The Journal of Open Source Software

multimatch-gaze: The MultiMatch algorithm for gaze path comparison in Python

Adina S. Wagner[1], **Yaroslav O. Halchenko**[3], **and Michael Hanke**[1, 2]

1 Institute of Neuroscience and Medicine, Brain & Behaviour (INM-7), Research Centre Jülich, Jülich, Germany **2** Institute of Systems Neuroscience, Medical Faculty, Heinrich Heine University Düsseldorf, Düsseldorf, Germany **3** Department of Psychological and Brain Sciences, Dartmouth College, Dartmouth, NH, United States

DOI: 10.21105/joss.01525

Software

- Review
- Repository
- Archive

Submitted: 17 May 2019
Published: 16 August 2019

Summary

`multimatch-gaze` is a Python package for computing the similarity of eye-movement sequences, so called scan paths. Scan paths are the trace of eye-movements in space and time, usually captured with eye tracking devices. Scan path similarity is a measure that is used in a variety of disciplines ranging from cognitive psychology, medicine, and marketing to human-machine interfaces. In addition to quantifying position and order of a series of eye-movements, comparing their temporo-spatial sequence adds an insightful dimension to the traditional analysis of eye tracking data. It reveals commonalities and differences of viewing behavior within and between observers, and is used to study how people explore visual information. For example, scan path comparisons are used to study analogy-making (French, Glady, & Thibaut, 2017), visual exploration and imagery (Johansson, Holsanova, & Holmqvist, 2006), habituation in repetitive visual search (Burmester & Mast, 2010), or spatial attention allocation in dynamic scenes (Mital, Smith, Hill, & Henderson, 2011). The method is applied within individuals as a measure of change (Burmester & Mast, 2010), or across samples to study group differences (French et al., 2017).

Therefore, in recent years, interest in the study of eye movement sequences has sparked the development of novel methodologies and algorithms to perform scan path comparisons. However, many of the contemporary scan path comparison algorithms are implemented in closed-source, non-free software such as Matlab.

`multimatch-gaze` is a Python-based reimplementation of the MultiMatch toolbox for scan path comparison, originally developed by Jarodzka, Holmqvist, & Nyström (2010) and implemented by Dewhurst et al. (2012) in Matlab. This algorithm represents scan paths as geometrical vectors in a two-dimensional space: Any scan path is built up of a coordinate vector sequence in which the start and end position of vectors represent fixations, and the vectors represent saccades. Two such vector sequences are, after optional simplification based on angular relations and amplitudes of saccades, compared on the five dimensions "vector shape", "vector length (amplitude)", "vector position", "vector direction", and "fixation duration" for a multidimensional similarity evaluation.

This reimplementation in Python aims at providing an accessible, documented, and tested open source alternative to the existing MultiMatch toolbox. The algorithm is an established tool for scan path comparison (N. C. Anderson, Anderson, Kingstone, & Bischof, 2015), and improved availability aids adoption in a broader research community. `multimatch-gaze` is available from its Github repository and as the Python package `multimatch-gaze` via `pip install multimatch-gaze`. The module contains the same functionality as the original Matlab toolbox, that is, scan path comparison with optional simplification according to user-defined thresholds, and it provides this functionality via a command line interface or a Python API.

Data for scan path comparison can be supplied as nx3 fixation vectors with columns corresponding to x-coordinates, y-coordinates, and duration of the fixation in seconds (as for the original Matlab toolbox). Alternatively, `multimatch-gaze` can natively read in event detection output produced by REMoDNaV (Dar, Wagner, & Hanke, 2019), a velocity-based eye movement classification algorithm written in Python. For REMoDNaV-based input, users can additionally specify whether smooth pursuit events in the data should be kept in the scan path or discarded.

Acknowledgments

We thank Dr. Richard Dewhurst for kindly and swiftly providing the original Matlab code for the MultiMatch toolbox via e-mail and being supportive of an open source implementation.

References

Anderson, N. C., Anderson, F., Kingstone, A., & Bischof, W. F. (2015). A comparison of scanpath comparison methods. *Behavior Research Methods*, *47*(4), 1377–1392. doi:10.3758/s13428-014-0550-3

Burmester, M., & Mast, M. (2010). Repeated web page visits and the scanpath theory: A recurrent pattern detection approach. *Journal of Eye Movement Research*, *3*(4), 1–20. doi:10.16910/jemr.3.4.5

Dar, A. H., Wagner, A. S., & Hanke, M. (2019). REMoDNaV: Robust eye movement detection for natural viewing. *bioRxiv*. doi:10.1101/619254

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. doi:10.3758/s13428-012-0212-2

French, R. M., Glady, Y., & Thibaut, J.-P. (2017). An evaluation of scanpath-comparison and machine-learning classification algorithms used to study the dynamics of analogy making. *Behavior Research Methods*, *49*(4), 1291–1302. doi:10.3758/s13428-016-0788-z

Jarodzka, H., Holmqvist, K., & Nyström, M. (2010). A vector-based, multidimensional scanpath similarity measure. In *Proceedings of the 2010 symposium on eye-tracking research & applications* (pp. 211–218). ACM. doi:10.1145/1743666.1743718

Johansson, R., Holsanova, J., & Holmqvist, K. (2006). Pictures and spoken descriptions elicit similar eye movements during mental imagery, both in light and in complete darkness. *Cognitive Science*. doi:10.1207/s15516709cog0000_86

Mital, P. K., Smith, T. J., Hill, R. L., & Henderson, J. M. (2011). Clustering of gaze during dynamic scene viewing is predicted by motion. *Cognitive Computation*, *3*(1), 5–24. doi:10.1007/s12559-010-9074-z