HMLET – High-resolution Machine Learning Eye-tracking Toolbox

Alireza Kazemi[a,b], Joy Geng[a,b], and Simona Ghetti[a,b]

[a]Department of Psychology, University of California, Davis, CA 95616

[b]Center for Mind and Brain, University of California, Davis, CA 95616

Corresponding Author: Alireza Kazemi, kazemi@ucdavis.edu and Simona Ghetti, sghetti@ucdavis.edu

Abstract

Eye-tracking technology yields high temporal and spatial resolution data on visual behavior. Detection and quantification of cognitive processes underlying looking behavior can be challenging using conventional processing and statistical analysis. Non-parametric statistical methods are powerful alternatives for high-resolution time series analysis and data driven machine learning techniques can provide an unbiased and explorative description in the analysis of high dimensional data. This paper introduces high-resolution machine learning eye-tracking toolbox (HMLET) that provides an accurate implementation of non-parametric statistical methods in R language and an executable graphic user interface (GUI) comprising novel methods and tools for extraction and analysis of informative measures from eye-tracking data with high temporal resolution. This package supports processing of raw eye-tracking data for basic preprocessing, high-resolution time series analysis, non-parametric permutation tests, data-driven evaluation and selection of eye-tracking measures, a novel visual exploration similarity analysis, along with a sets of visualization tools. HMLET is freely available for download and contribution on GitHub https://github.com/Alireza-Kazemi/MLET, distributed under the MIT license.

Keywords: Eye-tracking, Time-course Analysis, Visual Exploration, Toolbox, Similarity Analysis, Feature Selection

Introduction

Eye-tracking is a well-established method to collect data on looking behaviors (Mele & Federici, 2012). Looking behaviors are governed by numerous cognitive processes including, but not limited, to perception and visual attention (Borys & Plechawska-Wójcik, 2017; Yu et al., 2022), emotion (Lim et al., 2020), decision-making (Gidlöf et al., 2013; Selmeczy et al., 2021; Yu et al., 2022), and memory (Oakes et al., 2013; Pathman & Ghetti, 2016; Richmond & Nelson, 2009). In addition, there is evidence that eye movements are influenced by the autonomic nervous system (McDougal & Gamlin, 2014). As a result, even in well-controlled experiments, eye-tracking data reflect the combination of ongoing cognitive and autonomic processes as well as recording-related errors (Carter & Luke, 2020). As a result, it can be challenging to detect and quantify what aspects of looking behaviors reflect a specific cognitive process (e.g., episodic memory), particularly when the effect of interest is small and/or its temporal dynamics are likely changing over the course of the trial. In order to make the best use of eye-tracking data, researchers should use methods that maximize their ability to utilize the available data without sacrificing precision and leverage robust statistical methods to compare relevant experimental conditions.

Most studies average specific measures (e.g., looking proportions) across many timepoints (Hannula, 2010). Although this practice is typically used to improve the chance of detecting a hypothesized effect reliably by avoiding multiple comparison correction (Hannula & Ranganath, 2009; Pathman & Ghetti, 2016), there is the risk of losing important information about temporal dynamics. Capturing the high temporal resolution of the eye-tracking data can increase statistical power especially when the target effect is hypothesized to vary across time and its size may be limited (Andersson et al., 2010; Carter & Luke, 2020; Prasse et al., 2020). Modern eye-tracking devices can record gaze positions up to 2000 times per second (2000 Hz) resulting in a dense time series of gaze points with 0.5ms time interval between samples. Such a high temporal resolution may seem unnecessary when one considers that it takes tens of milliseconds for the visual cortex to receive the visual input (Albrecht et al., 2002; Shapley et al., 2007). However, eye-tracking technology relies on discrete time sampling of eye movements and based on the Nyquist-Shannon theorem (Shannon, 1949) the sampling rate must be at least twice the bandwidth of the actual signal. Bandwidth of a signal is defined as the difference between upper and lower rate of change in the signal (Oppenheim & Schafer, 1975). Eye movements include very slow drifts to ultra-rapid saccades that can peak at 650 degrees per seconds in some individuals (Wilson et al., 1993). Sampling rate for a signal with peak bandwidth of 650hz should be at least 1300hz to provide enough data for accurate estimation of all eye-movements from slow drifts and steady fixations to ultra-rapid saccades. Therefore, the sampling frequency of an eye-tracker limits the estimation of fast dynamics in the eye-movements consistent with the evidence that accurate detection of fast events can benefit from higher temporal resolutions (Kirchner & Thorpe, 2006).

As a result, analyzing the time series of eye-movements may provide more accurate estimation of the dynamics of the targeted effects compared to estimating fixations and saccades collapsed within pre-determined time bins. To mitigate the multiple-comparison problem in the analysis of time series data, permutation tests are a suitable solution (Maris & Oostenveld, 2007). In some research, the same comparison might be repeated across different measures and conditions (Carter & Luke, 2020; Wang et al., 2020). Critically, permutation tests are computationally demanding, and multiple permutation tests to investigate hypothesized effects on all possible measures (e.g., looking proportions, switching frequency, and saccades frequency) not only demand for high computational resource, but also may increase the chance of false discovery through “fishing” (Carter & Luke, 2020). Therefore, it is helpful to explore novel methods capable of cross validating a custom set of measures. Similar challenges have been addressed in the field of machine learning through data-driven methods of feature evaluation/selection (Ang et al., 2016; Solorio-Fernández et al., 2020). For the sake of consistency, we are going to use the term feature to refer to measures obtained from eye-tracking data. In this paper, we introduce a package for analysis of high-temporal resolution eye-tracking time series, data driven evaluation and selection of numerous features, and a novel visual exploration similarity analysis.

Statistical Analysis of the Time Series

Commercial eye-tracking software solutions such as Tobii Pro Lab, formerly known as Tobii Pro Studio (Tobii AB, 2024) are usually limited to a specific eye-tracking hardware and provide limited analytical methods. However, most of them can export raw or preprocessed data for further analysis in other software and packages. Several open-source packages have been introduced for eye-tracking data analysis with different capabilities. EyetrackingR (Dink, J. W. & Ferguson, B., 2015) is the most cited package that provides tools to perform different analyses including onset contingent analysis (Fernald et al., 2008), growth curve analysis (Mirman et al., 2008), and divergence analysis (Wendt et al., 2014). Some of these methods use permutation tests for statistical comparison and to the best of our knowledge, eyetrackingR is the only package that provides permutation test for high-temporal resolution time series of eye-tracking data. However, the current version of the eyetrackingR does not account for the repetition of the permutation samplings. In a permutation test, the null distribution is estimated nonparametrically through a Monte Carlo approximation which includes computing the desired statistic in a relatively large number of unique permutation of the input data (Maris & Oostenveld, 2007). If one does not control for the repeated permutations, false positive or false negative results may be observed depending on the number of observations, simulations, and the likelihood and distribution of the repeated permutations (Mahadevan, 1997; Rubinstein & Kroese, 2016).

Moreover, as Maris and Oostenveld (2007) discussed, in within-subject studies, permutations can be completed either across subject-level averages (i.e., subject-level permutations) or across trials within each subject (i.e., trial-level permutations). Critically, in studies with small sample sizes, the total number of possible subject-level permutations theoretically limits the accuracy of p-value estimation. For example, in a study with 5 participants with 2 within-subject conditions, only $2^5 = 32$ unique subject-level permutations are possible hence the smallest possible p-value is 1/32 = 0.031. In cases of small sample sizes, but large numbers of trials available for each participant, trial-level permutation is a better estimation tool, which is not available in eyetrackingR or other open-source eye-tracking packages.

In HMLET, we implemented both trial-level and subject-level permutation tests, equipped with routines to produce unique permutation labels. We also provided a couple of preprocessing routines for checking data for common issues in ordering and indexing, resampling, binning in time, and interpolation (See Time Series Analysis Section).

Informative Measures

The field has generated several tools to preprocess and analyze eye-movement. For example, GazeR (Geller et al., 2020) and PyTrack (Ghose et al., 2020) are successful open-source packages which provided a strong set of tools for different stages of eye-tracking data analysis including, preprocessing of the raw data, common and novel feature and parameter extraction, statistical analysis and visualization. Despite the strengths of existing analytical tools, to our knowledge, there is no available eye-tracking package that supports unbiased and comprehensive feature analysis, which is a necessary step to conduct a formal and replicable data exploration in a principled fashion (Carter & Luke, 2020). The use of systematic and reproducible analytical methods for feature evaluation and selection across different studies is crucial to improve replicability and reliability of the scientific findings. In HMLET, we provided a set of complementary tools to perform feature evaluation and provide a set of temporal and overall ranking of the measures in terms of their informativeness in dissociating specified experimental conditions (See Feature Analysis Section).

Visual Exploration

Another desirable ability of eye-tracking packages is to produce estimates of visual exploration, given its importance for attention allocation (Borys & Plechawska-Wójcik, 2017; Yu et al., 2022) and information processing (Krajbich et al., 2010; Krajbich & Rangel, 2011). Scanpath similarity analysis has been shown to be an informative eye-tracking measure which quantifies the extent to which visual exploration is similar across different conditions or between encoding and retrieval (Fahimi & Bruce, 2021). Fixation density maps are conceptually a mapped version of scanpaths on a 2 dimensional plane, which have been widely used in the similarity-based analysis of visual exploration (Engelke et al., 2013; Olsen et al., 2014). For example, similarity analysis of fixation density maps in a memory task showed that revisiting the same fixation points that were visited during the encoding, in the retrieval improve visuospatial memory (Wynn et al., 2018). Eyesim (https://github.com/bbuchsbaum/eyesim) is an opensource package that provides similarity analysis for fixation density maps. Critically, the temporal resolution of the fixation density maps is bound to the fixation durations, and this is not necessarily temporally comparable across participants.

In HMLET, we introduced a novel method for visual exploration similarity analysis in which we used time series of gaze points rather than fixations to estimate the pattern of the gaze distribution across time and space. We also provided some routines to create output videos (See Visual Exploration Similarity Analysis Section).

HMLET workflow

In the following sections, we provide an overview of the functions and capabilities implemented in HMLET. The workflow and main modules are presented in Figure 1.

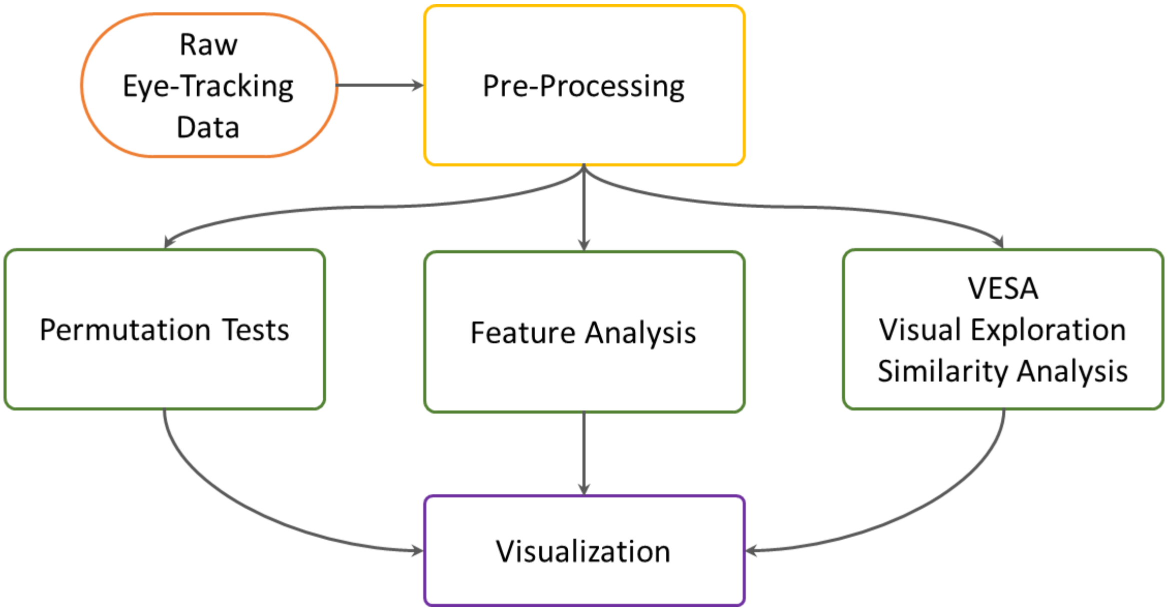

Figure 1. HMLET workflow. Raw eye-tracking data should include gaze point coordinates, time stamps, and grouping variables in CSV format. Pre-processing includes functions and routines for AOI definition, gaze point mapping, and data preparation for the subsequent tools. Feature analysis includes functions and routines for feature evaluation and selection based on machine learning. VESA includes functions and routines for visual exploration similarity analysis. Visualization includes functions and routines to visualize data and results.

The preprocessing module is designed to be compatible with all types of eye-tracking technology that supports CSV or TXT file export. We encourage users to export raw gaze-point coordinates with time stamps to preserve high temporal resolution and use pre-processing functions to define AOIs. However, HMLET is amenable for users exporting AOIs and fixation data computed by their software of choice. In this case, users would skip pre-processing steps in HMLET and use the functions for data preparation for the desired module which convert user data to a compatible format for HMLET modules.

Data Structure

Data must be prepared in the long format. General data structure should include subject index (within-trial permutation tests support single subject data), time variable indicating time stamp in milliseconds (e.g., 16, 32, 48, …), or indexes (e.g., 1, 2, 3, …), grouping variables indicating trial or subject level conditions, and response types (e.g., hit, false-alarm) and eye-tracking variables indicating gaze point coordinates on screen or they can be relative coordinates with respect to a fixed or changing center (e.g., fixation [x, y] coordinate relative to the stimuli center), areas of interest, and gaze proportions. Additional variables can be included, and some can be replaced as an alternative to the listed required variables. For example, HMLET supports reverse time analysis which allows for examining eye-movement patterns backward in time from a specific incident (e.g., response submission) which can be replaced with the normal time stamp or time index variables. See Table.1 for details.

Table 1. Eye-tracking data structure supported by HMLET. Required variables are the variables that must exist in the data for some or all modules of HMLET. Computable variables are those that HMLET can compute for the user if they are not already in the data.

Categories	*Variables*	*Details*	*Required*	*Computable*
Subjects	IDs	Unique Subject IDs (categorical)	✓	-
Time Variables	Time Stamp	time in milliseconds/index (numerical)	✓	-
	Time Bin	arbitrary time bin (ordinal)	-	✓
	Forward Index	forward time index (ordinal)	-	✓
	Reverse Index	backward time index (ordinal)	-	-
	Sampling Duration	duration of each sample in milliseconds (numerical)	✓	-
Grouping Variables	Within Subjects	trial level (categorical)	✓	-
	Between Subjects	subject level (categorical)	✓	-
	Response Type	trial level accuracy (categorical)	✓	-
Eye-Tracking	Gaze coordinates	[x, y] on screen at each time point (numerical)	✓	-
	Relative coordinates	[x, y] relative to a center at each time point (numerical)	-	✓
	AOIs	Areas of interest (categorical)	✓	✓
	Gaze proportions	looking proportions (numerical)	✓	✓
	Features	Different Eye-tracking Measures	-	✓

HMLET Modules in R

Preprocessing

Eye-tracking data in raw format consist of a temporal sequence of 2-dimensional gaze coordinates within the experimental screen per trial/subject/condition. Data must be loaded into a variable of type dataframe in R and sorted in long format. The standard preprocessing pipeline available in HMLET for raw data is shown in Figure 2. We suggest using these modules in the presented order to avoid unwanted error propagation. In the following sections, we provide detailed functions of each module.

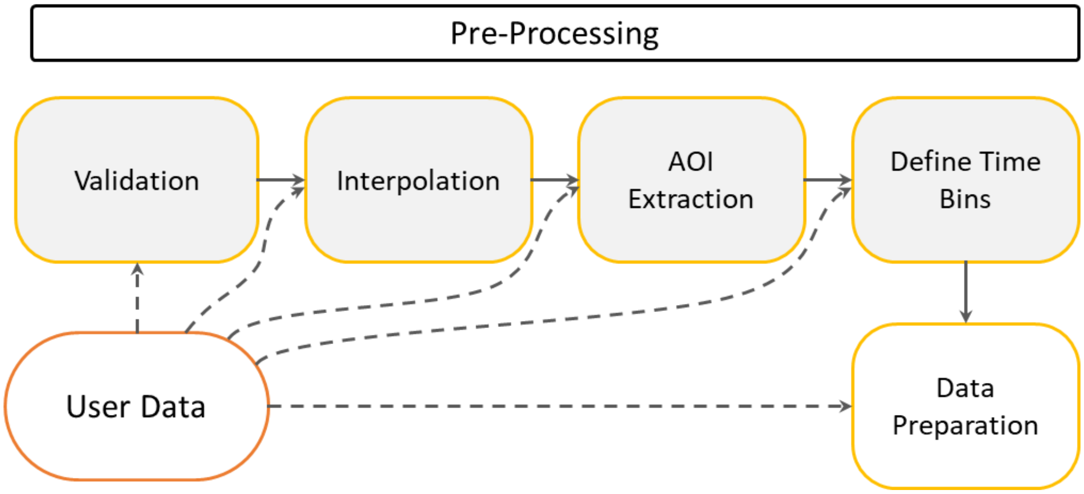

Figure 2. Preprocessing modules offered in HMLET. These modules are suggested to be used in the presented order. Validation, Interpolation, AOI Extraction, Relative Gaze Points, and Define Time Bins, modules are not mandatory if the corresponding process was already performed in the user data.

Validation. In the analysis of time series data, it is crucial to have the exact time stamp of all data points. The time interval between samples must also be equal to implement permutation tests because average values in each time point across trials and participant ID is used to compute the statistic that is later used to form clusters across time points. Even a small deviation or mislabeling of the time stamps between trials and participants can result in large errors in the average time series and corresponding permutation tests. In practice, due to truncation errors, it is very common to have slight deviations in the time stamps exported by the eye-trackers. For example, for an eye-tracker with sampling rate of 60hz the time interval between samples will be 16.(6)ms, which has an indefinitely repeating 6 in its decimal, but a rounded or truncated version of this number will be stored by the software which may cause varying time intervals across samples (e.g., with presumably an ideal 60hz sampler, time intervals frequently alternate between 16ms and 17ms). As a result, in practice, there can be discrepancies in time intervals between samples that have to be fixed before proceeding to permutation tests. The function *ComputeSamplingInterval_HMLET* provides estimates of the sampling interval in the given data. This function produces a warning if there are discrepancies in the current data. To fix the issue, *Preprocess_FixSampling_HMLET*, can be used in which all the data points will be labeled with consistent time stamps.

We strongly suggest using the function *Preprocess_CheckData_HMLET* for quality to check data for critical issues such as consistency of time stamps and identification of missing values due to blink or head movements.

Interpolation. It is very common to have missing gaze points in any given temporal sequence of gaze coordinates. This may happen due to blinking, the participant not looking within the screen area, or device error. Critically, when it comes to high temporal resolution analysis of eye-tracking data, removing missing gaze points results in misalignment of data time points across trials. Users can estimate gaze point trajectory for missing time points using the *Preprocess_Interpolate_HMLET* function. This function performs linear interpolation for the specified number of consecutive missing gaze points and leaves the value of longer consecutive missing gaze points as *NA*. This step is not mandatory, and it can be ignored.

AOI Extraction. This step adds a categorical column to the data indicating the name of the AOI the subject is looking at during each time point. Users need to provide 4 arguments for rectangular AOIs as {Center_X, Center_Y, Width, Height}, indicating center of each AOI and its width and height in pixels with respect to top left corner of the screen (i.e., origin {0, 0}). Areas outside the listed AOI will be labeled as Content (user can change the label). The function *Preprocess_AOIExtraction_HMLET* performs these computations. The AOI column is mandatory for permutation tests, but if it is already available in the data this step can be ignored.

Define Time Bins. Time bin is defined as an interval of time across which individual time point data is aggregated. This is most commonly used for looking proportions. Time bins can be defined as arbitrary time intervals (not smaller than sampling duration) using the function *CreateTimeBinData_HMLET*. This step is not mandatory. However, it is recommended with high temporal resolution data with many missing gaze points. The aggregating function can be set by the user and is set to the *mean()* function by default. This function first creates new columns for all possible AOIs with binary values indicating whether gaze point was inside the AOI or not. If these binary AOI columns are already presented in the data, this step will be ignored. Subsequently, it applies the aggregating function on these binary AOIs columns within time bins for each level of the grouping variables defined by the user. If users do not specify any grouping variables, time point data will not be aggregated (e.g., averaged, summed, etc.) within time bins and only the time bin information will be added to the current data frame in the form of two additional columns. One column specifies the corresponding time interval for each sample as labels (e.g., for time bins of size 250ms: “0-250”, “250-500”, …), and the other column specifies the index of the time bins in numerical format (e.g., 1, 2, 3, …). This is not a mandatory step and user may ignore this step.

Data Preparation. This is a mandatory step through which data column names are converted to a compatible format for HMLET modules. To prepare data for permutation tests, the function *PermutationTestDataPrep_HMLET* should be called. Alternatively, to prepare the data for GUI use and performing feature analysis and VESA modules, the function *ExportDataForGUI_HMLET* should be used.

Time Series Analysis

In HMLET, we implemented both subject-level and trial-level permutation tests as introduced by Maris and Oostenveld, (2007) for time series data. Critically, to have a valid estimation of the null distribution, individual permutations of conditions must be unique. If not, the Monte Carlo assumption would be violated and the estimated distribution would be invalid (Romano & Tirlea, 2022). In HMLET, we first produce unique random permutation labels for the number of simulations defined by the user. Subsequently, we perform simulations on these labels to estimate the null distribution. Users can directly pass the prepared data to the function *PermutationTest_HMLET*, which runs the permutation test pipeline at the subject-level or trial-level based on the specified binary value to the input argument *permuteTrialsWithinSubject* (i.e., trial-level will be used if it is set as true, otherwise participants level permutation will be performed). In principle, both methods are expected to produce the same results when ample number of permutation samples are tested (Maris & Oostenveld, 2007). However, in practice, with small numbers of subjects contributing data, desired number of permutations across subjects may not be possible or enough to reliably estimate the p-value. In these cases, with large number of trials for each experimental condition, trial-level permutation is more appropriate. Subject-level permutation is computationally faster and when enough subjects are available, users may prefer to run subject-level tests. Trial-level permutation test can compare experimental conditions in one subject which is implemented in a separate function *PermutationTestSingleParticipant_HMLET*. Results of the permutation tests can be visualized using functions *PlotTimeSeries_HMLET* and *PlotNullDistribution_HMLET*.

HMLET GUI Modules

Feature analysis and VESA modules are available in the GUI which is implemented in MATLAB™ (2024a). To run the GUI, MATLAB™ compiler runtime is required which is freely available on MathWorks™ website[1]. Source scripts for these modules are also available which

[1] https://www.mathworks.com/products/compiler/matlab-runtime.html

can be directly used in MATLAB™ environment. Users can use *ExportDataForGUI_HMLET* function in R to prepare their data for the GUI.

Feature Analysis

The workflow of the feature analysis module is represented in Figure 3. In this module, desired features (i.e., eye-tracking measures) extracted from the data are analyzed to score their informativeness in discriminating between specified conditions. Note that any measure of eye-tracking can be treated as a feature, but only conceptually comparable ones can be analyzed together in the feature analysis. Therefore, one application of the feature analysis module is to identify the optimal parameters for a specific eye-tracking measure. For instance, in practice, the size of AOI around a stimulus (e.g., fixation cross) is often defined heuristically, especially when the stimulus lacks explicit borders. As a result, the value of an eye-tracking measure (e.g., looking time at that area) may vary with different sizes of AOIs with significant impact on the results. As preliminary assessment, all possible values of the looking time resulting from different AOI sizes can be compared in the feature analysis.

Features must be added in separate columns in the long format. In addition to the features provided by the user, HMLET can compute some basic features from the preprocessed data as listed in Table 2. If features were computed within different time windows during trials, their informativeness within each time window would be compared and scored separately. After outlier removal and normalization of the features, based on a user defined categorical dependent variable (e.g., Accuracy, Experimental Condition, etc.), features are ranked using chi square tests (Jović et al., 2015) and minimum redundancy (Ding & Peng, 2005). Subsequently, the ranking of the features can be validated on the performance of a KNN or SVM classifier (can be chosen by the user) to predict the desired trial/subject level variable using the feature time series.

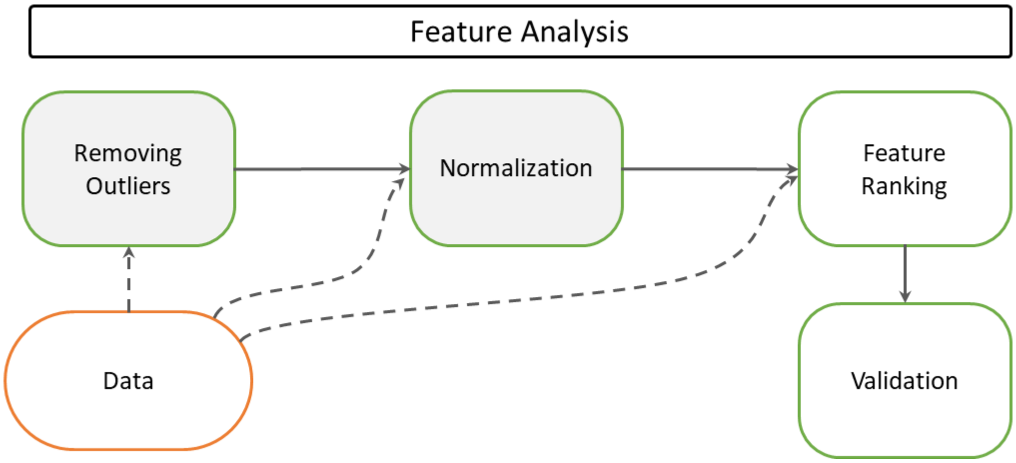

Figure 3. Workflow of feature analysis module in HMLET. This pipeline automatically runs in the GUI. Users can mark Feature Extraction, Removing Outliers, and Normalization steps to be ignored if these steps are not necessary for their input data.

In this step, a set of features will be extracted from the preprocessed data. Features can be extracted from absolute gaze points, relative gaze points, and/or gaze proportions. By default, HMLET extracts different groups of features listed in Table 2. Users can choose all or a subset of these features. Alternatively, users can provide their own features in the input data and ignore the feature extraction step.

Table 2. Default features HMLET feature analysis module extracts from eye-tracking data.

Feature Type	**Details**	**Unit**
Unique Gaze Points	Number of unique gaze points in each AOI and overall	Count
Unique Spots	Number of unique gaze clusters, grouped based on a user defined spot size	Number
Revisited Spots	Number of time same spots are revisited	Number
Looking Time	The time spent within each AOI and overall	Time(ms)
Area of Visit	Total area explored within each AOI and overall	Pixel Square
Total Distance	Total distance explored within each AOI and overall	Pixel
Average Distance	Average of Euclidean distance between consecutive gaze points	Pixel
Number of switches	Number of time gaze points switched between AOIs	Count
Looking Proportions	Proportional viewing time of the AOIs	Percent
Vicinity Score	Scoring gaze points based on their vicinity to a specified point	Pixel

Visual Exploration Similarity Analysis (VESA)

VESA is a novel technique in which absolute or relative gaze point coordinates are used to compute similarity of gaze patterns between two conditions (e.g., Similarity between encoding and retrieval in a memory task). VESA builds an exploration map which is a 2-dimensional representation of the gaze distribution within a sliding time window (with user defined length and overlap). Therefore, a set of exploration maps will be computed for each trial. Users can choose to compare exploration-maps of the corresponding time windows between trials which will produce a time series of similarity scores for each pair of compared trials. Alternatively, exploration maps within each trial can be stacked temporally and similarity computed between the 3D volumetric representations which will result one similarity score for each pair of compared trials (See Figure 4). In both cases, similarity scores are computed using a Pearson's correlation.

Each exploration map is created in 3 steps. First, a 2-dimensional pictorial representation of all gaze points in the selected time window is created as a 2D picture with the same size as the original screen in which each pixel takes the value corresponding to the number of gaze points in that coordinate. Second, the 2D images are smoothed by a specified factor. Finally, these images are rescaled with the desired factor defined by user. Rescaling smaller images is necessary for computational efficiency. However, if ample computational resources are available, users may use factor 1 for rescaling. For smoothing, users can choose between three different methods: grid-based, Gaussian, and convolution. In grid-based smoothing, for a smoothing factor *R*, the screen is divided into grids of $R \times R$ pixels. Smoothing and scaling are performed simultaneously by assigning one pixel to each grid, with its value being the sum of all pixel values within that grid.

Gaussian smoothing applies a 2D Gaussian filter with a standard deviation of *R* to the image. In convolution smoothing, the value of each pixel is replaced with the average value of all pixels within a circle of radius *R* centered on that pixel. These methods offer different approaches to smoothing based on user preferences and requirements.

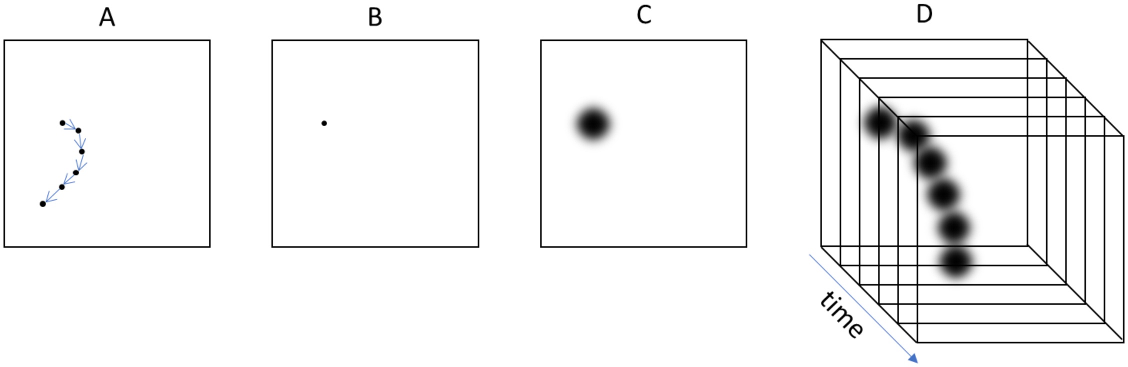

Figure 4. Steps of creating an exploration map. A) Hypothetical scan-path with only 1 gaze point at each fixation point depicted by black dots. B) Two-dimensional pictorial representation of the first gaze point. C) Exploration-map representation for the first gaze point. D) Three-dimensional representation of the scan-path by stacking exploration-maps of the gaze points in their temporal order.

Conclusions

In conclusion, the introduction of this new toolbox for eye-tracking data analysis fills a critical gap in the current landscape of eye-tracking research tools. Given the ubiquitous application of eye-tracking technology across various fields of research, the availability of comprehensive and consistent analytical tools is paramount for comparability and replicability of research findings. HMLET addresses needs that are not currently fully met by providing new and essential functions. By providing unique tools for high-temporal time-series analysis, this toolbox is a complementary package to the currently available open-source toolboxes for eye-tracking data analysis. The current version of HMLET is mainly developed to support common experimental designs performed on the screen. Critically, the current version does not directly support virtual reality or eye-tracking data from 3D environments. The computational performance of the current implementation can also be improved. As a result, the package is released under MIT license and completely open source therefore, the research community can contribute to improving the current version.

Funding

The development of this package is partly supported by UC Davis Department of Psychology Summer Research Fellowship and Dissertation Research Fellowship awarded to A.K.

Competing interest statement

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

Albrecht, D. G., Geisler, W. S., Frazor, R. A., & Crane, A. M. (2002). Visual Cortex Neurons of Monkeys and Cats: Temporal Dynamics of the Contrast Response Function. *Journal of Neurophysiology*, *88*(2), 888–913. https://doi.org/10.1152/jn.2002.88.2.888

Andersson, R., Nyström, M., & Holmqvist, K. (2010). Sampling frequency and eye-tracking measures: How speed affects durations, latencies, and more. *Journal of Eye Movement Research*, *3*(3). https://doi.org/10.16910/jemr.3.3.6

Ang, J. C., Mirzal, A., Haron, H., & Hamed, H. N. A. (2016). Supervised, Unsupervised, and Semi-Supervised Feature Selection: A Review on Gene Selection. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *13*(5), 971–989. https://doi.org/10.1109/TCBB.2015.2478454

Borys, M., & Plechawska-Wójcik, M. (2017). *Eye-tracking metrics in perception and visual attention research*. 13.

Carter, B. T., & Luke, S. G. (2020). Best practices in eye tracking research. *International Journal of Psychophysiology*, *155*, 49–62. https://doi.org/10.1016/j.ijpsycho.2020.05.010

Ding, C., & Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, *3*(2), 185–205. https://doi.org/10.1142/s0219720005001004

Dink, J. W., & Ferguson, B. (2015). *eyetrackingR: An R Library for Eye-tracking Data Analysis.* [R]. http://www.eyetrackingr.com

Engelke, U., Liu, H., Wang, J., Le Callet, P., Heynderickx, I., Zepernick, H.-J., & Maeder, A. (2013). Comparative Study of Fixation Density Maps. *IEEE Transactions on Image Processing*, *22*(3), 1121–1133. https://doi.org/10.1109/TIP.2012.2227767

Fahimi, R., & Bruce, N. D. B. (2021). On metrics for measuring scanpath similarity. *Behavior Research Methods*, *53*(2), 609–628. https://doi.org/10.3758/s13428-020-01441-0

Fernald, A., Zangl, R., Portillo, A. L., & Marchman, V. A. (2008). Looking while listening: Using eye movements to monitor spoken language comprehension by infants and young children. In *Developmental psycholinguistics: On-line methods in children's language processing* (pp. 97–135). John Benjamins Publishing Company. https://doi.org/10.1075/lald.44.06fer

Geller, J., Winn, M. B., Mahr, T., & Mirman, D. (2020). GazeR: A Package for Processing Gaze Position and Pupil Size Data. *Behavior Research Methods*, *52*(5), 2232–2255. https://doi.org/10.3758/s13428-020-01374-8

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

Gidlöf, K., Wallin, A., Dewhurst, R., & Holmqvist, K. (2013). Using Eye Tracking to Trace a Cognitive Process: Gaze Behaviour During Decision Making in a Natural Environment. *Journal of Eye Movement Research*, *6*(1), Article 1. https://doi.org/10.16910/jemr.6.1.3

Hannula, D. E. (2010). Worth a glance: Using eye movements to investigate the cognitive neuroscience of memory. *Frontiers in Human Neuroscience*, *4*. https://doi.org/10.3389/fnhum.2010.00166

Hannula, D. E., & Ranganath, C. (2009). The Eyes Have It: Hippocampal Activity Predicts Expression of Memory in Eye Movements. *Neuron*, *63*(5), 592–599. https://doi.org/10.1016/j.neuron.2009.08.025

Jović, A., Brkić, K., & Bogunović, N. (2015). A review of feature selection methods with applications. *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 1200–1205. https://doi.org/10.1109/MIPRO.2015.7160458

Kirchner, H., & Thorpe, S. J. (2006). Ultra-rapid object detection with saccadic eye movements: Visual processing speed revisited. *Vision Research*, *46*(11), 1762–1776. https://doi.org/10.1016/j.visres.2005.10.002

Krajbich, I., Armel, C., & Rangel, A. (2010). Visual fixations and the computation and comparison of value in simple choice. *Nature Neuroscience*, *13*(10), 1292–1298. https://doi.org/10.1038/nn.2635

Krajbich, I., & Rangel, A. (2011). Multialternative drift-diffusion model predicts the relationship between visual fixations and choice in value-based decisions. *Proceedings of the National Academy of Sciences*, *108*(33), 13852–13857. https://doi.org/10.1073/pnas.1101328108

Lim, J. Z., Mountstephens, J., & Teo, J. (2020). Emotion Recognition Using Eye-Tracking: Taxonomy, Review and Current Challenges. *Sensors*, *20*(8), Article 8. https://doi.org/10.3390/s20082384

Mahadevan, S. (1997). Monte Carlo Simulation. In *Reliability-Based Mechanical Design*. CRC Press.

Maris, E., & Oostenveld, R. (2007). Nonparametric statistical testing of EEG- and MEG-data. *Journal of Neuroscience Methods*, *164*(1), 177–190. https://doi.org/10.1016/j.jneumeth.2007.03.024

McDougal, D. H., & Gamlin, P. D. (2014). Autonomic Control of the Eye. In *Comprehensive Physiology* (pp. 439–473). John Wiley & Sons, Ltd. https://doi.org/10.1002/cphy.c140014

Mele, M. L., & Federici, S. (2012). Gaze and eye-tracking solutions for psychological research. *Cognitive Processing*, *13*(S1), 261–265. https://doi.org/10.1007/s10339-012-0499-z

Mirman, D., Dixon, J. A., & Magnuson, J. S. (2008). Statistical and computational models of the visual world paradigm: Growth curves and individual differences. *Journal of Memory and Language*, *59*(4), 475–494. https://doi.org/10.1016/j.jml.2007.11.006

Oakes, L., Baumgartner, H., Barrett, F., Messenger, I., & Luck, S. (2013). Developmental changes in visual short-term memory in infancy: Evidence from eye-tracking. *Frontiers in Psychology*, *4*. https://www.frontiersin.org/articles/10.3389/fpsyg.2013.00697

Olsen, R. K., Chiew, M., Buchsbaum, B. R., & Ryan, J. D. (2014). The relationship between delay period eye movements and visuospatial memory. *Journal of Vision*, *14*(1), 8. https://doi.org/10.1167/14.1.8

Oppenheim, A. V., & Schafer, R. W. (1975). *Digital Signal Processing*. Prentice-Hall.

Pathman, T., & Ghetti, S. (2016). More to it than meets the eye: How eye movements can elucidate the development of episodic memory. *Memory*, *24*(6), 721–736. https://doi.org/10.1080/09658211.2016.1155870

Prasse, P., Jäger, L. A., Makowski, S., Feuerpfeil, M., & Scheffer, T. (2020). On the Relationship between Eye Tracking Resolution and Performance of Oculomotoric Biometric Identification. *Procedia Computer Science*, *176*, 2088–2097. https://doi.org/10.1016/j.procs.2020.09.245

Richmond, J., & Nelson, C. A. (2009). Relational memory during infancy: Evidence from eye tracking. *Developmental Science*, *12*(4), 549–556. https://doi.org/10.1111/j.1467-7687.2009.00795.x

Romano, J. P., & Tirlea, M. A. (2022). Permutation testing for dependence in time series. *Journal of Time Series Analysis*, jtsa.12638. https://doi.org/10.1111/jtsa.12638

Rubinstein, R. Y., & Kroese, D. P. (2016). *Simulation and the Monte Carlo Method*. John Wiley & Sons.

Selmeczy, D., Kazemi, A., & Ghetti, S. (2021). Developmental Differences in Subjective Recollection and Its Role in Decision Making. *Child Development*, *92*(6). https://doi.org/10.1111/cdev.13611

Shannon, C. E. (1949). Communication in the Presence of Noise. *Proceedings of the IRE*, *37*(1), 10–21. https://doi.org/10.1109/JRPROC.1949.232969

Shapley, R., Hawken, M., & Xing, D. (2007). The dynamics of visual responses in the primary visual cortex. In *Progress in Brain Research* (Vol. 165, pp. 21–32). Elsevier. https://doi.org/10.1016/S0079-6123(06)65003-6

Solorio-Fernández, S., Carrasco-Ochoa, J. A., & Martínez-Trinidad, J. Fco. (2020). A review of unsupervised feature selection methods. *Artificial Intelligence Review*, *53*(2), 907–948. https://doi.org/10.1007/s10462-019-09682-y

Tobii AB. (2024). *Tobii Pro Lab* (1.241) [Computer software]. Tobii AB. https://www.tobii.com/

Wang, Y., Yu, S., Ma, N., Wang, J., Hu, Z., Liu, Z., & He, J. (2020). Prediction of product design decision Making: An investigation of eye movements and EEG features. *Advanced Engineering Informatics*, *45*, 101095. https://doi.org/10.1016/j.aei.2020.101095

Wendt, D., Brand, T., & Kollmeier, B. (2014). An Eye-Tracking Paradigm for Analyzing the Processing Time of Sentences with Different Linguistic Complexities. *PLOS ONE*, *9*(6), e100186. https://doi.org/10.1371/journal.pone.0100186

Wilson, S. J., Glue, P., Ball, D., & Nutt, D. J. (1993). Saccadic eye movement parameters in normal subjects. *Electroencephalography and Clinical Neurophysiology*, *86*(1), 69–74. https://doi.org/10.1016/0013-4694(93)90068-7

Wynn, J. S., Olsen, R. K., Binns, M. A., Buchsbaum, B. R., & Ryan, J. D. (2018). Fixation reinstatement supports visuospatial memory in older adults. *Journal of Experimental Psychology: Human Perception and Performance*, *44*(7), 1119–1127. https://doi.org/10.1037/xhp0000522

Yu, X., Johal, S. K., & Geng, J. J. (2022). Visual search guidance uses coarser template information than target-match decisions. *Attention, Perception, & Psychophysics*, *84*(5), 1432–1445. https://doi.org/10.3758/s13414-022-02478-3