Open Framework for Error-Compensated Gaze Data Collection with Eye Tracking Glasses

Kari Siivonen, Joose Sainio, Marko Viitanen, Jarno Vanne, Timo D. Hämäläinen
Laboratory of Pervasive Computing
Tampere University of Technology, Finland
{kari.siivonen, joose.sainio, marko.viitanen, jarno.vanne, timo.d.hamalainen}@tut.fi

***Abstract*— Eye tracking is nowadays the primary method for collecting training data for neural networks in the Human Visual System modelling. Our recommendation is to collect eye tracking data from videos with eye tracking glasses that are more affordable and applicable to diverse test conditions than conventionally used screen based eye trackers. Eye tracking glasses are prone to moving during the gaze data collection but our experiments show that the observed displacement error accumulates fairly linearly and can be compensated automatically by the proposed framework. This paper describes how our framework can be used in practice with videos up to 4K resolution. The proposed framework and the data collected during our sample experiment are made publicly available.**

Keywords—eye tracking, open-source, open data set, error correction

I. INTRODUCTION

Eye tracking is the process of measuring the gaze point of a person. The gathered data can serve as an input for studies on the *Human Visual System* (HVS) that is of particular interest for computer vision and video coding. Computer vision seeks to mimic the behavior of HVS where eye tracking can be used to collect ground truth data and verify the computational HVS models [1]. In video coding, the models [2] or original gaze data can be used to extract the region of interest from the scene in order to improve subjective quality or coding efficiency. Additionally, eye tracking data can act as a basis for objective video quality metrics [3], [4].

Many of the existing saliency models make use of neural networks [5], [6] but they are still lacking, especially with videos, since unlike static images, the viewer cannot scan through the whole video frame but only some salient details in it. Videos also introduce temporal continuity to the observation making the models, which are aimed at static images, insufficient. According to [3], observations made by around twenty persons are reliable enough to construct a generic heat map for a video sequence. Coupling this with the fact that neural networks require massive amounts of data to produce accurate models, there is an apparent need for a straightforward methodology to collect gaze data from videos.

Currently, there exists three distinct classes of eye tracking equipment used commercially: screen based solutions, eye tracking glasses, and accessories integrated into *Head Mounted Displays* (HMDs). Screen based solutions can be either integrated into a single screen or a distinct sensor used with various screens. The devices integrated into a screen offer the best tracking quality but are often too expensive. In addition, they provide a level of detail not needed in applications that combine data from multiple observers. Distinct sensors also offer good quality but they are often intended for at most 24 inch screens [7], which is not adequate for the latest video formats such as Ultra HD that typically feature screen sizes above 24 inches.

The advantage of eye tracking glasses is that they can be used in a large variety of scenarios, e.g., with any screen size. However, since the glasses are attached physically to the user, they may cause a slight discomfort and are prone to moving. The displacement error of moving glasses can be corrected by recalibrating the eye tracker periodically. However, manually verifying the calibration multiple times can be cumbersome and time consuming. Additionally, if the tracker goes out of calibration, all data since the latest calibration has to be discarded as it is impossible to detect the point of invalidation. For these reasons, all known eye tracking studies gather gaze data with screen based eye trackers [4], [8].

The proposed framework uses offline drift correction to improve the quality of the data collected with eye tracking glasses. Self-calibration methods based on saliency models have been proposed in other similar works [9] but they require a robust saliency model. Our proposal can automatically compensate the displacement error of the glasses, which is found to accumulate fairly linearly. In addition, this work describes how the eye tracking experiment can be conducted in practice. The proposed framework and the eye tracking data gathered in our sample experiment are available for download [10]. The framework is built using Python and the OpenCV, numpy, and scipy libraries.

Currently, a few companies offer eye tracking glasses, most notably Tobii [11] and Pupil Labs [12]. Tobii is probably the best known supplier, but Pupil Mobile Eye Tracking Headset was chosen for this experiment due to its competitive price and open-source software stack. In principle, the framework could also process gaze data collected by HMDs but those experiments are out of the scope of this paper.

The rest of this paper is structured as follows. Chapter II gives an overview of the gaze data collection conducted in practice. Chapter III explains our methodology for fixing the systematic error that accumulated during the experiment due to the moving glasses. Chapter IV concludes the paper.

II. EYE TRACKING EXPERIMENT

Pupil Mobile Eye Tracking Headset [12] used in our experiments was equipped with a world camera for capturing the view from the subject's perspective and two eye cameras for recording pupil positions. The world camera supports several capture modes with fixed sampling frequency and resolution. In our tests, the world camera operated at 720p60 and the eye cameras at 240p120. The manufacturer promises a 0.60° accuracy and 0.08° precision. The used monitor was a 27-inch Lenovo ThinkVision X1 [13] 4K screen with 60 Hz refresh rate. A snapshot of the test setup is depicted in Fig. 1.

This work was supported in part by Business Finland and the Academy of Finland (decision no. 301820).

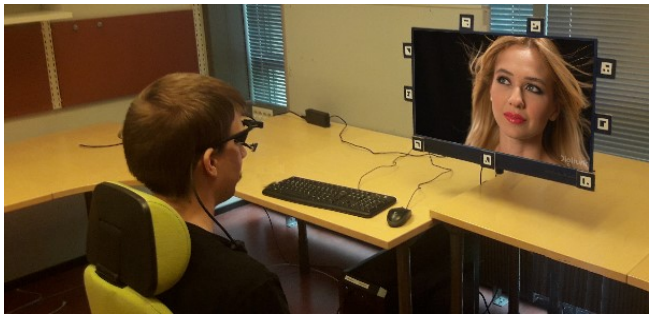


Fig 1. The test environment.

A. Test Group

The test group consisted of 37 people out of which 17 were female and twenty male. The minimum, maximum, mean, and median ages were 13, 43, 27.4, and 26, respectively. Most of the participants were of Finnish background, either students or staff of Tampere University of Technology.

B. Test Material

Table I tabulates characteristics of our test set. It includes all 8-bit HEVC common test sequences [14], seven videos from Ultra Video Group [15], one from AWS Elemental [16], and twelve from Xiph.org [17]. The sequences were stretched to the screen while maintaining the original aspect ratio. A black bar was added on both sides for videos narrower than 16:9. The refresh rate of videos from [15] was downsampled from 120 Hz to 60 Hz by removing every other frame because no 120 Hz 4K display was available at the time.

In [8], eye tracking data for the HEVC common test sequences is provided but only fixations are included instead of raw gaze data for each frame. In [18], eye tracking data for sequences in [15] were collected but they were played at nearly fifth of the original speed.

C. Test Environment

The experiments were conducted in a well lit room with a participant sitting about one meter away from the monitor. The participants were allowed to freely move in the seat to stay comfortable. The screen position was adjusted so that the viewer's gaze was in the center of the screen when looking straight on. The viewing conditions were prepared based on the recommendations of ITU-T P.910 [19] despite lighting conditions, which are movie-theater-like.

D. Test Preparation

Each test session began with calibrating the eye tracker. The calibration results were verified by the test personnel. In most cases, only a single calibration attempt was necessary but in the worst (single) case three attempts were needed. Viewers were instructed to perform a free-viewing task, i.e., only watching the videos without any specific goal. The experiment itself consisted of the subject watching a sequence of videos with a calibration check after every five videos.

E. Test Execution

The videos were displayed using *Media Player Classics – Home Cinema* (MPC-HC) [20]. The Pupil Capture software uses *ZeroMQ* (ZMQ) sockets for internal and external communication, and the recording can be started and stopped through the socket interface. MPC-HC was modified to facilitate ZMQ. It starts the recording slightly before the video is played and stops after the video has finished playing.

TABLE I. TEST VIDEO SEQUENCES

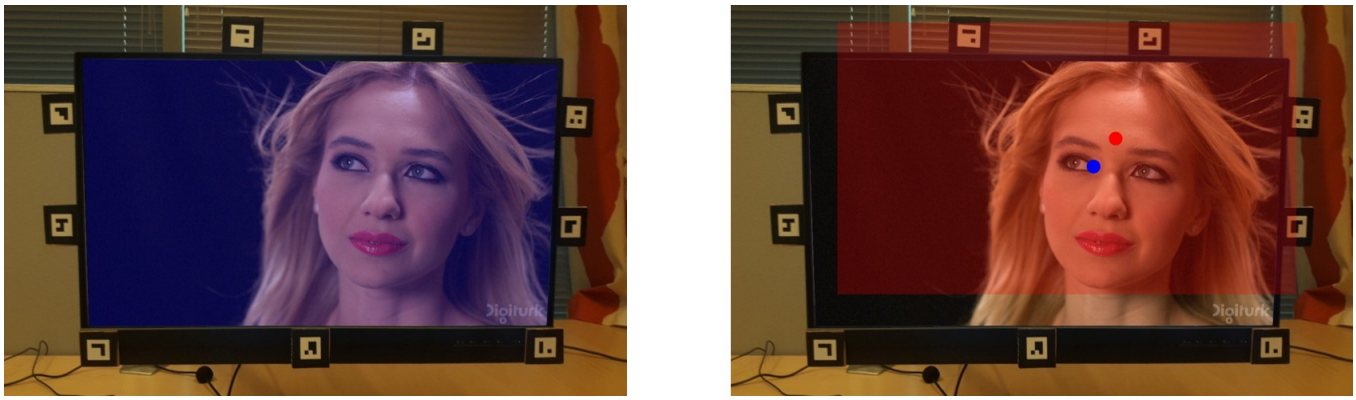
Sequence	Resolution	FPS	Duration (s)
BaskeballPass [14]	416×240	50	10
BasketballDrill [14]	832×480	50	10
BasketballDrillText [14]	832×480	50	10
BasketballDrive [14]	1920×1080	50	10
Beauty [15]	3840×2160	60	5
BlowingBubbles [14]	416×240	50	10
Bosphorus [15]	3840×2160	60	5
BQMall [14]	832×480	60	10
BQSquare [14]	416×240	60	10
BQTerrace [14]	1920×1080	60	10
Cactus [14]	1920×1080	50	10
ChinaSpeed [14]	1024×768	30	16.7
crowdRun [17]	1920×1080	50	10
Foreman 4k [16]	3840×2160	24	10.4
FourPeople [14]	1280×720	60	10
HoneyBee [15]	3840×2160	60	5
Jockey [15]	3840×2160	60	5
Johnny [14]	1280×720	60	10
Kimono1 [14]	1920×1080	24	10
KristenAndSara [14]	1280×720	60	10
oldTownCross [17]	1920×1080	50	10
parkrun [17]	1280×720	50	10
ParkScene [14]	1920×1080	24	10
PartyScene [14]	832×480	50	10
pedestrianArea [17]	1920×1080	25	15
PeopleOnStreet [14]	2560×1600	30	5
RaceHorses [14]	832×480	30	10
ReadySteadyGo [15]	3840×2160	60	5
rushHour [17]	1920×1080	25	20
ShakeNDry [15]	3840×2160	60	2.5
shields [17]	1280×720	50	10
SlideEditing [14]	1280×720	30	10
SlideShow [14]	1280×720	20	25
speedBag [17]	1920×1080	30	19
station2 [17]	1920×1080	25	12.5
stockholm [17]	1280×720	60	10
Traffic [14]	2560×1600	30	5
vidyo1 [17]	1280×720	60	10
vidyo3 [17]	1280×720	60	10
vidyo4 [17]	1280×720	60	10
YachtRide [15]	3840×2160	60	5

The actual start time of the video can be obtained by comparing the brightness of the screen area on the recording. The screen is completely black before the video starts but not during video playback. This method introduces, on average, an 8 ms error to the positions of the gaze points on the video timeline. An alternative solution would have been to start the recording exactly when the video is supposed to appear on the screen. However, the used monitor has a reported latency of 6 ms [13] and it also depends on the content making the error more unpredictable than in the former method.

From 37 participants, five had to be discarded due to erroneous data, leaving 32 valid. In four cases, makeup interfered with pupil detection and in one case mistake was made when setting up the eye tracker.

F. Test Feedback

After the experiment, the participants were verbally questioned on whether they had any comments on the experiment. The majority of the feedback was positive but some reported minor exhaustion because the test videos were so short. The selected videos are widely adopted in the field so their length was not considered an issue. Notably none of the participants found the glasses uncomfortable, so they were



a) b)

Fig 2. Gaze point mapping. (a) Physical screen surface for gaze points (in blue tint). (b) The drifted surface due to the moving glasses (in red tint). The red dot visualises where the gaze point will be mapped on the actual screen and the blue one where it should be mapped.

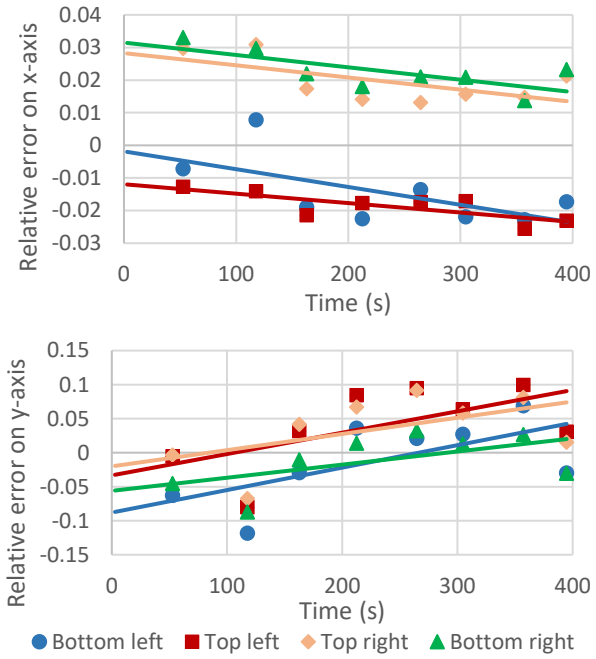


Fig 3. Error propagation at the corner calibration points in an example test.

unlikely to pay attention to the glasses. Therefore, it is improbable that wearing the glasses affected the gaze points.

III. Methodology for Fixing Systematic Error

During the experiment, gaze data was collected at a sample rate of 120 Hz while the world view was recorded at 60 Hz. The Pupil Labs software calculates the screen surface from the world view. The position and orientation of the surface is based on the tags attached to the monitor.

A. Error Detection

Fig. 2(a) depicts the physical screen with the highlighted screen surface. The gaze data is exported to csv format, during which the measured gaze points are projected onto the surface. The gaze data contains the world timestamp, a frame index, a gaze timestamp with normalized coordinates, and coordinates scaled to surface resolution. The data also indicates whether the gaze point is on screen and gives a confidence value for the detection. Unobscured, clearly visible pupils usually result in a perfect confidence value of 1.0.

The calibration was checked eight times per experiment. A single calibration check consists of a sequence of five calibration symbols appearing in the constant order: center, bottom left, top left, top right, and bottom right. These checks were used to extract error data for the correction process. The error was measured relative to screen dimensions. The bottom left and top right of the screen were designated to (0, 0) and (1, 1), respectively. The recorded gaze points were not limited to these boundaries, as it is possible to look outside of the screen boundaries. Each gaze point was compared against its corresponding calibration point on screen and the difference was recorded separately for *x*- and *y*-axis.

A common visual observation among all test cases is that the displacement error propagates fairly linearly. The linear nature of the error is illustrated in Fig. 3 with an example test case. In addition, the magnitude of the error is relative to position on screen. Gaze points on the left side of the screen have lower error values on the *x*-axis than points on the right side. This is also true for *y*-axis, where points at the bottom half of the screen have lower error values than the points at the top half. The average error scaled to 4K resolution was 121 and 153 pixels for *x*- and *y*-axis, respectively.

B. Error Correction

The offline correction is done by analyzing the gaze data from the calibration checks and calculating a correction factor based on the results of the analysis. The process consists of the following steps:

1. Prune erroneous gaze points;
2. Cluster the remaining gaze points;
3. Create linear error timeline based on the clusters;
4. Calculate a correction factor and apply to video gaze data.

In the first phase, erroneous gaze points, such as those when the participant has blinked, are pruned from the point cloud. The software provided by Pupil Labs uses confidence values to clean most of the erroneous points but still some of them persist. For example, during blinking the software might falsely detect the pupil from the eye lashes for a couple of frames. These erroneous points are usually isolated from normal points and regions in the data timeline containing these points can be easily detected and removed.

After removal of erroneous regions, some outlying points may still remain in the data. Here, the outlying points are a result of saccades between fixation points or random errors in pupil tracking. These points are usually isolated from the rest

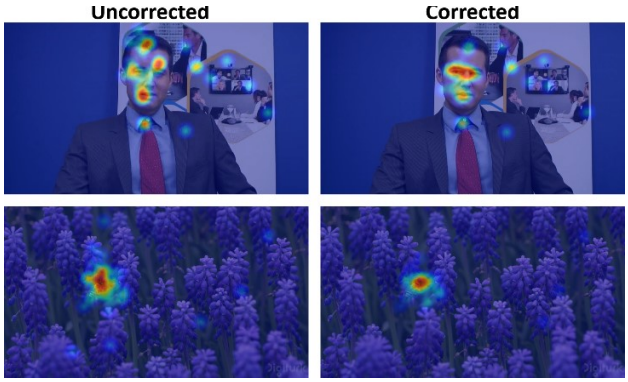


Fig 4. Comparison of uncorrected and corrected heatmaps. The pictures are from sequences Johnny frame 300 (top) and Honeybee frame 240 (bottom).

and can be detected by calculating the local outlier factors [21] for them. The method calculates the local density of each point as a function of distance to its *k*-nearest neighbors, where $k = 10$. The points with substantially lower local density over their neighbors will be removed as outliers.

The next step is to assign all valid gaze points into clusters, which represent fixations in gaze data. At this point, the optimal number of clusters is unknown. If the data is already packed in a single dense cluster, no further clustering is done. Otherwise, the optimal number is determined by silhouette analysis [22] using *k-means* as the clustering algorithm. For each point, the method calculates a similarity score over the other points in the same cluster and difference score over the points in other clusters. The similarity and difference scores are measured as Euclidean distances. The number of clusters with the best average score among all points is selected after which the clusters are formed. The largest one is selected for processing and the rest will be discarded.

The spatially clustered data is ordered based on the calibration point it belongs to. An average point for both x- and y-axes is calculated from each cluster and the points are arranged into a timeline based on which calibration they correspond to. This results in a graph which shows the evolution of error over time for each separate calibration point (Fig. 3). A linear model is fitted to each calibration point, allowing the approximation of the gaze error at any given point of time. The approximation can be used to form the correction factor.

Due to the glasses moving, the surface where the gaze points are projected is not stationary, as depicted in Fig. 2. The surface can be transformed back where it is supposed to be by calculating a perpective transformation matrix. It is derived from the points of the original and shifted surface. The original surface is defined by the corner calibration points whereas the shifted surface is created by shifting the original points based on the linear equation. The perspective transform matrix is calculated for each test video separately. The correction is done by applying a perspective transform with the supplied transform matrix on the collected gaze data.

IV. Conclusions

This paper proposed an open framework for gathering gaze data with eye tracking glasses and compensating the displacement error of the moving glasses automatically. The effect of correction can be seen in Fig. 4. From the 32 valid participants, 27 received noticeable benefits from the correction. The median of correction was 75 % and 78 % for *x*- and *y*-axis, respectively, where the percentages are the ratio of the correction to the uncorrected error. Among the eight subjects who had the largest error, the error was corrected by 52 and 581 pixels on *x*- and *y*-axis, respectively. The raw eye tracking data and the corrected data along with the Python code is public [10].

Acknowledgment

The authors would like to thank all volunteers for taking part in our eye tracking experiment.

V. References

[1] L. Itti, "Automatic foveation for video compression using a neurobiological model of visual attention," *IEEE Trans. Image Process.*, vol. 13, no. 10, pp. 1304-1318, Oct. 2004.

[2] C. Guo and L. Zhang, "A novel multiresolution spatiotemporal saliency detection model and its applications in image and video compression," *IEEE Trans. Image Process.*, vol. 19, no. 1, pp. 185-198, Jan. 2010.

[3] Y. Gitman, M. Erofeev, D. Vatolin, B. Andrey, and F. Alexey, "Semiautomatic visual-attention modeling and its application to video compression," *in Proc. IEEE Int. Conf. on Image Process*., Paris, France, Oct. 2014.

[4] Z. Li, S. Qin, and L. Itti, "Visual attention guided bit allocation in video compression," *Image and Vision Computing*, vol. 29, no. 1, pp. 1-14, Jan. 2011.

[5] S. S. S. Kruthiventi, K. Ayush, and R. V. Babu, "DeepFix: A fully convolutional neural network for predicting human eye fixations," *IEEE Trans. Image Process.,* vol. 26, no. 9, pp. 4446-4456, Sept. 2017.

[6] X. Li, L. Zhao, L. Wei, M. H. Yang, F. Wu, Y. Zhuang, H. Ling, and J. Wang, "DeepSaliency: Multi-task deep neural network model for salient object detection," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3919-3930, Aug. 2016.

[7] *Tobii Pro x3-120* [Online]. Available: https://www.tobiipro.com/product-listing/tobii-pro-x3-120/

[8] M. Xu, L. Jiang, X. Sun, Z. Ye and Z. Wang, "Learning to detect video saliency with HEVC features," *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 369-385, Jan. 2017.

[9] Y. Sugano and A. Bulling, "Self-calibrating head-mounted eye trackers using egocentric visual saliency," *in Proc. ACM Symp. User Interface Software and Technol.*, Charlotte, North Carolina, USA, Nov 2015.

[10] *Eye-tracking framework* [Online]. Available: https://github.com/ultravideo/eye-tracking-framework

[11] *Tobii Pro Glasses 2* [Online]. Available: https://www.tobiipro.com/product-listing/tobii-pro-glasses-2/

[12] *Pupil Labs* [Online]. Available: https://pupil-labs.com/pupil/

[13] *ThinkVision X1 Datasheet* [Online]. Available: https://support.lenovo.com/fi/en/solutions/pd104221

[14] F. Bossen, "Common HM test conditions and software reference configurations," JCTVC-L1100, *12th JCT-VC meeting, Geneva,* Jan. 2013.

[15] *Ultra Video Group* [Online]. Available: http://ultravideo.cs.tut.fi/

[16] *AWS Elemental* [Online]. Available: https://www.elemental.com/resources/4k-test-sequences

[17] *Xiph.org, Derf's Collection* [Online]. Available: https://media.xiph.org/video/derf/

[18] T. Vigier, J. Rousseau, M. Perreira Da Silva, and P. Le Callet, "A new HD and UHD video eye tracking dataset," *in Proc. ACM Multimedia Syst. Conf.*, Klagenfurt, Austria, May 2016.

[19] ITU: "Subjective video quality assessment methods for multimedia applications," ITU-T Recommendation P.910, Geneva, Switzerland, Apr. 2008.

[20] *Media Player Classic – Home Cinema* [Online]. Available: https://mpc-hc.org/

[21] M. Breunig, H. Kriegel, R. Ng, and J. Sander, "Lof: identifying density-based local outliers," *in Proc. Int. Conf.* on Management of Data, Dallas, May 2000.

[22] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53-65, Nov. 1987.