Improving the Accuracy of Gaze Input for Interaction

Manu Kumar
GazeWorks, Inc.
P.O. Box 901
Palo Alto, CA 94302-0901
manu@gazeworks.com

Jeff Klingner, Rohan Puranik, Terry Winograd, Andreas Paepcke
Stanford University
Gates Building, 353 Serra Mall
Stanford, CA 94305, USA
{klingner, rpuranik, winograd, paepcke}@stanford.edu

Abstract

Using gaze information as a form of input poses challenges based on the nature of eye movements and how we humans use our eyes in conjunction with other motor actions. In this paper, we present three techniques for improving the use of gaze as a form of input. We first present a saccade detection and smoothing algorithm that works on real-time streaming gaze information. We then present a study which explores some of the timing issues of using gaze in conjunction with a trigger (key press or other motor action) and propose a solution for resolving these issues. Finally, we present the concept of Focus Points, which makes it easier for users to focus their gaze when using gaze-based interaction techniques. Though these techniques were developed for improving the performance of gaze-based pointing, their use is applicable in general to using gaze as a practical form of input.

CCS: H.5.2 [Information interfaces and presentation]: User Interfaces. – Input devices and strategies.

General terms: Human Factors, Algorithms, Performance, Design

Keywords: Eye Tracking, Gaze Input, Gaze-enhanced User Interface Design, Fixation Smoothing, Eye-hand coordination, Focus Points.

1. Introduction

The eyes are a rich source of information for gathering context in our everyday lives. A user's gaze is postulated to be the best proxy for attention or intention [Zhai 2003]. Using eye-gaze information as a form of input can enable a computer system to gain more contextual information about the user's task, which in turn can be leveraged to design interfaces which are more intuitive and intelligent. Recent research has explored the use of gaze information as a practical form of input by augmenting rather than replacing existing interaction techniques. Kumar et al have developed gaze-enhanced interaction techniques for pointing and selection [Kumar et al. 2007b], scrolling [Kumar et al. 2007c], password entry [Kumar et al. 2007a] and other everyday computing tasks.

In the paper describing EyePoint [Kumar et al. 2007b], it was reported that while the speed of a gaze-based pointing technique was comparable to the mouse, error rates were significantly higher. To address this problem we conducted a series of studies to better understand the source of these errors and identify ways to improve the accuracy of gaze-based pointing.

In this paper we present three methods for improving the accuracy and user experience of gaze-based pointing: an algorithm for real-time saccade detection and fixation smoothing, an algorithm for improving eye-hand coordination, and the use of focus points. These methods boost the basic performance for using gaze information in interactive applications and in our applications made the difference between prohibitively high error rates and practical usefulness of gaze-based interaction.

2. Real-Time Saccade Detection And Fixation Smoothing

Basic eye movements can be broken down into two types: *fixations* and *saccades*. A fixation occurs when the gaze rests steadily on a single point. A saccade is a fast movement of the eye between two fixations. However, even fixations are not stable and the eye jitters during fixations due to drift, tremor and involuntary micro-saccades [Yarbus 1967]. This gaze jitter, together with the limited accuracy of eye trackers, results in a noisy gaze signal.

The prior work on algorithms for identifying fixations and saccades [Monty et al. 1978, Salvucci 1999, Salvucci and Goldberg 2000, Yarbus 1967] has dealt mainly with post-processing previously captured gaze information. For using gaze information as a form of input, it is necessary to analyze eye-movement data in real-time.

To smooth the data from the eye tracker in real-time, it is necessary to determine whether the most recent data point is the beginning of a saccade, a continuation of the current fixation or an outlier relative to the current fixation. We use a gaze movement threshold, in which two gaze points separated by a Euclidean distance of more than a given saccade threshold are labeled as a saccade. This is similar to the velocity threshold technique described in [Salvucci and Goldberg 2000], with two modifications to make it more robust to noise. First, we measure the displacement of each eye movement relative to the current estimate of the fixation location rather than to the previous measurement. Second, we look ahead one measurement and reject movements over the saccade threshold which immediately return to the current fixation. This prevents single outliers of the current fixation from being mislabeled as saccades. It should be noted that this look-ahead introduces a one-measurement latency (20ms for the Tobii 1750 eye tracker [Tobii Technology 2006]) at saccade thresholds into the gaze data provided to the application.

ETRA 2008, Savannah, Georgia, March 26–28, 2008.
© 2008 ACM 978-1-59593-982-1/08/0003 $5.00

The algorithm maintains two sets of points: the current fixation window and a potential fixation window. If a point is close to (within a saccade threshold) the current fixation, then it is added to the current fixation window. The new current fixation is calculated by a weighted mean which favors more recent points (described below). If the point differs from the current fixation by more than a saccade threshold, then it is added to the potential fixation window and the current fixation is returned. When the next data point is available, if it was closer to the current fixation, then we add it to the current fixation and throw away the potential fixation as an outlier. If the data point is closer to the potential fixation, then we add the point to the potential fixation window and make this the new current fixation.

The fixation point is calculated as a weighted mean (a one-sided triangular filter) of the set of points in the fixation window. The weight assigned to each point is based on its position in the window. For a window with n points (P_0, P_1… P_{n-1}) the mean fixation would be calculated by the formula:

$$P_{fixation} = \frac{1P_0 + 2P_1 + ... + nP_{n-1}}{(1 + 2 + ... + n)}$$

The size of the fixation window (n) is capped to include only data points that occurred within a dwell duration [Majaranta et al. 2004, Majaranta et al. 2003] of 400-500ms. We do this to allow the fixation point to adjust more rapidly to slight drifts in the gaze data.

Figure 1 shows the output from the smoothing algorithm for the x-coordinate of the eye-tracking data. We also show a Kalman filter applied to the entire raw gaze data and a Kalman filter applied in parts to the fixations only. A Kalman Filter applied over the entirety of the raw gaze data smoothes over saccade intervals. The nature of eye movements, in particular the existence of saccades, necessitates that the smoothing function only be applied to fixations, i.e. within saccade boundaries. Applying the Kalman filter in parts to the fixations only does yield comparable results to our one-sided triangular filter discussed above. It is possible that applying a non-linear variant of the Kalman filter [Arulampalam et al. 2002], or a better process model of eye movements for the Kalman filter may yield better smoothing results. The advantages of our approach relative to Kalman filters are simplicity and a design specific to eye movements, which can tolerate the noise typically occurring in eye tracking data.

While there is still room for improvement in the algorithm above by taking into account the directionality of the incoming data points, we found that our saccade detection and smoothing algorithm significantly improved the reliability of the results for applications which rely on the real-time use of eye-tracking data. A more detailed description of our algorithm including pseudocode is available in [Kumar 2007].

.

3. Eye-Hand Coordination

Previous research on using a combination of gaze and keyboard for performing a pointing task [Kumar et al. 2007b] showed that error rates were very high. Additional work on using gaze-based password entry [Kumar et al. 2007a] showed that the high error rates existed only when the subjects used a combination of gaze

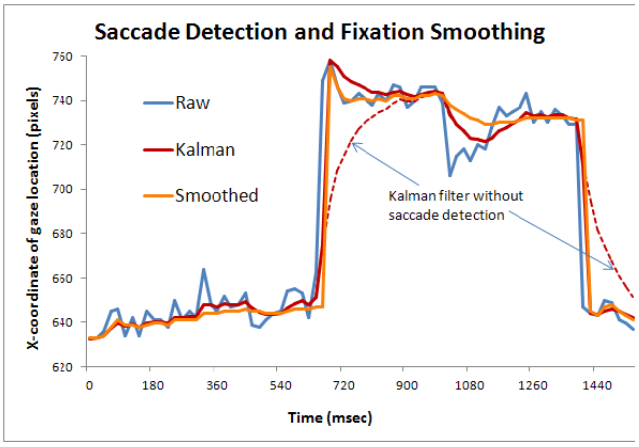

Figure 1. Results of our real-time saccade detection and smoothing algorithm. Note that the one measurement look-ahead prevents outliers in the raw gaze data from being mistaken for saccades, but introduces a 20ms latency on saccade thresholds.

plus a keyboard trigger. Using a dwell-based trigger exhibited minimal errors. These observations led us to hypothesize that the errors may be caused by a failure of synchronization between gaze and triggers.

To determine the cause and the number of errors, we conducted two user studies with 15 subjects (11 male, 4 female, average age 26 years). In the first study, subjects were presented with a red balloon. Each time they looked at the balloon and pressed the trigger key, the red balloon popped and moved to a new location (Moving Target Study). In the second study, subjects were presented with 20 numbered balloons on the screen and asked to look at each balloon in order and press the trigger key (Stationary Targets Study). Subjects repeated each study twice, once optimizing for speed and trying to perform the task as quickly as possible and the second time optimizing for accuracy and trying to perform the study as accurately as possible. The order of the studies was varied for counter-balancing and trials were repeated in case of an error.

An in-depth analysis of the data and classification of the errors from the two studies revealed multiple sources of error:

Tracking errors: caused due to the eye tracker accuracy. These include cases in which the gaze data from the eye tracker is biased or when the location of the target closer to the periphery of the screen results in lower accuracy from the eye tracker [Beinhauer 2006].

Early-Trigger errors: caused because the trigger happened before the user's gaze was in the target area. Early triggers can happen because a) the eye tracker introduces a sensor lag of about 33ms in processing the user's eye gaze, b) the smoothing algorithm introduces an additional latency of 20ms at saccade thresholds c) in some cases (as in the Moving Target study) the users may have only looked at the target in their peripheral vision and pressed the trigger before they actually focused on the target.

Late-Trigger errors: caused because the user had already moved their gaze on to the next target before they pressed the trigger. Late triggers can happen only in cases when multiple targets are

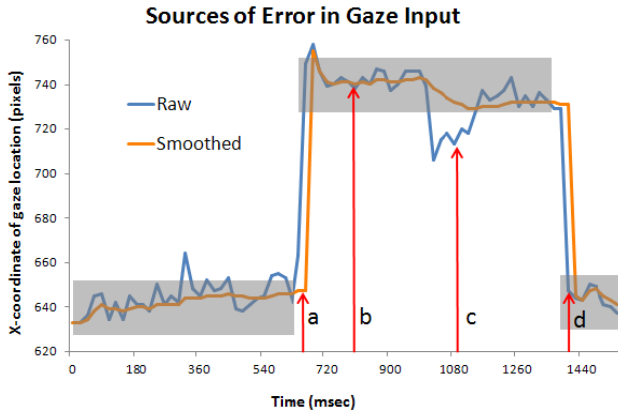

Figure 2. Sources of error in gaze input. Shaded areas show the target regions. Example triggers are indicated by red arrows. The triggers shown are all different attempts to click on the upper target region. The trigger points correspond to: a) early trigger error, b) raw hit and smooth hit, c) raw miss and smooth hit, and d) late trigger error.

visible on the screen, as in the Stationary Targets study or in gaze-based typing.

Other errors: these include a) smoothing errors caused when the smoothed data happened to be outside the target boundary, but the raw data point would have, by chance, resulted in a hit, b) human errors where the subject just was not looking at the right thing or the subject looked down at the keyboard before pressing the trigger.

Figure 2 illustrates the different error types. Figure 3 shows how often each type of error occurred in the two studies.

To improve the accuracy of gaze-based pointing in the case of the speed task, we implemented an early trigger correction (ETC) algorithm which delays trigger points by 80ms to account for the sensor lag, smoothing latency and peripheral vision effects. We simulated this algorithm over the data from the Moving Target

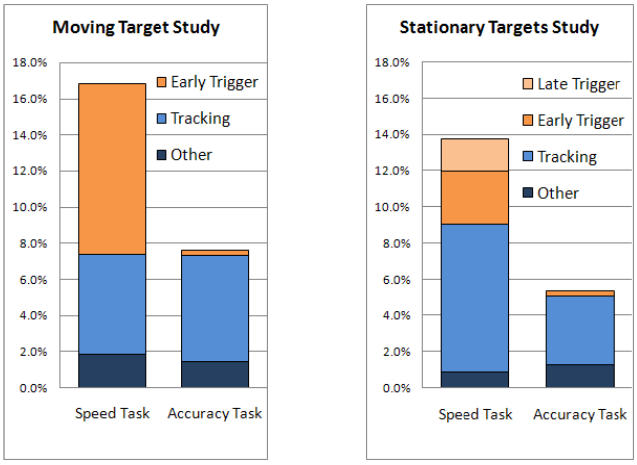

Figure 3. Analysis of errors in the two studies show that a large number of errors in the Speed Task happen due to early triggers and late triggers – errors in synchronization between the gaze and trigger events.

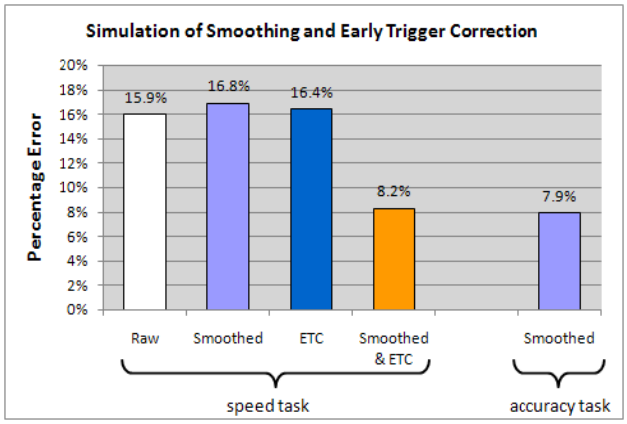

Figure 4. Simulation of smoothing and early trigger correction (ETC) on the speed task for the Moving Target Study shows that the percentage error of the speed task decreases significantly and is comparable to the error rate of the accuracy task.

study. Figure 4 shows the outcome from the simulated results. It should be noted that both smoothing and early-trigger correction by themselves actually increased the error rate, because smoothing introduces a latency that the early trigger would be correcting. The error rate in the speed task when using a combination of smoothing and early trigger correction approaches the error rate of the accuracy task — without compromising the speed of the task.

While we were able to identify late-trigger errors in the analysis of the data, it is difficult to distinguish a late trigger from an early trigger or even an on-time trigger without using information about the location of the targets. Since our approach has focused on providing a generally applicable techniques for gaze-input which do not rely on application or operating system specific information we did not attempt to correct for late triggers, though we note that the use of information about target locations has the potential to significantly improve the accuracy of gaze-based input by allowing the current fixation to be applied to the closest target.

4. Focus Points

In previous work on gaze-based pointing [Kumar et al. 2007b], pointing was aided by the use of *Focus Points*—a grid pattern of dots overlaid on the magnified view that contained the targets (see Figure 5). It was hypothesized that focus points assist the user in making a more fine-grained selection by focusing the user's gaze, thereby improving the accuracy of the eye tracking. However, the studies in the EyePoint paper showed no conclusive effect of an improvement in tracking accuracy when using focus points.

To test this hypothesis further, we conducted a user study with 17 subjects (11 male, 6 female, average age 22). In the first part of the study subjects were shown a red balloon and asked to look at the center of the balloon. Once they had looked at the balloon for a dwell duration (450ms) the balloon automatically moved to a new location. In the second part, subjects repeated the study, but with the center point of the balloon clearly marked with a focus point. The order was varied and each subject was shown 40 balloons. The user's raw and smoothed gaze positions were logged for each balloon. At the end of the study users were presented with a 7-point Likert scale questionnaire which asked them which condition was easier and whether they found the focus point at the center of the balloon useful.

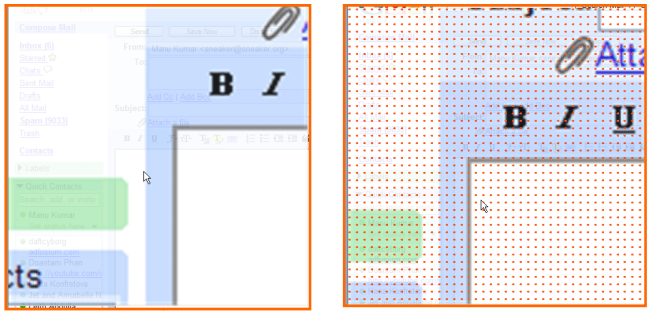

Figure 5. Magnified view for gaze-based pointing technique with and without focus points. Using focus points provides a visual anchor for subjects to focus their gaze on making it easier for them to click in the text box.

We computed the standard deviation of the Euclidean distance of each gaze point from the center point of the target. The results from the study show that within the bounds of the measurable accuracy of the eye tracker[1], the use of focus points did not have a significant impact in concentrating the user's gaze on the center of the target. The questionnaire results however, indicate that subjects found the condition with the focus point easier and found the focus point to be useful when trying to look at the center of the target. These results are consistent with the findings in the original EyePoint paper [Kumar et al. 2007b].

We conclude that while the use of focus points may not measurably improve the accuracy of the raw gaze data from the eye tracker, they do indeed make pointing easier and provide a better user experience. This is illustrated by Figure 5, which shows two views of the magnified view from EyePoint. If the subject intends on clicking in the text area in the bottom right of the magnified view, the task is easier for the subject in the condition with focus points, since the focus points provide a visual anchor for the subject to focus upon.

5. Conclusion

The techniques presented above improve the use of gaze for input by addressing changes in how we interpret the gaze data from eye trackers (real-time saccade detection and smoothing), how to match gaze input with an external trigger (eye-hand coordination) and by introducing features which make it easier for the user to look at the desired target (focus points). The above techniques describe software changes that can be made at the operating system layer to improve the use or gaze as a form of input and are orthogonal to any improvement in the underlying tracking technology that provides for more accuracy and head movement from the eye tracker.

[1] The accuracy of the eye tracker is approximately 1˚ of visual angle which provides a spread of 33 pixels in any direction (diameter of spread ~66 pixels)

References

ARULAMPALAM, M.S., MASKELL, S., GORDON, N., AND CLAPP, T. 2002. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*. 50(2): p. 174.

BEINHAUER, W. 2006. A Widget Library for Gaze-based Interaction Elements, in *ETRA: Eye Tracking Research and Applications Symposium*. San Diego, California, USA: ACM Press. p. 53-53.

KUMAR, M. 2007. GUIDe Saccade Detection and Smoothing Algorithm. Technical Report CSTR 2007-03, Stanford University

KUMAR, M., GARFINKEL, T., BONEH, D., AND WINOGRAD, T. 2007a. Reducing Shoulder-surfing by Using Gaze-based Password Entry. Technical Report CSTR 2007-05, Stanford University

KUMAR, M., PAEPCKE, A., AND WINOGRAD, T. 2007b. EyePoint: Practical Pointing and Selection Using Gaze and Keyboard, in *CHI*. San Jose, California, USA: ACM Press.

KUMAR, M., WINOGRAD, T., AND PAEPCKE, A. 2007c. Gaze-enhanced Scrolling Techniques, in *CHI*. San Jose, California, USA: ACM Press.

MAJARANTA, P., AULA, A., AND RÄIHÄ, K.-J. 2004. Effects of Feedback on Eye Typing with a Short Dwell Time, in *ETRA: Eye Tracking Research & Applications Symposium*. San Antonio, Texas, USA: ACM Press. p. 139-146.

MAJARANTA, P., MACKENZIE, I.S., AULA, A., AND RÄIHÄ, K.-J. 2003. Auditory and Visual Feedback During Eye Typing, in *CHI*. Ft. Lauderdale, Florida, USA: ACM Press. p. 766-767.

MONTY, R.A., SENDERS, J.W., AND FISHER, D.F. *Eye Movements and the Higher Psychological Functions*. 1978, Hillsdale, New Jersey, USA: Erlbaum.

SALVUCCI, D.D. 1999. Inferring Intent in Eye-Based Interfaces: Tracing Eye Movements with Process Models, in *CHI*. Pittsburgh, Pennsylvania, USA: ACM Press. p. 254-261.

SALVUCCI, D.D. AND GOLDBERG, J.H. 2000. Identifying Fixations and Saccades in Eye-Tracking Protocols, in *ETRA: Eye Tracking Research & Applications Symposium*. Palm Bech Gardens, Florida, USA: ACM Press. p. 71-78.

TOBII TECHNOLOGY, AB. 2006 Tobii 1750 Eye Tracker. http://www.tobii.com.

YARBUS, A.L. *Eye Movements and Vision*. 1967, New York: Plenum Press.

ZHAI, S. 2003. What's in the Eyes for Attentive Input, in *Communications of the ACM*, Vol. 46, (3)