The Journal of Open Source Software

Pyxations: organizing, parsing, and analysing eye-movement data using Python

Gonzalo Ruarte[1], **Gustavo Juantorena**[1], **Joaquín González**[1,3], and **Juan Esteban Kamienkowski**[1,2]

1 Laboratorio de Inteligencia Artificial Aplicada, Instituto de Ciencias de la Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires - CONICET, Argentina **2** Maestría de Explotación de Datos y Descubrimiento del Conocimiento, FCEyN-FI, UBA, Argentina **3** Departamento de Física, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 26 December 2025
Published: unpublished

Summary

Pyxations is a Python-based toolbox designed to unify the organization, parsing, and analysis of eye-movement data. It supports multiple common recording formats and detection algorithms, and offers integrated tools for preprocessing, visualization, and downstream analysis. By standardizing these steps and including a BIDS-inspired data structure, Pyxations facilitates reproducibility and makes it easier to compare results across tasks, devices, and studies.

Research Impact Statement

Pyxations has already been deployed to standardize workflows in active research environments. It was utilized to process and analyze eye-tracking data for a study on hybrid search strategies, which combined Bayesian and neural network models (Ruarte et al., 2025).

The library's versatility has been further validated through the successful harmonization of diverse datasets. It has been used to process data from GazePoint 3 trackers, in-house web-based eye-trackers, and driving simulation experiments (courtesy of collaborators mentioned in the acknowledgements), demonstrating its capacity to handle real-world heterogeneity beyond standard laboratory tasks. Furthermore, we are currently collaborating with three additional research groups to integrate Pyxations into their data analysis pipelines, establishing it as a growing standard for reproducible eye-tracking research.

Software Design

Pyxations is designed as a reproducible and extensible framework that unifies the complementary strengths of previous developments while addressing their limitations. Its core contributions can be summarized (Fig. 1) as follows:

Standardized dataset organization. Pyxations enforces a BIDS-inspired hierarchy (subjects, sessions, derivatives) (Gorgolewski et al., 2016), automatically harmonizing file naming and structure across vendors. This facilitates transparent sharing, version control, and collaborative reuse.

Cross-format parsing and harmonization. It supports multiple native formats (e.g., EyeLink .edf/.asc, Tobii, WebGazer, and text-based legacy files) through a unified parsing API, reducing friction when combining data from different acquisition systems.

Calibration- and validation-aware preprocessing. Pyxations directly parses calibration and

The Journal of Open Source Software

validation reports (e.g., EyeLink VALIDATION blocks), extracting per-session accuracy, offset, and drift metrics. These can be used for automated exclusion or weighting of low-quality data.

Flexible trial segmentation. The framework accommodates multiple trialing paradigms: explicit start and end timestamps, event-based markers, or fixed-duration trials. All of them with overlap controls and regular-expression message matching. This flexibility enables consistent parsing of diverse experimental logics without manual preprocessing.

Declarative, provenance-aware workflow. Every preprocessing operation (e.g., interpolation, blink rejection, event detection) is logged automatically in machine-readable JSON recipes and provenance sidecars. This ensures exact reproducibility of analysis pipelines across computing environments.

Scalability and performance. Built on the Polars data engine (Vink, 2022), Pyxations executes parallelized operations on large eye-tracking datasets, significantly outperforming traditional pandas-based workflows (McKinney & others, 2020). This makes it suitable for multi-subject, multi-session analyses typical in modern cognitive experiments.

Visualization, statistics, and inspection tools. In addition to standard gaze plots, Pyxations includes dynamic scanpath visualizations, hierarchical data analysis (experiment, subject, session, trial), per-trial calibration visualization, and task-specific visualization utilities tailored to paradigms such as visual search. Multimatch metrics (Dewhurst et al., 2012) are also embedded at the trial level to compare similarity between scanpaths.

Integration and interoperability. Pyxations can interface with existing libraries such as PyMovements for event detection (Krakowczyk & others, 2023) or PyTrack for visualization within its standardized processing pipeline (Ghose et al., 2020). It thus functions not as a replacement but as an orchestration layer that harmonizes and scales the use of existing tools.

Figure 1

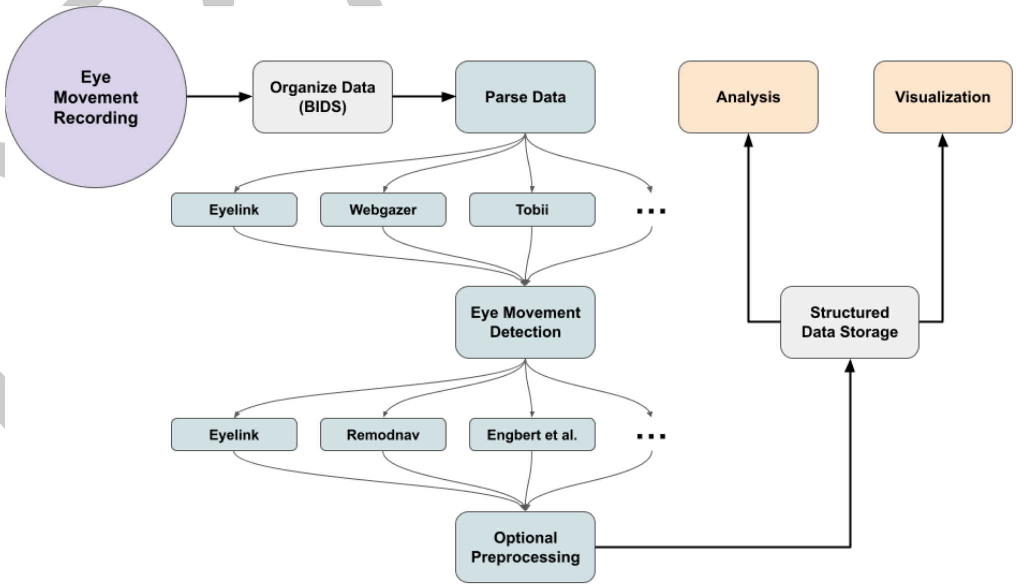

Figure 1: Preprocessing and analysis workflow showing parsing, event detection, preprocessing, and analysis stages.

In summary, Pyxations extends current open-source eye-tracking software by offering a scalable, provenance-aware, and calibration-informed framework that unifies parsing, preprocessing, and analysis within a standardized data structure. We want to keep building upon this framework which is why we decided to use design patterns for code scalability.

Statement of Need

Looking at someone's eyes, searching for your keys or reading, are all active processes in which eye movements take a crucial role (Holmqvist et al., 2011; Land & Tatler, 2009; Rayner, 1998; Wade & Tatler, 2005). These movements include the saccades and fixations, focusing on the most relevant regions of the scene, but also smooth pursuit, microsaccades, or vergence. They are usually measured by optical eye-trackers, which consist of a camera collecting images from the eyes that ultimately provide the position and pupil size of both eyes on the scene. The cameras range from high-speed cameras (up to 2 kHz) to low-cost commercial webcams, and they also differ on the zoom applied and the inclusion of an IR source/filter. Finally, to go from the recording of the eyes' position to actual eye movements, it is necessary to detect such eye movements, for which there are many algorithms available, such as EyeLink (SR Research Ltd., 2021), Engbert and Mergenthaler (Engbert & Mergenthaler, 2006), REMoDNaV (Dar et al., 2021), among others.

Despite the central role of eye movements in vision research, there is currently no standardized pipeline or framework for organizing, parsing, and analyzing eye-tracking data equivalent to the Brain Imaging Data Structure (BIDS) (Gorgolewski et al., 2016). The heterogeneity of experimental paradigms, eye-tracking devices, and event-detection algorithms, combined with the variability introduced by calibration procedures and file formats, makes it difficult to move from raw recordings to reproducible and comparable results. This lack of standardization also complicates cross-study and cross-laboratory comparisons, as even similar experiments often yield data in incompatible formats or rely on different preprocessing pipelines.

State of the Field

In recent years, several open-source toolboxes have been developed to support eye-tracking data processing, including PyMovements (Krakowczyk & others, 2023), PyTrack (Ghose et al., 2020), and SPEED (Lozzi et al., 2025). Each of these offers valuable functionality within the growing ecosystem of open tools for gaze analysis, yet none fully address the combined challenges of dataset heterogeneity, reproducibility, and large-scale workflow automation that Pyxations is designed to solve. Below we summarize their key features and how Pyxations extends or complements them.

PyMovements provides a modular Python interface for parsing, preprocessing, and analyzing eye-tracking data. It supports velocity-based event detection (fixations, saccades), data-quality metrics, and reproducibility guidelines. However, PyMovements is agnostic to data organization and format diversity: it does not enforce or automatically adapt to standardized folder hierarchies (e.g., BIDS-like structures), nor directly support data from multiple tracker vendors or legacy file types.

PyTrack is an end-to-end toolkit featuring fixation, saccade, and microsaccade extraction, estimation of multiple metrics (including pupil and blinks), and visualizations. Its graphical interface and built-in statistical tools make it accessible to non-programmers. Nonetheless, PyTrack is less suited to heterogeneous datasets, as it assumes a relatively uniform input structure. It also provides fewer options for integrating custom detection algorithms or preprocessing modules, making it harder to enforce reproducibility across diverse experimental pipelines.

SPEED (LabSoC Standardized Processing and Extraction of Eye-tracking Data) also focuses on lowering the entry barrier. However, SPEED is tailored to Pupil Labs' data and lacks flexibility.

Therefore, Pyxations was designed as a reproducible and extensible framework that unifies these complementary strengths while addressing their limitations. It is a Python-based toolbox designed to unify the organization, parsing, and analysis of eye-movement data. It supports multiple common recording formats and detection algorithms, and offers integrated tools for preprocessing, visualization, and downstream analysis. By standardizing these steps and including a BIDS-inspired data structure, Pyxations facilitates reproducibility and makes it easier to compare results across tasks, devices, and studies.

AI Usage Disclosure

We utilized generative AI tools (primarily ChatGPT-4o and Gemini 2.5 Flash) to a limited extent, specifically for generating individual methods based on established architectural decisions, creating unit tests, and drafting documentation strings. We did not employ autonomous AI agents or large-scale automated coding pipelines. Regarding the manuscript, these tools were used solely for linguistic refinement, such as typo spotting, grammar correction, and minor stylistic improvements. The core scientific content, figures, and bibliography were not generated by AI.

Acknowledgements

This project was supported by CONICET and UBA. We thank Pablo Laciana for his contribution to the code and Damián Care, Fermín Travi and Bruno Bianchi for their expert insights and constructive discussions during the preparation of this work. We thank Stephanie Muller and Enzo Tagliazucchi for sharing the GazePoint 3 data. Finally, We express our gratitude to Matias Ison for sharing the Driving data.

References

Dar, H., Wagner, G., & Einhäuser, W. (2021). REMoDNaV: Robust eye-movement classification for dynamic stimulation. *Behavior Research Methods*, *53*, 1338–1360. https://doi.org/10.3758/s13428-020-01428-x

Dewhurst, R., Nyström, M., Jarodzka, H., Foulsham, T., Johansson, R., & Holmqvist, K. (2012). It depends on how you look at it: Scanpath comparison in multiple dimensions with MultiMatch, a vector-based approach. *Behavior Research Methods*, *44*(4), 1079–1100. https://doi.org/10.3758/s13428-012-0212-2

Engbert, R., & Mergenthaler, K. (2006). Microsaccades are triggered by low retinal image slip. *Proceedings of the National Academy of Sciences*, *103*(18), 7192–7197. https://doi.org/10.1073/pnas.0509557103

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

Gorgolewski, K. J., Auer, T., Calhoun, V. D., Craddock, R. C., Das, S., Duff, E., & others. (2016). The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments. *Scientific Data*, *3*, 160044. https://doi.org/10.1038/sdata.2016.44

Holmqvist, K., Nyström, M., Andersson, R., Dewhurst, R., Jarodzka, H., & Weijer, J. van de. (2011). *Eye tracking: A comprehensive guide to methods and measures*. Oxford University Press. https://doi.org/10.1080/17470218.2015.1098709

Krakowczyk, M., & others. (2023). PyMovements: An open-source python package for eye movement analysis. *Journal of Open Source Software*, *8*(86), 5235. https://doi.org/10.

1145/3588015.3590134

Land, M. F., & Tatler, B. W. (2009). *Looking and acting: Vision and eye movements in natural behaviour*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780198570943.001.0001

Lozzi, D., Di Pompeo, I., Marcaccio, M., Ademaj, M., Migliore, S., & Curcio, G. (2025). SPEED: A graphical user interface software for processing eye tracking data. *NeuroSci*, *6*(2). https://doi.org/10.3390/neurosci6020035

McKinney, W., & others. (2020). Data structures for statistical computing in python. In *Proceedings of the 9th python in science conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, *124*(3), 372–422. https://doi.org/10.1037/0033-2909.124.3.372

Ruarte, G., Bujia, G., Care, D., Ison, M. J., & Kamienkowski, J. E. (2025). Integrating bayesian and neural networks models for eye movement prediction in hybrid search. *Scientific Reports*, *15*(1), 16482. https://doi.org/10.1038/s41598-025-00272-3

SR Research Ltd. (2021). *Eyelink 1000 plus user manual*.

Vink, R. (2022). *Polars: DataFrame library for rust and python*.

Wade, N. J., & Tatler, B. W. (2005). *The moving tablet of the eye: The origins of modern eye movement research*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780198566175.001.0001