The Journal of Open Source Software

EyeDentify3D: A Python package for gaze behavior classification of mobile eye-tracking data

Eve Charbonneau [1,2¶], **Thomas Romeas** [1,2], and **Maxime Trempe** [3]

1 Université de Montréal, Canada **2** Institut national du sport du Québec, Canada **3** Bishop's University, Canada ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 31 October 2025
Published: unpublished

Summary

With the technological advances of mobile eye-tracking technologies, researchers can now place participants in real-world settings (or in virtual environments that simulate the real world) and measure their head and eye orientation to get the gaze orientation in 3D space. This raw gaze orientation is hardly interpretable and must be post-processed to extract gaze behaviours (e.g., fixations, saccades, smooth pursuits, visual scanning). However, most open-source gaze classification algorithms were developed for screen-based eye-tracking, where data is recorded on a 2D plane, and the participant's head is kept still. These algorithms are ill-suited for real-world mobile eye-tracking data (360°), as the gaze-vector origin and endpoint can move substantially due to head rotations, a challenge also present in head-mounted display systems used in virtual reality research. Additionally, eye-tracking researchers often rely on study-specific analysis pipelines, which leads to methodological discrepancies that impede cross-study comparison and the interpretation of results, ultimately limiting our An understanding of gaze behaviour-related phenomena. To address this gap, we developed `EyeDentify3D`, an automated and modular pipeline for analyzing 360° eye-tracking data.

Statement of need

`EyeDentify3D` is a Python package for identifying multiple gaze behaviours (blinks, fixations, saccades, smooth pursuits, visual scannings) from mobile eye-tracking data. It was designed to:

1. Interpret data from various mobile eye-tracking systems (e.g., Pupil Invisible, Tobii), including those embedded in head-mounted displays (e.g., HTC Vive Pro, Pico Neo 3 Pro Eye).
2. Provide a simple user interface that requires only a few lines of code to identify the desired gaze behaviours and extract related metrics.
3. Enable visual inspection of the classification results through figures and animation.

`EyeDentify3D` was designed to be used in science and human performance analysis. Our objective is to distribute the toolbox openly to help researchers more reliably identify and analyze gaze behaviours in real-world scenarios, which involve movements of the head, and promote standardization in gaze analysis, thereby improving our understanding of visual strategies.

State of the field

To address the need for automating the identification of gaze behaviours from eye-tracking data, a few open-source packages have been developed over the years. Most of the packages have focused primarily on the identification of fixations, either from fixed-screen eye-tracker data (Krassanakis et al., 2014) or mobile eye-tracker data Munn & Pelz (2009). Some have extended their identification capabilities to include other behaviours such as saccades, blinks, and micro-saccades, although these have remained limited to fixed-screen eye-trackers (Ghose et al., 2020, @ berger:2012). Notably, none of the existing packages have included the identification of gaze behaviours in dynamic environments that involve large eye and head movements, such as smooth pursuit and visual scanning. Existing eye-tracking data analysis packages are designed for Cartesian coordinates (Munn & Pelz, 2009) or areas of interest based analyses [West:2006]. `EyeDentify3D` differs by interpreting the eye-tracking data in spherical coordinates (360°), which is less prone to vergence errors and avoids the need to pre-define areas of interest, thus its gaze behaviour identification features could not be integrated into existing portable eye-tracking data analysis packages.

Gaze behaviour identification

For each trial recorded during an experiment, the eyes and head rotations are extracted from the data collected by the eye-tracker and the inertial measurement unit, respectively. The gaze orientation (head and eye rotations combined) expressed over a 360° range is then analyzed frame-by-frame. For each frame, the pipeline applies a step-by-step classification based on the following criteria:

1. **Invalid**: The eye-tracker has declared having low confidence in the gaze orientation measurement and considers the data invalid. This often happens when the eyes are closed (e.g., during a blink), the eye orientation is outside the eye-tracker's measurement range, or if the eye-tracker was not positioned properly on the participant.
2. **Blink**: The eye openness is below the user-defined threshold (Chen & Hou, 2021).
3. **Saccade**: Two criteria must be met to detect a saccade. 1) The eye movement must be faster than a dynamical threshold determined using a rolling median over a user-defined window size. 2) The eye movement acceleration must exceed a user-defined threshold for a user-defined number of frames. This ensures that the eyes move rapidly between two targets, accelerating as they leave the first target and decelerating as they approach the second (Van Opstal & Van Gisbergen, 1987).
4. **Visual scanning**: The gaze (head + eyes) velocity is larger than a user-defined threshold (McGuckian et al., 2020). Visual scanning should be identified after saccades, as visual scanning behaviours could also present high eye velocity.
5. **Inter-saccadic interval**: Our inter-saccadic interval classification was adapted from Larsson et al. (2015) implementation, designed for screen-based eye-tracking data, by replacing Cartesian coordinates (2D plane) with spherical coordinates (360° range of motion). The frames that remained unidentified after the previous steps are grouped into intervals. Intervals longer than a user-defined duration threshold are considered inter-saccadic intervals. These intervals are subdivided into windows of a user-defined size. Each window is classified as either coherent or incoherent based on the gaze movement (moving in a consistent direction or not). Adjacent coherent and incoherent windows are merged together to form segments. Then, these segments are further classified as either
6. **fixation** or **smooth pursuit** behaviours based on the four criteria described in Larsson et al. (2015):
   - Dispersion: $p_D < \eta_D$
   - Consistent direction: $p_{CD} > \eta_{CD}$
   - Positional displacement: $p_{PD} > \eta_{PD}$
   - Spatial range: $p_R > \eta_{maxFix}$

All behaviours are mutually exclusive, except for invalid and blink, which can occur together. For example, a frame cannot be classified as both a visual scanning and a smooth pursuit. Thus, the order of identification is important, as the first behaviour identified will take precedence over the others. More details on the definition of events and how they are identified can be found in the documentation.

Finally, `EyeDentify3D` enables the visualization of the classified gaze data and the extraction/export of metrics related to the behaviours (e.g., mean duration, time ratio spent in each behaviour, number of occurrences, saccade amplitude, smooth pursuit trajectory length, etc.).

Software design

The package is organized around two types of classes: `Data` and `BehaviorType`. Classes inheriting from `Data` are responsible for loading, storing, and preprocessing the raw eye-tracking data exported from different eye-trackers, and classes inheriting from `BehaviorType` are responsible for identifying and analyzing specific gaze behaviours. This separation facilitates extending the capabilities of `EyeDentify3D` by independently supporting new eye-trackers or gaze behaviors. The modular design also allows users to customize their analysis pipeline by selecting which gaze behaviours they want to identify and in what order.

Research impact statement

The package has already been used in sport psychology to analyze the gaze behaviour of basketball players (Trempe et al., 2025), and has been used in pilot studies involving trampolinists and boxers. As shown in the examples folder, the package is fully ready to be used by researchers and currently supports four commonly used eye-trackers. Moreover, the package's test coverage exceeds 90%, ensuring the reliability and reproducibility of its results. To help researchers get started with the package, we provide detailed documentation available at https://evecharbie.github.io/EyeDentify3d

Note on the implementation

We believe that the choices made in `EyeDentify3D` are the most suitable for the analysis of gaze behaviour in 3D space (especially in a sporting context). However, we are very open to implementing other identification methods that might be more suitable in other application contexts.

Acknowledgements

This project was supported by a Research and Creative Activity grant from Bishop's University.

Conflict of interest

The authors declare no conflict of interest.

AI usage disclosure

During the preparation of this work, the developer used ChatGPT, Claude, and Copilot to speed up development and enhance code clarity. Aider and Claude were also used to write tests. After using these tools/services, the developer reviewed and edited the content as needed and takes full responsibility for the content of the repository. ChatGPT and Grammarly were also used in the writing of the manuscript to revise the clarity and language.

References

Chen, X., & Hou, W. (2021). Visual fatigue assessment model based on eye-related data in virtual reality. *2021 IEEE 7th International Conference on Virtual Reality (ICVR)*, 262–268. https://doi.org/10.1109/ICVR51878.2021.9483841

Ghose, U., Srinivasan, A. A., Boyce, W. P., Xu, H., & Chng, E. S. (2020). PyTrack: An end-to-end analysis toolkit for eye tracking. *Behavior Research Methods*, *52*(6), 2588–2603. https://doi.org/10.3758/s13428-020-01392-6

Krassanakis, V., Filippakopoulou, V., & Nakos, B. (2014). EyeMMV toolbox: An eye movement post-analysis tool based on a two-step spatial dispersion threshold for fixation identification. *Journal of Eye Movement Research*, *7*(1). https://doi.org/10.16910/jemr.7.1.1

Larsson, L., Nyström, M., Andersson, R., & Stridh, M. (2015). Detection of fixations and smooth pursuit movements in high-speed eye-tracking data. *Biomedical Signal Processing and Control*, *18*, 145–152. https://doi.org/10.1016/j.bspc.2014.12.008

McGuckian, T. B., Beavan, A., Mayer, J., Chalkley, D., & Pepping, G. (2020). The association between visual exploration and passing performance in high-level U13 and U23 football players. *Science and Medicine in Football*, *4*(4), 278–284. https://doi.org/10.1080/24733938.2020.1769174

Munn, S. M., & Pelz, J. B. (2009). FixTag: An algorithm for identifying and tagging fixations to simplify the analysis of data collected by portable eye trackers. *ACM Transactions on Applied Perception (TAP)*, *6*(3), 1–25. https://doi.org/10.1145/1577755.1577759

Trempe, M., Charbonneau, E., Lévesque, V., Ba, A., & Romeas, T. (2025). Expanding the view: Differences in gaze behavior between immersive 360° videos and traditional fixed-screen videos . *Under Review*.

Van Opstal, A. J., & Van Gisbergen, J. A. M. (1987). Skewness of saccadic velocity profiles: A unifying parameter for normal and slow saccades. *Vision Research*, *27*(5), 731–745. https://doi.org/10.1016/0042-6989(87)90071-X

West, J. M., Haake, A. R., Rozanski, E. P., & Karn, K. S. (2006). eyePatterns: Software for identifying patterns and similarities across fixation sequences. *Proceedings of the 2006 Symposium on Eye Tracking Research & Applications*, 149–154. https://doi.org/10.1145/1117309.1117360