GazeCode: open-source software for manual mapping of mobile eye-tracking data

Jeroen S. Benjamins
Experimental Psychology, Helmholtz Institute, and Social, Health and Organizational Psychology, Utrecht University
Utrecht, The Netherlands
j.s.benjamins@uu.nl

Roy S. Hessels
Experimental Psychology, Helmholtz Institute, and Developmental Psychology, Utrecht University
Utrecht, The Netherlands

Ignace T.C. Hooge
Experimental Psychology, Helmholtz Institute, and Developmental Psychology, Utrecht University
Utrecht, The Netherlands

ABSTRACT

Purpose: Eye movements recorded with mobile eye trackers generally have to be mapped to the visual stimulus manually. Manufacturer software usually has sub-optimal user interfaces. Here, we compare our in-house developed open-source alternative to the manufacturer software, called GazeCode. **Method**: 330 seconds of eye movements were recorded with the Tobii Pro Glasses 2. Eight coders subsequently categorized fixations using both Tobii Pro Lab and GazeCode. **Results**: Average manual mapping speed was more than two times faster when using GazeCode (0.649 events/s) compared with Tobii Pro Lab (0.292 events/s). Inter-rater reliability (Cohen's Kappa) was similar and satisfactory; 0.886 for Tobii Pro Lab and 0.871 for GazeCode. **Conclusion**: GazeCode is a faster alternative to Tobii Pro Lab for mapping eye movements to the visual stimulus. Moreover, it accepts eye-tracking data from manufacturers SMI, Positive Science, Tobii, and Pupil Labs.

CCS CONCEPTS

• **Applied computing → Psychology**;

KEYWORDS

Mobile eye-tracking, manual classification, eyemovements, eye-tracking events

ACM Reference Format:
Jeroen S. Benjamins, Roy S. Hessels, and Ignace T.C. Hooge. 2018. GazeCode: open-source software for manual mapping of mobile eye-tracking data. In *ETRA '18: 2018 Symposium on Eye Tracking Research and Applications, June 14–17, 2018, Warsaw, Poland.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3204493.3204568

1 INTRODUCTION

The use of, and commercial availability of, mobile eye-trackers in research has grown in recent years. Mobile eye-trackers typically consist of a head-mounted system that contains two cameras; one camera is used to film the eye (eye-camera) while a second camera (scene-camera) films the world. As such, these devices allow for recording of a person's point of regard (POR) in daily life activities [Land and Hayhoe 2001; Land et al. 1999], shopping behavior in supermarkets [Brône et al. 2011], navigation using maps [Kiefer et al. 2014], viewing behavior of infants while crawling [Franchak et al. 2011] and in clinical and medical settings [Dik et al. 2016; Marx et al. 2012].

In most remote and tower eye-tracker measurements the POR is expressed in coordinates of the screen on which a visual stimulus is presented. This allows researchers to couple the POR to known object coordinates on the screen. In mobile eye-tracking measurements, mapping of the POR to the visual stimulus is more difficult [Brône et al. 2011]: The video of the scene-camera is dynamic, as it moves along with the head movements of the person wearing it. World-fixed objects change position in the scene camera image when the head rotates. As long as the relation between the head and the world is unknown, mapping gaze from a head mounted eye tracker is a difficult process that is often carried out manually.

One of the first examples of mapping the POR to meaningful locations in the world was provided by Land et al. [1999] with a recording of making a cup of tea. The resulting 4-minute video of the scene camera with the POR overlaid, was analyzed frame by frame. This required manual classification of about 36000 frames of video material, which is very time-consuming. Instead of analyzing mobile eye-tracking recordings frame by frame, this analysis can be sped up finding periods in the recording when gaze is fixed to world. These periods may occur in the eye-tracking data when the observer looks at an object and: (a) both observer and the object do not move with respect to each other, (b) the observer moves while the object does not, (c) the object moves, while the observer does not, or (d) observer and object both move. In all these examples, the resulting eye-tracker signal is a slow (or no) position change in coordinates of the scene-camera image. This is opposed to a fast position change due to a saccadic eye movement. We refer to these slow position changes in the eye-tracking data as "slow phases" in the remainder of this paper. To be able to classify these slow phases, different algorithms can be used [Hooge and Camps 2013; Salvucci and Goldberg 2000]. After classification these slow phases instead of individual POR video frames can be mapped to locations in the world.

For the remainder of this paper we will refer to manual mapping, by which we mean the mapping of the slow phases unto the visual stimulus by a manual coder assigning to it a pre-defined category

ETRA '18, June 14–17, 2018, Warsaw, Poland
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5706-7/18/06.
https://doi.org/10.1145/3204493.3204568

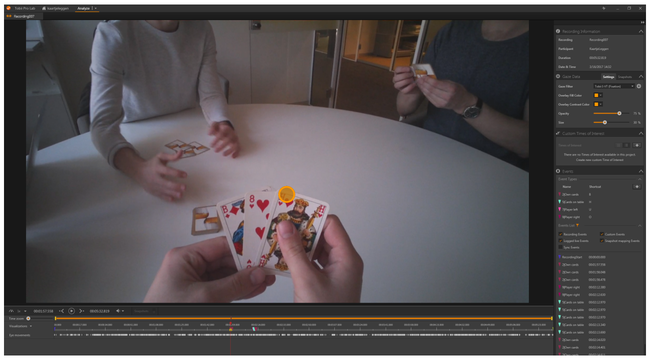

Figure 1: Tobii Pro Lab interface showing the *Analyse* mode with Gaze Data, and Event modules on the rights side of the interface.

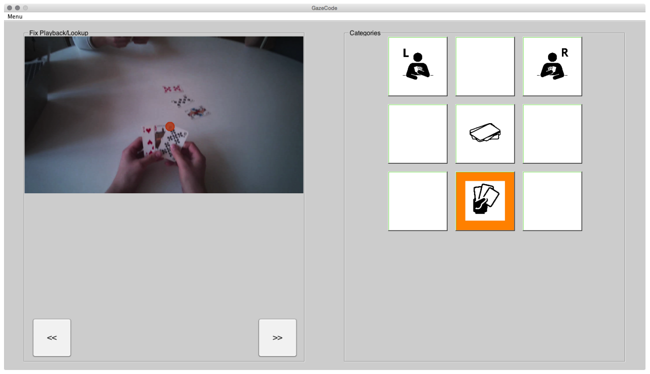

Figure 2: GazeCode interface with scene-video and buttons to browse slow phases in the left panel and coding buttons in the right panel. Buttons are also operable with keys.

(e.g. a person, a ball, a cup of tea, etc.). Some researchers and companies have been able to automate or semi-automate this mapping. Here we first describe the limitations of (semi-) automatic methods of mapping slow phases unto the visual stimulus. We then discuss the currently available software for manual mapping, how it can be improved, and present and compare our own software, GazeCode, as an alternative.

1.1 Limitations of software for (semi)-automatic mapping of slow phases to the scene video

As mentioned above, mapping of slow phases unto the visual stimulus can be (semi-) automated with different techniques. Object recognition algorithms [Felzenszwalb et al. 2010; Lowe 2004] can automatically map slow phases to objects in the scene video. Adding physical markers in the world [Brône et al. 2011; Munn and Pelz 2008; Munn et al. 2008] (e.g. infrared or fiduciary markers such as used with Tobii Glasses Generation 1 or Pupil-Labs respectively) can be used to automatically define areas of interest (AOIs) and determine whether slow phases fall in these AOIs. In a semi-automatic method, AOIs are manually defined in relevant frames of the scene video, automatically interpolated for the whole scene-video and then processed similarly as in method with markers.

Although promising developments have been made in (semi-)automatic mapping methods, these methods are not without flaws [Brône et al. 2011]. Most notably, these methods only work well for manual mapping of objects that are not displaced during recordings (e.g. a stationary product on a shelf or a poster on a wall). As a result, most data acquired in mobile eye- tracking experiments are still manually mapped to the world.

1.2 Current software for manual mapping of slow phases

Currently available software packages for the purpose of manual mapping seem suboptimal because of the user interface. Good interfaces should take into account the cognitive and perceptual limitations of human coders. Nielsen [1994], for example, suggests a couple of principles that one could consider when developing an interface. As an example, we have inspected Tobii Pro Lab, which was readily available to the authors, on these principles. This software has an Analyse mode that allows browsing through fixations and adding custom codes to these events (see Figure 1). In this Analyse mode we note that there is: (1) poor visibility of system status (i.e. interface with white letters on black background, random color-coding of event codes, small list of coded events), (2) no match between system and world (limited amount of keyboard buttons for event coding available), (3) poor error prevention (e.g. double event codes are possible) and poor correction methods, and (4) no minimalist design principle (a lot of irrelevant information). Moreover, manufacturer software suites are typically only able to handle data from one type of mobile-eye tracking device, namely that of the manufacturer itself.

In this paper we compare manual mapping between Tobii Pro Lab and GazeCode, our open-source software. We will use one mobile eye-tracking recording made with Tobii Pro Glasses 2. We will compare agreement between human codings for both methods. We expect GazeCode to be more efficient than Tobii Pro Lab as should be reflected in a faster coding speed using GazeCode, without being less effective as indicated by inter-rater reliability measures.

2 GAZECODE

GazeCode has been developed as a software suite that can be used in Matlab®. We followed the principles of good interface design (cf. [Nielsen 1994]) when we built the graphical interface. Unlike Tobii Pro Lab, GazeCode was developed to only do one task, namely manual mapping.

In GazeCode there is: (1) clear visibility of system status (see Figure 2; bright orange border around the button for a coded event), (2) a clear match between system and world (buttons are operable by keyboard keys that match the layout of the interface, e.g. numpad keys that matches the 3 by 3 grid of coding buttons), (3) good error prevention and good error correction (events can not be double-coded and codes can be easily reset), and (4) a minimalistic interface design (no superfluous information). Moreover, GazeCode can handle data from both Tobii Pro Glasses 2 and the Pupil-Labs eye- tracker. A beta version is in development that handles data from SMI and Positive Science eye-trackers as well.

GazeCode has been tested and operates on both Windows and Apple operating systems (with Matlab 2014a on Mac OSX 10.10.5 and Matlab 2013a on Windows 7 Enterprise, Service Pack 1). GazeCode is freely available online through software repository site GitHub (https://github.com/jsbenjamins/gazecode).

3 METHODS

3.1 Stimulus

To compare manual mapping in both GazeCode and Tobii Pro Lab, a short measurement of 330 seconds was made. Three volunteering students from Utrecht University played the gambling card game Thirty-One. One of the players was equipped with Tobii Pro Glasses 2 sampling at 50 Hz. The scene video of this recording thus had a perspective of the game in which four clear categories of mapping of slow phases are present. The following categories were defined for manual mapping: (1) The cards that were in the player’s own hand (cards in own hand), (2) any of the cards on the table (cards on table), (3), the player to the left of the player (player left), and (4) the player to the right of the player (player right).

3.2 Eye-movement event classifiers and number of events

In both software packages, a slow phase classifier is applied to the eye-tracking data. For this classification of events, GazeCode and Tobii Pro Lab use different algorithms are used in. Tobii Pro lab uses an implementation of the I-VT classifier as described by Salvucci and Goldberg [2000] and Komogortsev et al. [2010]. The eye-movement slow phase classifier used in GazeCode is the same as in Hooge and Camps [2013], which is currently implemented as the standard in GazeCode as this has proven to be a good algorithm for data acquired with eye-tracker that sample at lower frequencies (i.e. 50 Hz with Tobii Pro Glasses 2). However, the algorithm can be replaced to fit a researcher’s choice. Using these eye-movement event classifiers resulted in 792 events to be mapped by the human coders when using Tobii Pro Lab and 560 events when using GazeCode.

3.3 Task

We had human coders manually assign slow phases to the aforementioned categories. The recording was prepared for this task in both Tobii Pro Lab and GazeCode in such a way that a human coder could start mapping the first slow phase immediately after having started a stopwatch. After mapping all slow phases, the human coder was instructed to stop the stopwatch. Human coders were asked to write down their coding time rounded to periods of 30 seconds.

3.4 Participants

Eight human coders participated in this software comparison. Six coders were naïve to the purpose of the comparison. Two coders were authors. Coders differed in their experience in analyzing (mobile) eye-tracking data from no previous experience to multiple years of experience (maximum 4 years). Half of the coders were female and average age was 27.9 years (age range 23 to 37). Manually mapping slow phases in the mobile eye-tracking recording both in Tobii Pro Lab and GazeCode was counterbalanced, such that half of the coders first mapped the recording in Tobii Pro Lab and then in GazeCode and vice versa.

3.5 Analysis

To compare manual mapping of events we compared the set of 560 category codes when using GazeCode and the 792 category codes when using Tobii Pro Lab between all combinations of human coders by using Cohen’s Kappa [Cohen 1960]. This agreement measure can be calculated for individual human coder pairs and can be averaged across all human coders. When Cohen’s Kappa reaches a value of 0.81 or higher for each method, it can be considered to reflect almost perfect agreement [Landis and Koch 1977] and would allow us to claim that both methods are equally effective.

The efficiency of the software will be determined by the time it takes human coders for manual mapping. We examine both the coding time for manual mapping, and the number of events coded per second. The second measure is a better estimator for coding speed. Lastly, we will report qualitative observations from the human coders, which we compare with our own observations as mentioned in section 1.2.

4 RESULTS AND DISCUSSION

4.1 Human coder agreement (effectivity)

Manual mapping agreement, expressed in Cohen’s Kappa, was almost perfect [Landis and Koch 1977] and very similar for both GazeCode and Tobii Pro Lab: For GazeCode, the agreement was on average almost perfect at 0.871 (minimum: 0.80, maximum: 0.96, standard deviation: 0.042). For Tobii Pro Lab this is similar with an average of 0.886 (minimum: 0.82, maximum: 0.97, standard deviation: 0.040). As two of the coders are authors, one may assert that the agreement measures are biased. However, computing Cohen’s Kappa without the authors did not yield substantially different values (0.862 for GazeCode vs. 0.874 for Tobii Pro Lab).

4.2 Manual mapping time and speed (efficiency)

Interestingly, the time needed for the human coders for manual mapping differs dramatically: The average time for manual mapping in GazeCode is 69% shorter compared to the time for manual mapping in Tobii Pro Lab. Human coders took 879 seconds on average (SD: 124 seconds) using GazeCode whereas they took 2880 seconds on average (SD: 674 seconds) using Tobii Pro Lab.

It may be that manual mapping time was longer in Tobii Pro Lab than in GazeCode, as more events were classified in the eye-tracking data in Tobii Pro Lab. To determine the manual mapping speed, we computed the number of events coded per second. When considering this measure, manual mapping in GazeCode was faster than manual mapping in Tobii Pro Lab. The average speed of coding using GazeCode is 0.649 events per second (SD: 0.088) whereas this speed is on average 0.292 (SD: 0.075) events per second when using

Tobii Pro Lab. Again, computing these measures without the non-naïve authors did not yield substantially different values: average time to code without authors is 905 seconds using GazeCode, and 3105 seconds using Tobii Pro Lab. Average manual mapping speed is 0.629 events per second in GazeCode, and 0.265 events per second in Tobii Pro Lab.

4.3 Qualitative observations made by the human coders

Coders reported several problems with feedback and error-proneness of their manual mapping in Tobii Pro Lab. All coders noted a decrease in operating speed of Tobii Pro Lab as they progressed. These observations seem to suggest that using the Tobii Pro Lab interface compared to using GazeCode's interface is suboptimal.

5 CONCLUSIONS

In this paper we set out to compare in-house developed software, GazeCode, with Tobii Pro Lab for use in manual mapping of slow phases to the scene-video of a mobile eye-tracking measurement. To this end, we had human coders manually map slow phases of a mobile eye-tracking recording acquired with Tobii Pro Glasses. Agreement between human coders was high and similar for both methods with a Cohen's Kappa of around 0.88. Moreover, manual mapping speed was much higher using GazeCode, which we attribute to the optimization of the interface for this single task. GazeCode is open-source, allowing for implementation of any eye-movement classification algorithm, a variable number of coding categories and variable interface layout options. It currently handles eye-tracking data from four models of mobile eye trackers. To conclude, we have created a fast, free and versatile method for mobile eye-tracking analysis that outperforms manufacturer software.

ACKNOWLEDGMENTS

The authors would like to thank Heather Kirkorian, Sharisse van Driel & Halszka Jarodzka for providing us with a sample measurement recorded with Positive Science and SMI mobile eye-trackers respectively to be able to make GazeCode compatible with recordings from these devices as well. We also like to thank Gijs Holleman, Nikki Timmers, and Joris Klunder for volunteering to participate in the mobile eye-tracking measurement.

REFERENCES

Geert Brône, Bert Oben, and Toon Goedemé. 2011. Towards a more effective method for analyzing mobile eye-tracking data: integrating gaze data with object recognition algorithms. In *Proceedings of the 1st international workshop on pervasive eye tracking & mobile eye-based interaction*. ACM, New York, NY, USA, 53–56. https://doi.org/10.1145/2029956.2029971

Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement* 20, 1 (1960), 37–46. https://doi.org/10.1177/001316446002000104

Vincent K Dik, Ignace TC Hooge, Martijn G van Oijen, and Peter D Siersema. 2016. Measuring gaze patterns during colonoscopy: a useful tool to evaluate colon inspection? *European Journal of Gastroenterology & Hepatology* 28, 12 (2016), 1400–1406. https://doi.org/10.1097/MEG.0000000000000717

Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. 2010. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence* 32, 9 (2010), 1627–45. https://doi.org/10.1109/TPAMI.2009.167

John M Franchak, Kari S Kretch, Kacey C Soska, and Karen E Adolph. 2011. Head-mounted eye tracking: a new method to describe infant looking. *Child development* 82, 6 (2011), 1738–50. https://doi.org/10.1111/j.1467-8624.2011.01670.x

Ignace TC Hooge and Guido Camps. 2013. Scan path entropy and arrow plots: capturing scanning behavior of multiple observers. *Frontiers in Psychology* 4 (2013), 996. https://doi.org/10.3389/fpsyg.2013.00996

Pieter Kiefer, Ioannis Giannopoulos, and Martin Raubal. 2014. Where Am I? Investigating Map Matching During Self-Localization With Mobile Eye Tracking in an Urban Environment. *Transactions in GIS* 18, 5 (2014), 660–686. https://doi.org/10.1111/tgis.12067

Oleg V Komogortsev, Denise V Gobert, Sampath Jayarathna, Do H Koh, and Sandeep M Gowda. 2010. Standardization of Automated Analyses of Oculomotor Fixation and Saccadic Behaviors. *IEEE Transactions on Biomedical Engineering* 57, 11 (2010), 2635–2645. https://doi.org/10.1109/TBME.2010.2057429

Michael Land and Mary Hayhoe. 2001. In what ways do eye movements contribute to everyday activities? *Vision Research* 41 (2001), 3559–3565. https://doi.org/10.1016/S0042-6989(01)00102-X

Michael Land, Neil Mennie, and Jennifer Rusted. 1999. The roles of vision and eye movements in the control of activities of daily living. *Perception* 28, 11 (1999), 1311–1328. https://doi.org/10.1068/p2935

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics* 33, 1 (1977), 159–74. https://doi.org/10.2307/2529310

David G Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60, 2 (2004), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

Svenja Marx, Gesine Respondek, Maria Stamelou, Stefan Dowiasch, Josef Stoll, Frank Bremmer, Wolfgang H Oertel, GÃijnter U Hoglinger, and Wolfgang Einhauser. 2012. Validation of mobile eye-tracking as novel and efficient means for differentiating progressive supranuclear palsy from Parkinson's disease. *Frontiers in Behavioral Neuroscience* 6 (2012), 88. https://doi.org/10.3389/fnbeh.2012.00088

Susan M Munn and Jeff B Pelz. 2008. 3D point-of-regard, position and head orientation from a portable monocular video-based eye tracker. In *Proceedings of the 2008 symposium on Eye tracking research & applications*. ACM, New York, NY, USA, 181–188. https://doi.org/10.1145/1344471.1344517

Susan M Munn, Leanne Stefano, and Jeff B Pelz. 2008. Fixation-identification in dynamic scenes: Comparing an automated algorithm to manual coding. In *Proceedings of the 5th symposium on Applied perception in graphics and visualization*. ACM, New York, NY, USA, 33–42. https://doi.org/10.1145/1394281.1394287

Jakob Nielsen. 1994. *Heuristic evaluation*. John Wiley & Sons, New York, NY.

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