min. time	[17, 24]
◐	Multimodal selection	Fixation or gaze point	[17, 20]
◐	Selection by following a moving object	Smooth pursuit	[35]
●	Drag and drop	Fixation or gaze point	[17, 20]
●	Rotary control and sliders	Smooth pursuit	[40]
◐	Switching windows	Gaze point	[10]
◐	Image annotation	Fixation	[10]
○	Reading	Fixation or gaze point	[42]
○	Focus of attention	Fixation or gaze point, saccade	[9, 15, 27, 29, 37]

Table 1. An overview of interactions performed in gaze-enabled applications. Circles mark the degree of explicit control from full control (●) to no conscious control (○). Most interactions rely on the detection of fixations and saccades, while a few utilize smooth pursuit of a target.

Accuracy and Precision of Gaze Tracking

The (spatial) quality of the tracked gaze can be measured in terms of precision and accuracy. Accuracy denotes the offset between the true gaze position and the recorded position, whereas precision measures the dispersion of recorded gaze points during a fixation (i.e. , the gaze points' standard deviation). We measure accuracy and precision separately in the x- and y-directions. Accuracy is often reported by manufacturers to be $< 0.5°$. However, in particular for remote eye trackers, offset from the true gaze point is often found to be larger than 1° [5, 30], even in controlled environments.

Factors that influence tracking quality

The tracking method considered in this paper relies on pupil and corneal reflection tracking, the most common method for estimating the gaze location [11]. The image in Figure 1 shows a user's eyes and the location of the pupil and corneal reflection, as estimated by the SMI REDn scientific. The heavy mascara, thick frames of glasses, and reflections shown in the image could potentially impair the quality of the captured gaze. In general, tracking quality is influenced by the interplay of various factors related to properties of the user's eye, the tracking environment, the tracking method, as well as the calibration procedure (e.g. [11, 30]). Such properties include glasses or contact lenses, varying eye physiologies, different levels of sunlight, properties of artificial lighting, and the resolution and focus of the camera, among others. In the resulting signal, four different types of noise have been recognized: system inherent, physiological, optic artifacts, and data loss [11]. Artifacts and data loss impact the interaction, but can be detected and removed from the data stream (at the cost of signal delay), whereas the detection of physiological and system generated noise relies on filtering by the system.

Algorithmic approaches to robust interaction

Various algorithmic methods have been proposed to filter or correct the estimated gaze point (see [11] for a review), but many rely on post-hoc processing that introduces high latency, making them unsuitable for real-time interaction. We extend

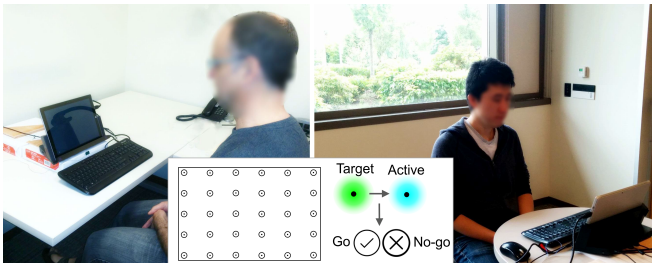

Figure 1. Image of the study setup. Left: in a room with no windows illuminated by artificial light. Right: in a room with large windows facing the tracker, recorded on a cloudy day. Inset: design of targets and go/no-go symbols and their distribution over the screen.

prior work [39] by evaluating several real-time filtering techniques for interactive applications based on our collected data, and propose an approach to choose optimal parameters. Recent efforts model and predict the error when estimating the gaze position on head-worn eye tracking glasses [2, 3]. Such models are the first step towards adaptive and personalized user interfaces for gaze interaction that adapt to the quality of the tracked gaze. However, at this stage, they are complex to train and implement, and it is unclear how they perform under less controlled conditions. Our focus is on remote eye tracking rather than head-mounted systems.

Design approaches to robust interaction

Several interaction methods and design approaches have been proposed to deal with poor accuracy and precision. One obvious approach is to enlarge targets. However, this is not always feasible as the overall size of the interface is limited. Space-efficient techniques, such as hierarchical menus, can facilitate interaction with large targets, but slow down performance, as they require more steps for selection. Dynamic approaches, such as zooming or fisheye lenses [1, 4, 9] can make selection more robust, but may visually distract the user. Gaze gestures [8, 16, 33, 47] have been found to be more robust against noise in comparison to target-based selections, but may require unnaturally large saccades to overcome tracking problems and may be hard to learn. Smooth pursuit [35, 40, 46] shows promise as a robust selection technique, but requires a moving stimulus that the eye can follow which is often not suitable for many traditional user interfaces. Requiring longer dwell times in dwell-based gaze selection is another approach to avoiding accidental target selection, though this slows down interaction times; expert users may be able to reliably select targets with dwell times near 300 ms, but only if target sizes are appropriate for the tracking quality [24]. Our findings can inform the design of many of the applications shown in Table 1 to optimally adapt to the tracking quality at a given time.

DATA COLLECTION

We collected gaze data across a broad range of people, using two different eye trackers in two environments with different lighting. We controlled important factors to be the same throughout the study, such as position of the user relative to the tracker, but kept the setup as natural as possible to capture how much accuracy and precision of eye tracking can vary in tracking conditions similar to those in practice.

Age	18–24	25–34	35–44	45–54	55–64
	12	45	14	8	1
Eye color	Blue	Green	Hazel	Dark brown	
	11	9	10	50	
Ethnicity	Asian or Pac. Isl.	Black or Afr. Am.	Hispanic	White	Other or Mixed
	23	7	9	34	7
Experience	Never used eye tracking			Has used eye tracking	
	52			28	

Table 2. Study participants' demographic information

Participants: 81 participants took part in the experiment (62 male, 19 female). When advertising the study, we explicitly encouraged people to take part even if they wore glasses or contact lenses, or had experienced problems using eye tracking before. Table 2 shows demographic information. The calibration process failed for only one participant (male) for whom no data could be collected, and which is excluded in Table 2. Participants were compensated with a $5 lunch coupon.

Eye trackers: We collected gaze data with 2 different eye trackers: Tobii EyeX [45] and SMI REDn scientific [14], both at 60 Hz. The first is an affordable tracker developed for consumer use, the second is sold for scientific purposes and is more expensive. The trackers have a similar form factor and can be attached to the bottom of screens up to 27". They operate at a similar tracking range, not restricting the user in their natural movement. SMI reports an accuracy of 0.4° and a spatial resolution (root mean square) of 0.05. No such values are available for the Tobii EyeX [44]. Participants were assigned randomly to use one of the eye trackers.

Tracking Environment: We collected data in two different lighting environments, shown in Figure 1. The *artificial* environment was a room with no windows; light was from halogen MR16 spot lights and two fluorescent 40-watt Biax lamps in 2×2 fixtures. The *daylight* environment was a room with large windows facing the tracker; additional lighting came from the same fluorescent lights used in the artificial setup. Data was collected on a cloudy day. While both conditions do not provide the optimal lighting recommended for an eye tracking lab [11], they allow us to capture the noise induced by external lighting that may be encountered in practice in end users' homes, offices, etc. Participants were assigned randomly to one environment.

Setup: The equipment setup was the same in both environments. We used a Microsoft Surface Pro 3 with an Intel Core i5-4300U 1.9 GHz and 4 GB memory. The application was shown on the 12" screen at a resolution of 1440×960 pixel (144 DPI). Both trackers were magnetically affixed to a horizontal bar at the bottom of the Surface's docking station with the material provided by the manufacturers. We placed the display on a table at the same height in both environments. To compensate for different participant heights, we tilted the docking station towards the user to adapt the gaze angle between the user's eyes and the eye tracker, until the manufacturer's calibration software showed a good tracking position. Participants were positioned at 65 cm from the screen. They were told to try to stay still, but that they could move their head naturally. With this setup, a visual angle of 1° corresponds to a movement of ca. 1.1 cm on the screen.

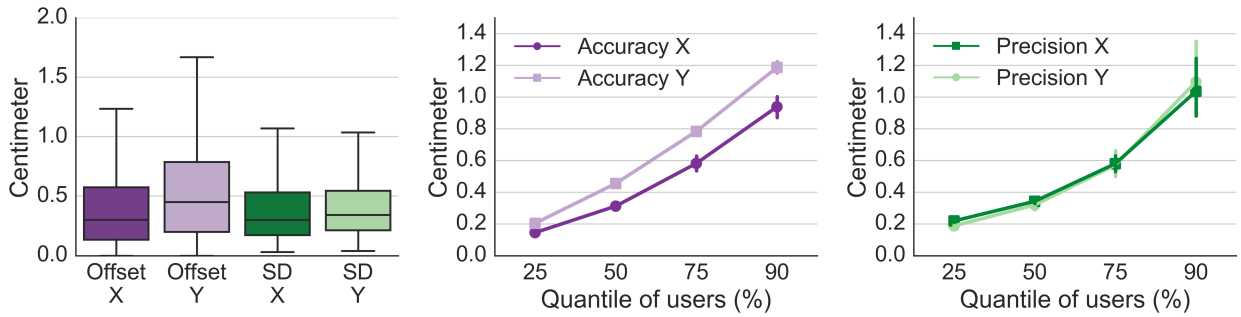

Figure 2. Left: box plot of the accuracy (offset) and precision (standard deviation) over all target fixations. Center and right: different percentiles, averaged over all targets, show that accuracy and precision vary largely in practical tracking conditions.

Task and Procedure: The participants' task was to look at targets randomly presented on the screen for about two seconds each. If the target turned into a checkmark, they were instructed to press the spacebar as quickly as possible; if it turned into a cross, they should wait for the next target. This go/no-go task was used to keep the participants' attention on the target [12]. Thirty targets were shown in random order, distributed over the screen as shown in Figure 1, with a vertical distance of 214 pixel (3.32°, 3.77 cm), and a horizontal distance of 267 pixel (4.14°, 4.71 cm) between neighboring targets. Each target's (invisible) activation area was a circular region of 500 pixel diameter (8.8 cm, 7.7°). When the participant's gaze was inside the target for 500 ms, it was activated and its color turned cyan. After 1250 ms the go/no-go task started. If a target could not be activated after 5 s (because the tracked gaze was too noisy), the next target appeared. Figure 1 shows the targets and the go/no-go icons. Before the task started, we calibrated the eye trackers with a 7-point calibration method provided by the manufacturers. The study started with a demographic questionnaire. After that, the instructions for the go/no-go task were displayed, and then the task began.

Data Collection and Preprocessing: All gaze points were recorded as an (x,y) position in screen coordinates, as given by the API of the respective tracker, with filtering turned off. For each target, we extracted the gaze points of a 1 s window (60 frames) during which the participant looked at the target. We did not use a fixation detection algorithm, which cannot reliably distinguish between saccades and noise artifacts, but extracted the data in one of two ways: (1) *Online:* A target was activated if the gaze of the user stayed within a 250 pixel radius from the target for 500 ms. Gaze points were extracted for the next 1000 ms and saved if the gaze was still within that region after that time; otherwise, the activation was reset. The go/no-go task was only shown if we could extract the gaze points. (2) *Offline:* If a target could not be activated for 5 s, the next one was shown. From the recorded data we extracted 1 s during which we could be sure that the participant looked at the target. Therefore, we chose a 1 s window for which on average the gaze points were closest to the target. This post-processing had to be done for 11% of the fixations

For each fixation, we then excluded the gaze points where no eyes could be tracked (reported as such by the eye tracker). If this was more than 2/3 of the gaze points, we excluded this fixation from the analysis.

ANALYSIS AND RESULTS

The analyzed dataset consists of 2,343 fixations from 80 users looking at 30 targets (2.4% of fixations were excluded, as described above). By "target fixation" we simply denote the data recorded while a user was looking at a target. No algorithmic method was involved to extract fixation events. In the following analysis, we first compute aggregate measures for each target. Where not denoted differently, we then average across the 30 targets for each user to account for the fact that fixations by the same participant are not independent observations.

Data Loss and Distribution of Gaze Points

Data loss occurs when the eye tracker cannot estimate the gaze position. On average, each target fixation consists of 55.6 gaze points recorded over 1 s (ca. 60 frames). The average data loss is 7.9% of frames. A two-way ANOVA shows significantly higher loss for the Tobii tracker than for the SMI: gaze position could not be estimated for 13.1% versus 2.8% of points per target ($F(1,79) = 18$, $p < 0.01$). No such difference was found for the tracking environments (8.1% in artificial versus 7.8% in daylight) and there was no interaction effect. For 1,450 fixations (62%) no data was lost. For the remaining 38%, tracking was lost on average 11 times during the 1 s fixation, for an average of 1.5 frames.

We performed normality tests on the gaze points of each target fixation, separately in the x- and y-directions (D'Agostino and Pearson's test as implemented by Python's SciPy package [7, 20] with a significance level of $p < .01$). They showed that on average gaze points are normally distributed in the x-direction for 71% of the fixations, and in the y-direction for 74%.

Precision and Accuracy

For all gaze points during a target fixation we compute accuracy and precision separately in the *x*- and *y*-direction. Accuracy denotes the absolute offset between the fixated target and the mean of the estimated gaze points. Precision is computed as the standard deviation (SD) of the gaze points.

Table 3 shows an overview of the results. Values are averaged across users and given for different conditions. We also show average accuracy and precision values for different percentiles of users fixating each target, e.g. only the best 25%. Standard deviations are given in brackets. We used multivariate ANOVA to test for significant differences in accuracy and precision between the tracker and light conditions. We report

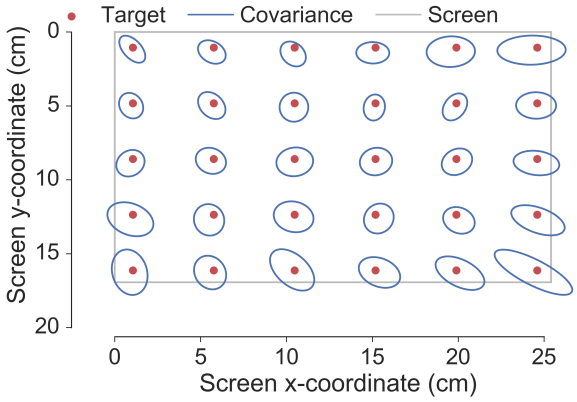

Figure 3. Precision of the estimated gaze points over the different targets. Covariance ellipses show the contour of the 2D Gaussian, fitted to the gaze points corresponding to all fixations of the same target.

values in cm, which is independent of screen resolution. For space reasons, we do not report the values in degrees of visual angle, but refer to the supplementary material for an extended version of the table. Our results should not be seen as absolute properties of the tested eye trackers or tracking environments, but show how much we can expect accuracy and precision to vary as eye tracking is used in everyday computer interaction.

Variation over all conditions

Figure 2 shows how much accuracy and precision can vary over all collected data. Table 3 gives the corresponding results. A paired samples t-test showed that on average, accuracy is worse in the y-direction by about 0.1 cm ($t(1,79) = 2.8, p < .01$). The middle and right plot in Figure 3 visualize how accuracy and precision increase over different percentiles. No significant difference between the x- and the y-direction was found for precision ($t(1,79) = 1.0.3$, $p = 0.3$).

The middle and right plots in Figure 2 visualize how accuracy and precision become worse for different percentiles of users. The results given in Table 3 are average values over the corresponding percentiles of each target. Very accurate fixations (25th percentile) are only 0.15 cm in the x- and 0.2 cm in the y-direction offset from the target. On the other hand, inaccurate fixations (90th percentile) can be as far offset as 0.93 cm in the x- and 1.19 cm in the y-direction, a more than six-fold difference. Similar for the spread of the gaze points.

Comparison between tracking conditions

Two-way MANOVA found a significant difference between the eye trackers used in the study ($F(1,79) = 7.7$, $p < .01$). Follow-up univariate testing showed that the Tobii EyeX tracker was more accurate than the SMI REDn scientific in both the x-direction ($F(1,79) = 9.4$, $p < .01$) and y-direction ($F(1,79) = 10.6$, $p < .01$), whereas no significant difference was found for precision. This stands in contrast to our previous finding that the Tobii EyeX has significantly more data loss than the SMI REDn, in which case it cannot track the gaze position at all. As shown in Table 3, tracking in the daylight condition was consistently better by about 0.1-0.15cm. However, this difference was not found to be significant($F(1,79) = 1.7$, $p = 0.16$) by the two-way MANOVA. Also there was no significant interaction effect.

	Accuracy (cm)		Precision (cm)	
	X	Y	X	Y
Overall	0.47 (0.64)	0.57 (0.52)	0.52 (0.73)	0.5 (0.56)
Percentile				
25%	0.15	0.2	0.19	0.22
50%	0.31	0.45	0.32	0.32
75%	0.58	0.78	0.57	0.58
90%	0.93	1.19	1.1	1.04
Tracker				
SMI REDn	0.58 (0.75)	0.66 (0.57)	0.51 (0.91)	0.51 (0.64)
Tobii EyeX	0.36 (0.5)	0.48 (0.45)	0.53 (0.44)	0.48 (0.44)
Light				
Artificial	0.51 (0.75)	0.65 (0.6)	0.6 (0.87)	0.56 (0.68)
Daylight	0.42 (0.51)	0.5 (0.42)	0.44(0.56)	0.44 (0.4)

Table 3. Average accuracy and precision across all users, across different percentiles of each target, and for different tracker and light conditions (lower values are better). Standard deviations are given in brackets. An extended table with degrees is provided in the supplementary material.

Comparison between screen regions

The precision of the estimated gaze for each target is worse toward the right and bottom edge of the screen, as shown in Figure 3. The ellipses represent the covariance matrix computed over all gaze points from all 80 users. Multivariate ANOVA and follow-up univariate analysis with TukeyHSD tests found that the precision of the bottom row was significantly worse compared to all other rows at the $p < .05$ (adjusted) significance level in both x- and y-direction. In contrast, there was no significant difference for accuracy between any of the rows.

Significant differences between columns were only found for precision in the x- direction. Precision was significantly worse for the right-most (6th) column in comparison to all other columns at the $p < .01$ (adjusted) significance level. In the supplementary material we provide exact accuracy and precision values for rows and columns.

Comparison over time

The tracking quality did not significantly change for the same participant during the experiment. MANOVA found no difference for either accuracy or precision between the first and last target fixation of each user ($F(1,158) = 0.9$, $p = 0.46$).

FROM ACCURACY AND PRECISION TO TARGET SIZE

Once we know the tracking accuracy and precision, we can derive appropriate target sizes. A target denotes any region in a user interface that should recognize if the user's gaze falls inside its borders in order to trigger a functionality. This can be an explicitly controlled button, the words and lines in an attentive reading application, the active recognition region for gaze gestures, or the area of interest in a research experiment.

When a user fixates the center of a target, we assume that the captured gaze points are normally distributed in the x and y directions around a mean with offset $O_{x/y}$ (accuracy) from the target center and a standard deviation $\sigma_{x/y}$(precision). For robust and smooth interaction, we suggest that at least 95% of gaze points should fall inside the target regions; this means we can derive the target width and height as:

$$S_{w/h} = 2\,(O_{x/y} + 2\,\sigma_{x/y}) \quad (1)$$

We multiply $\sigma_{x/y}$ by 2, as about 95% of values lie within two standard deviations of the mean for normally distributed data.

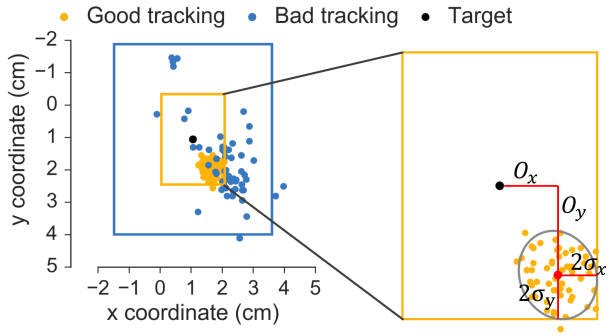

Figure 4. Given the accuracy and precision, target size is computed based such that 95% of gaze points fall inside the target. The plot shows example target sizes based on two different fixations of the same target, one with good and one with bad tracking quality.

An error rate of more than 5% (every 20th point) slows down performance and leads to errors that are often hard to recover from.However, if an interaction technique can be used reliably with a smaller percentage of points, Equation 1 can be easily modified by replacing the multiplier with the corresponding value from the standard normal table.

Figure 4 visualizes the computation of the target size based on the offset and SD of the raw gaze points and shows two example targets based on the gaze points of two fixations with good and bad tracking quality. Although Equation 1 is a simplification and not a statistical assessment of the distribution of the gaze points, it can be used to quickly compute appropriate target sizes based on precision and accuracy values. When we check this formula based on our collected data, we get that for each target fixation on average 94.9% of gaze points fall inside a target of width and height as computed by Equation 1. Table 5 shows the resulting average target sizes computed based over all target fixations, and over different percentiles of users for each target. Sizes can vary from 0.94×1.24 cm for users that track well, up to 5.96×6.24 cm if we want to allow robust interaction for nearly all users in our dataset. These target sizes are based on unfiltered data; next, we discuss filtering techniques and derive similar values based on filtered data.

PARAMETER OPTIMIZATION FOR FILTERS

Filters can significantly improve the precision of the tracked gaze by removing noise artifacts and reducing signal dispersion. Several filters have been proposed for real-time systems; however, developers implementing these filters must choose parameters that control their performance. This is often done suboptimally based on prior experience or heuristics. We describe an easy-to-implement procedure to optimize the parameters of any filter for minimum target size and minimum delay, based on calibration-style data as collected in our study. We then compare five different types of filters with three different kernel functions, and evaluate their effect on precision and accuracy in comparison to the unfiltered data.

Filters for Interactive Gaze Applications

In interactive gaze-enabled applications, no post-processing of the data is possible, and any filtering must be done in real-time. This limits the ability to detect outliers and artifacts, which introduce a delay of several frames. For most applications

Filter	Parameter	Ranges
Weighted Average (WA) [19, 48]	Window size N	2–40 frames
WA + saccade detection [39]	Saccade threshold s Max. window size N	1–4 cm (à .05) 2–40 frames
WA + saccade detection + 1-sample outlier correction	Saccade threshold s Max. window size N	1–4 cm (à .05) 2–40 frames
1€ filter [6]	Adaption rate β Cut-off frequency $f_{c_{min}}$	0–1.5 (à .01) 0.4–6.5 (à .1)
Stampe filter [41]	–	–

Table 4. Filtering techniques for which we optimize parameters and that we evaluate based on our collected data. Ranges denotes the parameter ranges considered in the grid search.

there are two main states that need to be recognized: fixations and saccades. Eye tremor, microsaccades, and noise should be filtered in order to stabilize the signal and improve precision. The quick and sudden change between those states must be accounted for by the filter. This makes commonly used methods, such as moving average, Kalman filter, and Savytzki-Golay, filter less useful [6, 39]. Based on prior work [39] and our own experience, we tested five different types of filters. They are summarized in Table 4 and described in the following.

The Stampe filter [41] is commonly used to remove outliers from the tracked gaze signal. It uses a two-step procedure that detects one- and two-frame deviations from a continuously increasing or decreasing signal and corrects them according to the surrounding points. This introduces a delay of up to two frames. No parameters are necessary.

Weighted average (WA) [19, 48] or finite impulse response filter (FIR) computes the filtered point $\hat{X}_t$ at time t as the weighted average over the last N points:

$$\hat{X}_t = \sum_{i=0}^{N} \frac{w_i}{\sum_j w_j} \cdot X_{t-i} \qquad (2)$$

where X_{t-i} is the point observed at time $t-i$, and w_i is the corresponding (normalized) weight. The only parameter for a WA filter is the window size N. The function defining the weight for each point in the averaging window, is called the kernel function. We test three different kernel functions:

1. A *Linear kernel* corresponds to a simple average of the last N points, $w_i = 1$.
2. A *triangular kernel* [21, 48] assigns a weight of 1 to the least recent point (N), 2 to the next point ($N-1$), and so on, $w_i = N - i + 1$.
3. A *Gaussian kernel* function [19] assigns weights according to a Gaussian function, where the variance is chosen such that no point in the averaging window is assigned a weight smaller than 0.05 [39], $w_i = e^{-\frac{(i-1)^2}{2\sigma^2}}$

Saccade detection can be used to extend the WA filter [36]. Similar to the identification by dispersion threshold algorithm [34], commonly used for fixation detection in offline processing [11], it defines a threshold s that is used to detect saccades. As long as the distance between successive gaze points is smaller than s, they are assumed to belong to the same fixation. The filtered gaze point is computed as the weighted average over the last N points of the same fixation. If the distance exceeds the threshold, a saccade is detected and

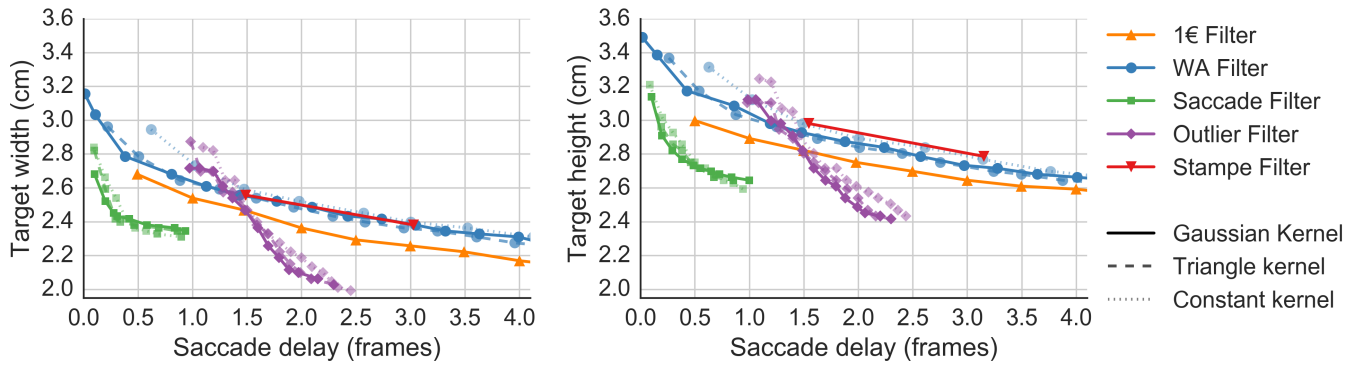

Figure 5. Pareto front of optimal parameter settings for each tested filter. Filters are evaluated based on the resulting target size and average delay in detecting a saccade. Corresponding parameters are given in the supplementary material. Depending on the application, appropriate parameters can be chosen that yield the desired trade-off between target size and signal delay.

no filter is applied. The new gaze point is the first of a new fixation. We test the same kernel functions as described above. In addition to the window size N, the filter requires to set the saccade threshold s.

Outlier correction further extends the WA filter [21]. It detects and removes single outliers that would be detected as two successive saccades by the previous filter. If the distance between successive gaze points exceeds the saccade threshold s, the new gaze point is saved as the start of a potential new fixation. The algorithm waits for the next observation to confirm the detection of a saccade. If the next gaze point is closer to the previous fixation, the potential fixation is discarded as an outlier. Otherwise, a new fixation is started. This introduces a one-frame delay. Points belonging to the same fixation are filtered by a WA filter, as before. The parameters are the size N of the averaging window and the saccade threshold s.

The 1€ filter [6] has been proposed for filtering the signal obtained from human motion tracking. It uses a low-pass filter but adapts the cut-off frequency depending on the movement speed. At low speeds, the signal is heavily smoothed, at the cost of responsiveness to signal changes. As speed increases, the cut-off frequency is increased, thus the filter is less effective but more responsive to the fast changes in the signal. This principle makes the 1€ filter potentially well-suited for eye tracking, where static fixations alternate with very fast saccades. To our knowledge it has not been applied to eye tracking data before. It has two parameters to control smoothness and responsiveness of the filter: the cut-off frequency $f_{c_{min}}$ and the adaption rate β.

Parameter Optimization

Most of these filters require the choice of one or more parameters, which determine the effectiveness of the filter. We propose an approach to optimize parameters based on data collected through a calibration-style study such as the one described earlier in this paper. In a grid search, we instantiate a filter with each possible parameter, apply it to the data, and evaluate its effect on two criteria: the resulting *target size* and the *delay in following a saccade*. For each possible parameter, the following steps are performed separately in the x- and y-direction:

1. **Target size:** For each target fixation in the data set, apply the filter, compute the accuracy and precision of the filtered gaze, and derive the corresponding target size according to Equation 1. Compute the target size $S_{75\%}$ over all fixations as the 75th percentile of all target sizes. This represents a target size that allows for robust interaction, where for 75% of fixations at least 95% of gaze points fall inside the target.
2. **Saccade simulation:** Simulate saccades between two neighboring targets of size $S_{75\%}$. For each target fixation, simulate a second fixation by shifting each gaze point by $S_{75\%}$ cm and append it to the original fixation. The resulting signal simulates a fixation-saccade-fixation sequence. Create a second sequence by shifting the original points by $-S_{75\%}$ cm in the negative direction.
3. **Saccade delay:** Apply the filter to the constructed sequences. For each filtered sequence, compute saccade delay as number of frames from the start of the second fixation until the signal first exceeds the lower target bound of the second target. Adjust the delay by any delay the unfiltered signal may have due to signal noise.

In this way, for each potential set of parameters, we compute the target size necessary for robust interaction and the average signal delay the filter would lead to when used in a gaze-enabled application. In order to decide on the best parameters, we have to trade off these two opposing criteria. Therefore, we identify the Pareto optimal solutions by discretizing the delay, and then for each delay, determining the parameters that yield the minimum target size without exceeding this delay.

Figure 5 shows the resulting Pareto fronts for applying each filter in the x- and y-direction. Each point denotes a parameter setting that yields the minimum target size for the corresponding delay. The *Weighted Average filter* can be seen as a baseline. Its only parameter, the size of the averaging window, determines the trade-off between small target size and short delays. The *Gaussian and Triangle kernel* perform slightly better than the *Constant kernel*. They give a higher weight to more recent points, thus allowing for a larger averaging window at the same signal delay than the Constant kernel. In practice the *Stampe filter* yields a larger delay than expected from its construction. The two points in the plot represent the first and second step during which one- and two-frame outliers

	Target width (cm)			Target height (cm)		
	Raw	Filt.	Diff	Raw	Filt.	Diff
Overall	3.0	2.02	33%	3.14	2.19	30%
Percentile						
25%	0.94	0.58	38 %	1.24	0.8	35%
50%	1.8	1.12	38 %	2.26	1.48	35%
75%	3.28	1.9	42 %	3.78	2.35	38%
90%	5.96	3.9	35 %	6.24	4.24	32%

Table 5. Recommended target sizes for robust and smooth interaction by eye gaze, and a comparison between sizes computed based on raw data and filtered data. Percentiles show how much target sizes varies for different levels of tracking quality.

are corrected, in theory yielding one- and two-frame delays. However, depending on the noise structure, the Stampe filter may detect saccades as outliers, thus yielding a larger delay. In practice, this leads to similar or worse performance than the weighted average filter. The *1€ filter* performs better than the WA filter, due to the dynamic adjustment of the cut-off frequency. However, in practice this adjustment is not enough for the filter to react to the sudden and large signal jumps corresponding to a saccade. Moreover, the cut-off frequency is also adjusted to noise artifacts that show similar dynamics as saccades. The *Saccade filter* performs best for a delay up to one frame (16 ms with a 60 Hz tracker). While in theory, it does not introduce any delay, in practice a saccade may not be detected, in which case a strong averaging filter is applied. On the other hand, choosing a small saccade threshold ensures minimal delay, but increases the number of outliers wrongly detected as saccades, thus yielding a larger target size. A similar trade-off applies to the *Outlier filter*. The construction of the filter introduces a minimum delay of one frame. However, stronger filtering is achieved if we increase the saccade threshold and thus take into account larger delays if a saccade is not recognized, on average up to about 2.5 frames.

The supplementary material contains corresponding parameter settings for each filter. Developers can use these results to choose the best filter settings appropriate for their application. If a different setup is used or as technology develops, the presented approach and metrics can be used to derive optimal parameters for any given filter based on data obtained from simple target fixations.

Effect on Accuracy and Precision

Based on our comparison, we choose to filter our data with the Outlier filter with a Gaussian kernel, which extends weighted averaging with saccade detection and outlier correction. According to our optimization, we choose a saccade threshold of 1.45 cm in x- and 1.65 cm in the y-direction, and a window size of 36 and 40 frames respectively. In practice, this would yield a delay of about two frames (32 ms with a 60 Hz tracker).

Figure 6 shows the resulting average target sizes based on the filtered and unfiltered data of each target fixation. Table 5 compares the average sizes based on the raw and filtered data. The first row shows the average overall data. Precision has improved from 0.53 cm to 0.29 cm in the x-direction and from 0.51 to 0.27 cm in the y-direction, while accuracy has approximately stayed the same. The other rows show the variation of target size for different percentiles of target fixations. They

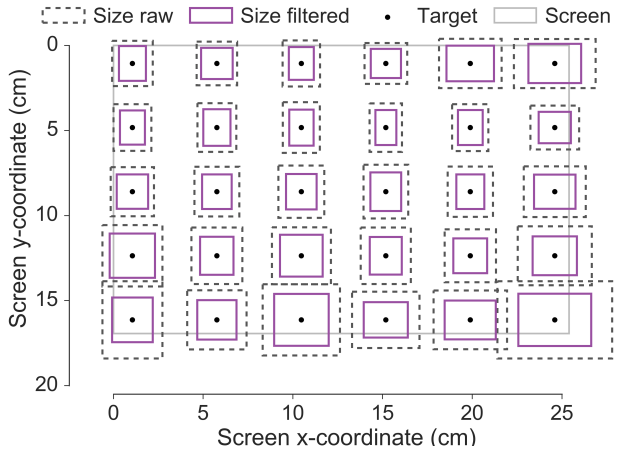

Figure 6. Appropriate target sizes for different screen positions, computed based on the raw and the filtered data.

are computed based on the corresponding percentile for each target and averaged over all targets. The optimized Outlier filter reduces the target size by up to 42%. However, the filter can only improve the precision of the data, not its accuracy.

IMPLICATIONS FOR GAZE-ENABLED APPLICATIONS

We can use our findings to derive implications for the design of gaze-enabled applications that are used by a broad range of users under varying tracking conditions. Such implications are useful for designers and developers, but also for researchers who want to minimize the risk of having to exclude users from their study or who want to conduct studies in more natural environments.

Target size and distances: Gaze-enabled regions should be designed slightly larger in height than in width, since we observed consistently larger standard deviations in the y-direction. According to our data, target sizes of at least 1.9×2.35 cm allow for reliable interaction for at least 75% of users if optimal filtering is used. If no filter is implemented, this should be enlarged to 3.28×3.78 cm. Targets of this size can be placed directly next to each other. Note that the size only denotes the region that is reactive to the eye gaze; this can be invisible to the user and the corresponding UI elements can be smaller, in which case the distance between graphical elements must then take into account the larger tracking area.

Screen regions: Avoid placing gaze-enabled elements towards the bottom or right edge of the screen, for which accuracy and precision was found to be significantly worse.

Filtering: Use a weighted average filter with a Gaussian or Triangular kernel and saccade detection. If the application can account for a two-sample delay, additional outlier correction (Kumar et al. 2008) can improve precision further and reduce recognition of false saccades. In this work, we obtained best results with the following parameters: window size of 36/40 frames, saccade threshold of 1.45 cm/1.65 cm in x-/y-direction. The latter one could be further adapted to the specific applications, for example when targets are far apart and expected saccades thus larger. Therefore, the supplementary material provides optimized parameters.

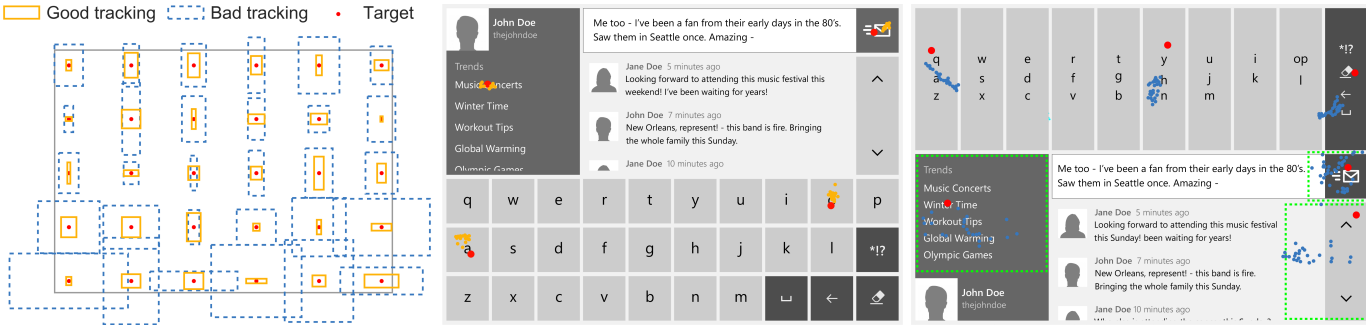

Figure 7. Example case that shows how the design of a gaze-controlled social network application can be adapted to the changing tracking quality. Left: Target sizes for good and bad tracking, based on data from two users. Middle: UI for good tracking quality. Right: UI adapted to bad tracking. The keyboard has moved up where tracking is best and letters are grouped to account for vertical offset and spread of gaze points. Green lines show that the active gaze region of the buttons is actually larger than the graphical UI element, accounting for bad tracking near the bottom of the screen.

Error-Aware and Adaptive Applications

Our findings open up a new set of questions, setting the stage for research in future areas such as adaptive UIs. We share the vision of other researchers (e.g. [2, 3, 36]) that applications should be aware of uncertainty in the input signal and adapt their interface and functionality accordingly. However, eye tracking poses a challenge in that the error of the estimated gaze point varies substantially depending on the interplay of many factors and thus is hard to predict and correct. Therefore, we envision applications consisting of flexible modules that adapt their graphical representation and functionality to variations in accuracy and precision of the tracked gaze. The results and methods presented in this paper are an important step towards the the design of such adaptive modules. To exemplify this, we sketch an example application in Figure 7. We show how the user interface of a social network application can be adapted based on example data of two different users from our data set. We envision the application to adapt to the changes in tracking quality by going through the following steps:

1. Collect information about tracking accuracy and precision by analyzing target fixations during calibration and usage. Utilize the data to compute appropriate target sizes as shown in Equation 1. The left plot in Figure 7 shows example target sizes based on data from two different users. Yellow denotes "good" and blue "bad" tracking quality.

2. Choose optimal filter parameters, trading off signal response and precision based on the characteristics of the application. For example, if error corrections are costly, a more aggressive filter could be chosen if signal quality drops, taking into account delayed response of user actions for a more robust interaction. Optimal parameters for a range of acceptable delays are given in the supplementary material.

3. Adapt functionality and design of the UI to account for the changes in tracking quality, either as predefined or optimized in real-time. This is exemplified in Figure 7. The middle part shows a UI designed for good tracking quality: it has a full keyboard placed on the bottom of the screen (as familiar from mobile interfaces); links and buttons can be selected individually. The UI on the right is adapted to poor tracking. The keyboard is moved to the top where tracking is relatively best, and letters are grouped to account for the vertical offset and spread of gaze points (grouped-key text entry may then be disambiguated algorithmically using text prediction, or combined with a secondary input action to select individual characters). The links under "Trends" are merged into one gaze target, while keeping them visually as before; if activated, an alternative pointing technique like zooming [4, 10] or fish-eye lenses [1] could be applied. Similarly, the scroll buttons are kept as before, but their active gaze region expands beyond the screen and toward the static message window.

4. Optimize when to adapt the UI, by monitoring changes in tracking and predicting the potential gain. In order for adaptive interfaces to be usable, the system needs to take into account the cost for the user to adapt to the new UI. To determine the best changes the system trades off criteria such as interaction cost, learning time, or user preference in interaction with the user. Toward this end, future work is needed to develop appropriate user models and optimization methods.

Other types of applications could benefit from information about accuracy and precision. For example, the more accurate and precise the estimated gaze points, the more complex and small the **gaze gestures** that can be used. During runtime, a gesture system could adapt to loss of tracking quality and encourage the user to perform larger gestures, or provide different gesture sets for different levels of tracking quality. Recognition algorithms could also take into account the quality of the tracked gaze to better differ between noise in the signal and noise in the gesture performance. **Attentive reading applications** (e.g. [13, 38]) could adapt their font size and word spacing for better recognition of fixations during reading. Visualizing tracking quality to the end user could be useful, for example in **remote collaborative settings** where information about collaborators' gaze can be a useful communication tool, but a noisy signal can be confusing [7].

DISCUSSION

We showed how eye tracking precision and accuracy can vary under tracking conditions similar to those encountered in practice. Our work underscores the importance of error-aware applications that adapt to variations in tracking quality, and we suggested several opportunities for how our results can inform the design of such applications. In the following, we discuss the generalizability of our findings to applications used in-the-wild and opportunities for further work.

Generalizability of Results

To assess the generalizability of our results to gaze input in everyday computer interaction, we have replicated our data collection with six experienced eye tracking users (people with ALS who rely on gaze-based AAC for communication), in environments and setups in which they use their eye trackers in practice. The task was the same as described in the Study section, but we only used the Tobii EyeX tracker, and mounted it on a rolling stand (rather than table) for wheelchair accessibility. Compared to the first study, gaze points were tracked with a similarly large variation in precision; multivariate ANOVA found no difference for precision in either the x- or the y-direction. However, accuracy of the gaze points varied even more, ranging from 0.16 cm and 0.21 cm in the x- and y-direction if tracking was good (25th percentile) to 1.32 cm and 1.67 cm in the x- and y-direction for poor tracking (90th percentile). Multivariate ANOVA and follow-up univariate testing found the difference in accuracy to the first study significant ($F(1,86) = 2.6$, $p = 0.04$). Further analysis showed, that in comparison to the first study, there were more outliers for which offset was over 1 cm. A Mood's median test found no significant difference in accuracy between the two studies.

This shows that the conditions created in our study closely resemble those that we can expect in practice. However, when applying our findings designers should keep in mind that in particular the accuracy can vary even more, suggesting that designers should be conservative in their design of targets.

Limitations and Opportunities for Future Work

Several characteristics of our study design may lead to even larger variations in-the-wild. We computed target sizes with the assumption that users look at the center of a target. If the user's gaze changes within a small region, accuracy and precision will likely stay the same but more gaze points will land outside a target as the gaze gets close to its border. It is a separate research question how the visual design of targets (label shape, color, etc.) changes the focus of a user.

Our study was short, thus we did not find differences in accuracy or precision between a user's first and last fixation. In practice, we expect that accuracy and precision further decrease over time (due to movement, fatigue, etc.).

We found that accuracy and precision become worse toward the bottom and right edge of the screen. However, both tested trackers can be used with screens up to 27", which is larger than the 12" screen we used. We expect tracking to also get worse toward the left and top edges as targets are placed closer to the spatial limits of the tracker; however, a similar study on a larger screen is necessary to confirm this expectation.

We used two commercially-available remote, IR eye trackers under two lighting setups. The specific accuracy and precision measures we report may vary under other environmental conditions and/or with different equipment. However, our study reveals that in practical conditions, tracking quality is much more affected by differences between individual users than trackers or lighting. Using mixed linear regression to model the accuracy and precision of our data, we found that on average differences in lighting and tracker only account for up to 8.3% of the deviation in tracking quality, whereas differences between users account for up to 32.8%. Thus, it is reasonable to assume that data collected under similar conditions with other eye trackers are likely to show a similarly large variation in tracking accuracy and precision; formally investigating this would be a valuable direction for future research. For applications to be usable in practice, it is important to account for these large variations in tracking quality, regardless of the source. But in order to generate updated measurements as use scenarios and eye tracking devices evolve, other researchers can use the experimental and analytic methodologies we described in this paper.

CONCLUSION

Eye tracking has the potential to revolutionize how we interact with computers. Attentive applications seamlessly integrate gaze input into existing interaction paradigms and explicit gaze control opens new design spaces for HCI. While researchers have developed many interaction techniques and applications that use gaze information, there was a gap in understanding the requirements of such applications as they are released to consumers. Toward this end, our work provides important contributions for developers and designers of gaze-enabled applications, as well as researchers using eye tracking:

1. We showed that there is **large variability in accuracy and precision** when eye tracking is used in more practical conditions. Differences between users can be more than six-fold, and we analyzed variations of trackers and screen regions.
2. We propose a way of formally going from measures of accuracy and precision to **target sizes**. We provide look-up tables for appropriate target sizes and discuss the best screen regions. Our findings can be directly used by people developing gaze-enabled applications.
3. We **optimize and evaluate filters** that work for eye tracking data in practical conditions. We propose an optimization approach to find parameters that minimize target size and signal delay, given calibration-style data. Developers can directly use our results to choose the filter and parameters appropriate for their application, while researchers can follow our approach as they develop new filters.
4. We outline our vision of **error-aware gaze application**, that adapt their design and interaction techniques to the current tracking quality. We discuss several examples and provide a walkthrough how our findings can be used to inform the design of such applications.

For eye tracking to become a ubiquitous part of our everyday interaction with computers, it is important to more systematically explore the question of "What is good design?' for gaze input, with the goal of establishing standards and frameworks for robust, usable gaze applications. The presented findings are an important step toward this goal, and our methods can be re-applied to future hardware versions or in other contexts to derive updated measurements as technology evolves.

ACKNOWLEDGMENTS

We thank Mike Bennett, Ed Cutrell, Shahram Eivazi, and Antti Oulasvirta for their insightful comments. We also thank our PALS and study participants, and the MSR Enable team.

REFERENCES

1. Michael Ashmore, Andrew T. Duchowski, and Garth Shoemaker. 2005. Efficient eye pointing with a fisheye lens. In *Proceedings of Graphics Interface 2005*. Canadian Human-Computer Communications Society, Waterloo, 203–210.
2. Michael Barz, Andreas Bulling, and Florian Daiber. 2016a. Computational Modelling and Prediction of Gaze Estimation Error for Head-mounted Eye Trackers DFKI Technical Report. (2016).
3. Michael Barz, Florian Daiber, and Andreas Bulling. 2016b. Prediction of gaze estimation error for error-aware gaze-based interfaces. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications - ETRA '16*. ACM Press, New York, New York, USA, 275–278. DOI: http://dx.doi.org/10.1145/2857491.2857493
4. Richard Bates and Howell Istance. 2002. Zooming interfaces!: enhancing the performance of eye controlled pointing devices. In *Proceedings of the fifth international ACM conference on Assistive technologies - Assets '02*. ACM Press, New York, New York, USA, 119. DOI: http://dx.doi.org/10.1145/638249.638272
5. Pieter Blignaut, Kenneth Holmqvist, Marcus Nyström, and Richard Dewhurst. 2014. Improving the Accuracy of Video-Based Eye Tracking in Real Time through Post-Calibration Regression. In *Current Trends in Eye Tracking Research*. Springer International Publishing, Cham, 77–100. DOI: http://dx.doi.org/10.1007/978-3-319-02868-2_5
6. Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1€ filter: a simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. ACM Press, New York, New York, USA, 2527. DOI: http://dx.doi.org/10.1145/2207676.2208639
7. Sarah D'Angelo and Darren Gergle. 2016. Gazed and Confused: Understanding and Designing Shared Gaze for Remote Collaboration. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, New York, New York, USA, 2492–2496. DOI: http://dx.doi.org/10.1145/2858036.2858499
8. Heiko Drewes and Albrecht Schmidt. 2007. *Interacting with the computer using gaze gestures*. Springer-Verlag. 475–488 pages.
9. Andrew T Duchowski, Nathan Cournia, and Hunter Murphy. 2004. Gaze-contingent displays: a review. *Cyberpsychology & behavior : the impact of the Internet, multimedia and virtual reality on behavior and society* 7, 6 (dec 2004), 621–34. DOI: http://dx.doi.org/10.1089/cpb.2004.7.621
10. David Fono and Roel Vertegaal. 2005. EyeWindows: evaluation of eye-controlled zooming windows for focus selection. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '05*. ACM Press, New York, New York, USA, 151. DOI: http://dx.doi.org/10.1145/1054972.1054994
11. Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost de Weijer. 2011. *Eye tracking: A comprehensive guide to methods and measures*. OUP Oxford.
12. Yu-Chun Huang and Fong-Gong Wu. 2015. Visual and manual loadings with QWERTY-like ambiguous keyboards: Relevance of letter-key assignments on mobile phones. *International Journal of Industrial Ergonomics* 50 (2015), 143–150. DOI: http://dx.doi.org/10.1016/j.ergon.2015.08.002
13. Aulikki Hyrskykari, Päivi Majaranta, Antti Aaltonen, and Kari-Jouko Räihä. 2000. Design issues of iDICT: A Gaze-Assisted Translation Aid. In *Proceedings of the symposium on Eye tracking research & applications - ETRA '00*. ACM Press, New York, New York, USA, 9–14. DOI:http://dx.doi.org/10.1145/355017.355019
14. SensoMotoric Instruments. 2016. SMI REDn Scientific – Specifiactions. (2016). http://www.smivision.com/fileadmin/user
15. Howell Istance and Aulikki Hyrskykari. 2011. Gaze-Aware Systems and Attentive Applications. In *Gaze Interaction and Applications of Eye Tracking*. Vol. 58. IGI Global, 175–195. DOI: http://dx.doi.org/10.4018/978-1-61350-098-9.ch013
16. Howell Istance, Aulikki Hyrskykari, Lauri Immonen, Santtu Mansikkamaa, and Stephen Vickers. 2010. Designing gaze gestures for gaming: an Investigation of Performance. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications - ETRA '10*. ACM Press, New York, New York, USA, 323. DOI: http://dx.doi.org/10.1145/1743666.1743740
17. Robert J. K. Jacob. 1991. The use of eye movements in human-computer interaction techniques: what you look at is what you get. *ACM Transactions on Information Systems* 9, 2 (Apr 1991), 152–169. DOI: http://dx.doi.org/10.1145/123078.128728
18. Shahram Jalaliniya and Diako Mardanbegi. 2016. Eyegrip: Detecting targets in a series of uni-directional moving objects using optokinetic nystagmus eye movements. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 5801–5811.
19. J Jimenez, D Gutierrez, and P Latorre. 2008. Gaze-based Interaction for Virtual Environments. *Journal of Universal Computer Science 14* 19 (2008), 3085–3098. http://giga.cps.unizar.es/
20. Manu Kumar, Tal Garfinkel, Dan Boneh, and Terry Winograd. 2007. Reducing shoulder-surfing by using gaze-based password entry. In *Proceedings of the 3rd symposium on Usable privacy and security - SOUPS '07*. ACM Press, New York, New York, USA, 13. DOI: http://dx.doi.org/10.1145/1280680.1280683

21. Manu Kumar, Jeff Klingner, Rohan Puranik, Terry Winograd, and Andreas Paepcke. 2008. Improving the accuracy of gaze input for interaction. In *Proceedings of the 2008 symposium on Eye tracking research & applications - ETRA '08*. ACM Press, New York, New York, USA, 65. DOI: http://dx.doi.org/10.1145/1344471.1344488

22. Manu Kumar, Andreas Paepcke, and Terry Winograd. 2007. EyePoint: practical pointing and selection using gaze and keyboard. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*. ACM Press, New York, New York, USA, 421. DOI: http://dx.doi.org/10.1145/1240624.1240692

23. Andrew Kurauchi, Wenxin Feng, Ajjen Joshi, Carlos Morimoto, and Margrit Betke. 2016. EyeSwipe: Dwell-free Text Entry Using Gaze Paths. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, New York, New York, USA, 1952–1956. DOI: http://dx.doi.org/10.1145/2858036.2858335

24. Päivi Majaranta, Ulla-Kaija Ahola, and Oleg Špakov. 2009. Fast gaze typing with an adjustable dwell time. In *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*. ACM Press, New York, New York, USA, 357. DOI: http://dx.doi.org/10.1145/1518701.1518758

25. Päivi Majaranta, Hirotaka Aoki, Mick Donegan, Dan Witzner Hansen, and John Paulin Hansen. 2011. *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies*. Information Science Reference - Imprint of: IGI Publishing.

26. Päivi Majaranta and Andreas Bulling. 2014. Eye Tracking and Eye-Based Human-Computer Interaction. Springer London, 39–65. DOI: http://dx.doi.org/10.1007/978-1-4471-6392-3_3

27. Michael Mauderer, Simone Conte, Miguel A. Nacenta, and Dhanraj Vishwanath. 2014. Depth perception with gaze-contingent depth of field. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*. ACM Press, New York, New York, USA, 217–226. DOI: http://dx.doi.org/10.1145/2556288.2557089

28. Msi. 2016. msi GT72S G TOBII (6TH GEN) (GTX 980M). (2016). https://us.msi.com/Laptop/GT72S-G-Tobii-6th-Gen-GTX-980M.html

29. Cuong Nguyen and Feng Liu. 2016. Gaze-based Notetaking for Learning from Lecture Videos. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, New York, New York, USA, 2093–2097. DOI: http://dx.doi.org/10.1145/2858036.2858137

30. Marcus Nyström, Richard Andersson, Kenneth Holmqvist, and Joost van de Weijer. 2013. The influence of calibration method and eye physiology on eyetracking data quality. *Behavior Research Methods* 45, 1 (mar 2013), 272–288. DOI: http://dx.doi.org/10.3758/s13428-012-0247-4

31. Kristien Ooms, Lien Dupont, Lieselot Lapon, and Stanislav Popelka. 2015. Accuracy and precision of fixation locations recorded with the low-cost Eye Tribe tracker in different experimental set-ups. *Journal of Eye Movement Research* 8, 1 (2015). DOI: http://dx.doi.org/10.16910/JEMR.8.1.5

32. Håkon Raudsandmoen and Børge Rødsjø. 2012. Empirically Based Design Guidelines for Gaze Interaction in Windows 7. (2012). http://urn.kb.se/resolve?urn=urn:nbn:no:ntnu:diva-19544

33. David Rozado, Javier S. Agustin, Francisco B. Rodriguez, and Pablo Varona. 2012. Gliding and saccadic gaze gesture recognition in real time. *ACM Transactions on Interactive Intelligent Systems* 1, 2 (jan 2012), 1–27. DOI: http://dx.doi.org/10.1145/2070719.2070723

34. Dario D. Salvucci and Joseph H. Goldberg. 2000. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the symposium on Eye tracking research & applications - ETRA '00*. ACM Press, New York, New York, USA, 71–78. DOI: http://dx.doi.org/10.1145/355017.355028

35. Simon Schenk, Philipp Tiefenbacher, Gerhard Rigoll, and Michael Dorr. 2016. SPOCK: A Smooth Pursuit Oculomotor Control Kit. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16*. ACM Press, New York, New York, USA, 2681–2687. DOI: http://dx.doi.org/10.1145/2851581.2892291

36. Julia Schwarz, Scott Hudson, Jennifer Mankoff, and Andrew D. Wilson. 2010. A framework for robust and flexible handling of inputs with uncertainty. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology - UIST '10*. ACM Press, New York, New York, USA, 47. DOI: http://dx.doi.org/10.1145/1866029.1866039

37. Baris Serim and Giulio Jacucci. 2016. Pointing while Looking Elsewhere: Designing for Varying Degrees of Visual Guidance during Manual Input. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (2016), 5789–5800. DOI: http://dx.doi.org/10.1145/2858036.2858480

38. John L. Sibert, Mehmet Gokturk, and Robert A. Lavine. 2000. The reading assistant: eye gaze triggered auditory prompting for reading remediation. In *Proceedings of the 13th annual ACM symposium on User interface software and technology - UIST '00*. ACM Press, New York, New York, USA, 101–107. DOI: http://dx.doi.org/10.1145/354401.354418

39. Oleg Špakov. 2012. Comparison of eye movement filters used in HCI. In *Proceedings of the Symposium on Eye Tracking Research and Applications - ETRA '12*. ACM Press, New York, New York, USA, 281. DOI: http://dx.doi.org/10.1145/2168556.2168616

40. 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 Press, New York, New York, USA, 287–290. DOI: http://dx.doi.org/10.1145/2857491.2857526

41. Dave M Stampe. 1993. Heuristic filtering and reliable calibration methods for video-based pupil-tracking systems. *Behavior Research Methods, Instruments, & Computers* 25, 2 (1993), 137–142.

42. Thomas Templier, Kenan Bektas, and Richard H.R. Hahnloser. 2016. Eye-Trace: Segmentation of Volumetric Microscopy Images with Eyegaze. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, New York, New York, USA, 5812–5823. DOI: http://dx.doi.org/10.1145/2858036.2858578

43. The Eye Tribe. 2016. Eye Tribe Tracker Pro. (2016). https://theeyetribe.com/products/

44. Tobii Technology. 2016a. EyeX controller accurary and precision test report, tobii developer zone. (2016). http://developer.tobii.com/community/forums/topic/eyex-controller-accurary-and-precision-test-report/

45. Tobii Technology. 2016b. Tobii EyeX Controller – Specifications. (2016). http://www.tobii.com/xperience/products/

46. 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 Press, New York, New York, USA, 439. DOI:http://dx.doi.org/10.1145/2493432.2493477

47. Jacob O. Wobbrock, James Rubinstein, Michael W. Sawyer, and Andrew T. Duchowski. 2008. Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In *Proceedings of the 2008 symposium on Eye tracking research & applications - ETRA '08*. ACM Press, New York, New York, USA, 11. DOI: http://dx.doi.org/10.1145/1344471.1344475

48. Erroll Wood and Andreas Bulling. 2014. EyeTab: model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications - ETRA '14*. ACM Press, New York, New York, USA, 207–210. DOI: http://dx.doi.org/10.1145/2578153.2578185

49. Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the SIGCHI conference on Human factors in computing systems the CHI is the limit - CHI '99*. ACM Press, New York, New York, USA, 246–253. DOI: http://dx.doi.org/10.1145/302979.303053

Appendix 1: Optimal parameters
1€ Filter

Note: parameter values are optimized with respect to input signal (gaze points) given in cm. The characteristics of the filter may lead to different values if other units are used, such as pixel, which lead to much larger input values.

x-direction

β	f_{cmin}	delay (frames)	target width (cm)
1	0.78	0.5	2.68
0.54	0.93	1.0	2.54
0.34	0.78	1.5	2.47
0.21	0.95	2.0	2.36
0.15	0.95	2.5	2.29
0.11	0.94	3.0	2.26
0.09	0.8	3.5	2.22
0.1	0.2	4.0	2.17
0.08	0.24	4.5	2.13
0.05	0.67	5.0	2.12

y-direction

β	f_{cmin}	delay (frames)	target width (cm)
0.99	0.87	0.50	3.00
0.53	0.93	1.00	2.89
0.33	0.94	1.50	2.82
0.23	0.79	1.98	2.75
0.17	0.79	2.50	2.70
0.12	0.87	3.00	2.65
0.09	0.84	3.49	2.61
0.07	0.85	4.00	2.59
0.07	0.46	4.49	2.56

Metrics Stampe Filter

x-direction

Step	target width (cm)	delay (frames)
1	2.557639	1.489
2	2.38125	3.028

y-direction

Step	target height (cm)	delay (frames)
1	2.980972	1.545
2	2.786944	3.152

Optimal parameters
Weighted Average Filter

x-direction

	Kernel: Constant		Kernel: Triangle		Kernel: Gaussian	
window size (frames)	target width (cm)	delay (frames)	target width (cm)	delay (frames)	target width (cm)	delay (frames)
2	2.95	0.6	2.96	0.2	3.16	0.0
3	2.73	1.0	2.79	0.5	3.03	0.1
4	2.59	1.5	2.65	0.9	2.79	0.4
5	2.52	2.0	2.59	1.2	2.68	0.8
6	2.45	2.6	2.54	1.6	2.61	1.1
7	2.40	3.0	2.49	1.9	2.56	1.4
8	2.36	3.5	2.43	2.3	2.52	1.8
9	2.31	4.1	2.40	2.6	2.49	2.1
10	2.26	4.6	2.36	2.9	2.43	2.4
11	2.22	5.1	2.35	3.2	2.42	2.7
12	2.19	5.6	2.31	3.6	2.38	3.0
13	2.15	6.1	2.28	4.0	2.35	3.3
14	2.13	6.7	2.26	4.2	2.33	3.6
15	2.12	7.1	2.22	4.6	2.31	4.0
16	2.10	7.7	2.19	5.0	2.26	4.3
17	2.06	8.3	2.17	5.5	2.24	4.8
18	2.06	8.9	2.17	5.9	2.22	5.1
19	2.05	9.3	2.15	6.1	2.20	5.5

y-direction

	Kernel: Constant		Kernel: Triangle		Kernel: Gaussian	
window size (frames)	target height (cm)	delay (frames)	target height (cm)	delay (frames)	target height (cm)	delay (frames)
2	3.32	0.6	3.37	0.3	3.49	0.0
3	3.12	1.0	3.18	0.5	3.39	0.2
4	2.98	1.5	3.03	0.9	3.18	0.4
5	2.89	2.0	2.96	1.2	3.09	0.9
6	2.84	2.6	2.89	1.6	2.98	1.2
7	2.77	3.1	2.84	2.0	2.93	1.5
8	2.70	3.8	2.80	2.4	2.88	1.9
9	2.66	4.3	2.75	2.8	2.84	2.2
10	2.63	4.9	2.70	3.2	2.79	2.6
11	2.59	5.4	2.68	3.5	2.73	3.0
12	2.56	5.9	2.65	3.9	2.72	3.3
13	2.54	6.4	2.63	4.3	2.68	3.6
14	2.52	6.9	2.59	4.7	2.66	4.0
15	2.50	7.4	2.58	4.9	2.65	4.3
16	2.50	7.9	2.56	5.3	2.63	4.7
17	2.50	8.6	2.54	5.7	2.59	5.1
18	2.49	9.2	2.54	6.1	2.59	5.4
19	2.47	9.6	2.52	6.5	2.58	5.7

Optimal parameters
Weighted Average Filter with Saccade Detection

x-direction

Kernel: Constant

saccade threshold (cm)	max. fixation window (frames)	target width (cm)	delay (frames)
2.1	2	2.84	0.09
1.5	15	2.66	0.20
1.75	12	2.54	0.30
1.7	36	2.42	0.39
1.75	38	2.36	0.47
1.8	37	2.35	0.58
1.85	35	2.33	0.68
1.85	35	2.33	0.68
1.9	36	2.31	0.90
1.90	36.0	2.31	0.90

Kernel: Triangle

saccade threshold (cm)	max. fixation window (frames)	target width (cm)	delay (frames)
1.3	11	2.82	0.10
1.5	33	2.59	0.20
1.7	37	2.42	0.29
1.8	37	2.40	0.34
1.8	39	2.38	0.46
1.8	39	2.38	0.46
1.9	35	2.36	0.69
1.9	35	2.36	0.69
2.0	39	2.35	0.86
2.0	39	2.35	0.86

y-direction

Kernel: Constant

saccade threshold (cm)	max. fixation window (frames)	target height (cm)	delay (frames)
1.20	5.0	3.21	0.08
1.75	6	3.02	0.20
1.7	12	2.93	0.30
1.75	15	2.86	0.38
1.7	39	2.75	0.50
1.8	36	2.72	0.60
1.85	38.0	2.68	0.69
1.95	34.0	2.65	0.76
1.95	37.0	2.63	0.85
2.00	39.0	2.59	0.94

Kernel: Triangle

saccade threshold (cm)	max. fixation window (frames)	target height (cm)	delay (frames)
1.65	4.0	3.14	0.10
1.3	35	2.96	0.19
1.6	38	2.86	0.30
1.7	39	2.82	0.38
1.85	31	2.75	0.46
1.9	36	2.72	0.54
2.00	40.0	2.66	0.67
2.00	40.0	2.66	0.67
2.10	40.0	2.65	0.85
2.10	40.0	2.65	0.85

Kernel: Gaussian

saccade threshold (cm)	max. fixation window (frames)	target width (cm)	delay (frames)
1.4	23.00	2.68	0.10
1.6	34.00	2.52	0.20
1.7	37.00	2.45	0.28
1.8	39.00	2.43	0.31
1.9	35.00	2.42	0.42
1.9	39.00	2.38	0.58
1.9	39.00	2.38	0.58
1.9	39.00	2.38	0.58
2.0	32.00	2.36	0.84
2.0	36.00	2.35	0.93

Kernel: Gaussian

saccade threshold (cm)	max. fixation window (frames)	target height (cm)	delay (frames)
1.15	22	3.14	0.10
1.65	20	2.91	0.20
1.75	36	2.82	0.29
1.85	38	2.77	0.38
1.95	35	2.73	0.49
2	37	2.72	0.55
2.05	35	2.70	0.65
2.1	35	2.68	0.72
2.15	40	2.66	0.84
2.2	39	2.65	1.00

Optimal parameters
Weighted Average Filter with Saccade Detection and Outlier Correction

x-direction

Kernel: Constant

saccade threshold (cm)	max. fixation window (frames)	target width (cm)	delay (frames)
2.7	1	2.84	1.10
2.45	1	2.82	1.19
2.5	2	2.65	1.30
1.9	2	2.65	1.39
1.85	3	2.54	1.45
1.45	6	2.40	1.59
1.45	8	2.33	1.69
1.35	11	2.28	1.80
1.3	17	2.22	1.90
1.4	17	2.19	2.00
1.45	20	2.13	2.09
1.40	30	2.10	2.20
1.45	35	2.05	2.28
1.45	40	2.01	2.33
1.50	38	1.99	2.45

Kernel: Triangle

saccade threshold (cm)	max. fixation window (frames)	target width (cm)	delay (frames)
2.7	1	2.88	0.98
2.6	2	2.70	1.07
2.5	2	2.70	1.18
2.4	3	2.58	1.28
2.1	3	2.58	1.37
2.0	5	2.47	1.50
1.5	8	2.38	1.59
1.2	14	2.33	1.70
1.3	20	2.22	1.80
1.3	32	2.15	1.89
1.35	39	2.10	1.99
1.45	38	2.06	2.09
1.55	37	2.03	2.30

y-direction

Kernel: Constant

saccade threshold (cm)	max. fixation window (frames)	target height (cm)	delay (frames)
3.05	1	3.25	1.09
2.7	1	3.23	1.20
2.85	2	3.07	1.30
1.55	2	3.05	1.40
1.9	3	2.91	1.46
2.35	4	2.80	1.59
1.8	6	2.72	1.70
1.3	8	2.72	1.80
1.4	11	2.65	1.89
1.5	13	2.61	1.98
1.55	15	2.58	2.09
1.25	36	2.54	2.18
1.30	39	2.50	2.30
1.35	40	2.47	2.36
1.50	35	2.43	2.44

Kernel: Triangle

saccade threshold (cm)	max. fixation window (frames)	target height (cm)	delay (frames)
3.1	2	3.10	0.98
3.0	2	3.10	1.08
2.7	2	3.10	1.20
2.8	3	2.95	1.27
2.9	4	2.89	1.40
2.0	5	2.80	1.50
1.6	7	2.75	1.59
1.6	9	2.70	1.70
1.3	15	2.65	1.80
1.8	13	2.59	1.90
1.25	26	2.54	2.00
1.35	35	2.47	2.09
1.55	39	2.43	2.16
1.65	40	2.42	2.30

Kernel: Gaussian

saccade threshold (cm)	max. fixation window (frames)	target width (cm)	delay (frames)
2.7	3	2.72	0.97
2.6	3	2.72	1.06
2.4	3	2.70	1.19
2.5	4	2.61	1.28
2.4	5	2.54	1.37
2.4	6	2.47	1.48
1.4	11	2.36	1.60
1.4	16	2.26	1.70
1.3	28	2.19	1.79
1.4	37	2.12	1.88
1.45	36	2.10	1.97
1.55	38	2.06	2.15

Kernel: Gaussian

saccade threshold (cm)	max. fixation window (frames)	target height (cm)	delay (frames)
3.1	3	s	0.99
3.0	3	3.12	1.06
3.0	4	3.00	1.20
2.7	4	2.98	1.29
2.6	5	2.91	1.40
1.7	7	2.82	1.49
1.6	10	2.72	1.59
1.3	17	2.65	1.70
1.2	36	2.61	1.79
1.3	40	2.54	1.88
1.35	39	2.49	1.99
1.65	40	2.45	2.06
1.85	39	2.43	2.20

Appendix 2:
Accuracy and precision of target fixations

Shown are mean values and standard deviations for different conditions investigated in the paper.
Results were computed in cm. Conversion in degrees of visual angle is only given for reference in relation to central line of sight, assuming a viewing distance of 65 cm.
[Reference: K. Holmqvist, M. Nyström, et al. 2011. Eye tracking: A comprehensive guide to methods and measures. OUP Oxford.]

	Accuracy								Precision							
	X (cm)		X (degree)		Y (cm)		Y (degree)		X (cm)		X (degree)		Y (cm)		Y (degree)	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Overall	0.47	0.36	0.829	0.635	0.58	0.29	1.022	0.511	0.53	0.43	0.934	0.758	0.51	0.35	0.899	0.617
Percentile																
- 25%	0.15		0.264		0.2		0.353		0.19		0.335		0.22		0.388	
- 50%	0.31		0.547		0.45		0.793		0.32		0.564		0.32		0.564	
- 75%	0.58		1.022		0.78		1.375		0.57		1.005		0.58		1.022	
- 90%	0.93		1.639		1.19		2.098		1.1		1.939		1.04		1.833	
Tracker																
SMI REDn	0.59	0.41	1.040	0.723	0.67	0.25	1.181	0.441	0.53	0.49	0.934	0.864	0.53	0.35	0.934	0.617
Tobii EyeX	0.36	0.26	0.635	0.458	0.48	0.3	0.846	0.529	0.53	0.35	0.934	0.617	0.49	0.35	0.864	0.617
Light																
Artificial	0.53	0.4	0.934	0.705	0.65	0.35	1.146	0.617	0.61	0.54	1.075	0.952	0.58	0.44	1.022	0.776
Daylight	0.42	0.32	0.740	0.564	0.5	0.19	0.881	0.335	0.44	0.24	0.776	0.423	0.44	0.21	0.776	0.370
Row																
1	0.51	0.72	0.899	1.269	0.58	0.4	1.022	0.705	0.5	0.6	0.881	1.058	0.38	0.44	0.670	0.776
2	0.38	0.34	0.670	0.599	0.55	0.36	0.970	0.635	0.35	0.36	0.617	0.635	0.43	0.32	0.758	0.564
3	0.42	0.45	0.740	0.793	0.54	0.31	0.952	0.547	0.45	0.44	0.793	0.776	0.48	0.31	0.846	0.547
4	0.49	0.47	0.864	0.829	0.61	0.39	1.075	0.688	0.57	0.51	1.005	0.899	0.54	0.4	0.952	0.705
5	0.6	0.49	1.058	0.864	0.59	0.4	1.040	0.705	0.84	0.67	1.481	1.181	0.75	0.72	1.322	1.269

Column																
1	0.48	0.47	0.846	0.829	0.6	0.41	1.058	0.723	0.46	0.46	0.811	0.811	0.55	0.45	0.970	0.793
2	0.45	0.39	0.793	0.688	0.54	0.3	0.952	0.529	0.44	0.29	0.776	0.511	0.5	0.33	0.881	0.582
3	0.42	0.47	0.740	0.829	0.6	0.4	1.058	0.705	0.51	0.52	0.899	0.917	0.52	0.43	0.917	0.758
4	0.4	0.36	0.705	0.635	0.57	0.3	1.005	0.529	0.46	0.31	0.811	0.547	0.48	0.3	0.846	0.529
5	0.5	0.43	0.881	0.758	0.57	0.35	1.005	0.617	0.5	0.42	0.881	0.740	0.48	0.38	0.846	0.670
6	0.58	0.46	1.022	0.811	0.58	0.31	1.022	0.547	0.77	0.89	1.357	1.569	0.54	0.41	0.952	0.723