Package: kollaR (via r-universe)

February 20, 2025

Type Package

Title Filtering, Visualization and Analysis of Eye Tracking Data

Version 1.0.4

Description Functions for analysing eye tracking data, including event detection (I-VT, I-DT and two means clustering), visualizations and area of interest (AOI) based analyses. See separate documentation for each function. The principles underlying I-VT and I-DT filters are described in Salvucci & Goldberg (2000,\doi{10.1145/355017.355028}). Two-means clustering is described in Hessels et al. (2017, \doi{10.3758/s13428-016-0822-1}).

License GPL-3

Encoding UTF-8

LazyData true

Imports dplyr, ggplot2, zoo, ggforce, tidyr, ggpubr, jpeg, patchwork, shiny, plotly, base64enc, magick, scales

RoxygenNote 7.3.1

NeedsCompilation no

Author Johan Lundin Kleberg [aut, cre]

Maintainer Johan Lundin Kleberg `<johan.lundin.kleberg@su.se>`

Depends R (>= 3.5.0)

Date/Publication 2025-02-20 17:20:05 UTC

Config/pak/sysreqs cmake libfontconfig1-dev libfreetype6-dev make libmagick++-dev gsfonts libicu-dev libjpeg-dev libssl-dev zlib1g-dev

Repository https://drjohanlk.r-universe.dev

RemoteUrl https://github.com/cran/kollaR

RemoteRef HEAD

RemoteSha f8eb29da52cdf98868df72e02d2901a6a207d602

Contents

`kollaR-package` *Filtering, Visualization, and Analysis of Eye Tracking Data*

Description

Functions for analyzing eye-tracking data, including fixation filtering/event detection (I-VT, I-DT, and two-means clustering), visualizations, and area of interest (AOI) based analyses. See separate documentation for each function. Make sure it works with your data. The principles underlying I-VT and I-DT filters are described in Salvucci & Goldberg (2000,doi:10.1145/355017.355028). Two-means clustering is described in Hessels et al. (2017, doi:10.3758/s1342801608221).

Details

Overview of functions: Pre-processing (smoothing, interpolation, downsampling):'process_gaze', 'downsample_gaze'

Fixation and Saccade Detection (Fixation Filters)**: 'ivt_filter', 'idt_filter', 'cluster2m'

Visualization of Output from Fixation Filter and Preprocessing Algorithms: 'filt_plot_temporal', 'filt_plot_2d', 'plot_velocity_profiles', 'plot_sample_velocity', 'plot_filter_results'

Visualization of Gaze Data: ‘static_plot‘, ‘animated_fixation_plot‘

AOI Based Analyses: ‘draw_aoi‘, ‘aoi_test‘

Author(s)

Johan Lundin Kleberg <`johan.lundin.kleberg@su.se`>

`animated_fixation_plot` *Create GIF animation of fixations on a stimulus images*

Description

This function plots and returns a .gif showing fixations on a background with one or multiple images, typically the stimuli. The interval to plot is defined by sample numbers. Fixations must have the variables x, y, and onset. The function works with .jpg images. If paths to multiple images are given, all will be displayed. Fixations are shown on a white background if no background images are defined. .gif images can be saved to a file. Gaze data are plotted on a reversed y-axis where x and y are 0 is the upper left corner, corresponding to the structure of data from Tobii eye trackers. If there are multiple participants specified in the variable id, each participant will get a unique color. You may get an error message if some participants lack data during single frames. This is usually no cause for concern.

Usage

```
animated_fixation_plot(
  gazedata,
  xres = 1920,
  yres = 1080,
  plot.onset,
  plot.offset,
  background.images = NA,
  filename = "scanpath.gif",
  save.gif = FALSE,
  gif.dpi = 300,
  gazepoint.size = 2,
  n.loops = 1,
  show.legend = TRUE,
  id_color_map = NA,
  resolution.scaling = 0.5,
  framerate = 10,
  show.progress = TRUE
)
```

Arguments

`gazedata`	Data frame with fixation data which must include columns for x and y coordinates as well as the variable onset which indicates the onset of the fixation. Make sure the onset variables match the timing the plot.onset and plot.offset input. If the categorical or factor variable id is included, separate colors will represent each participant. Make sure the onset variables match the timing the plot.onset and plot.offset input.
`xres`	horizontal resolution of the screen or area to plot on. Default 1920
`yres`	vertical resolution of the screen or area to plot on. Default 1080
`plot.onset`	Onset of the interval in the gaze_data$onset variable to plot in the same unit, typically milliseconds
`plot.offset`	Offset of the interval in the corresponding to the variable onset in the input data frame gazedata to plot in the same unit, typically milliseconds
`background.images`	data frame with information about background images to use as background. The data frame must include the variables min.x, min.y, max.x, and max.y variables representing where the images should be placed on the background, the variable path specifying a full file path, and the onset and offset of each image in units corresponding to the time stamps of the gazedata matrix. Background images should be in JPEG format. This is an example: `background.images <- data.frame( min.x = c(100, 800), min.y = c(100, 100), max.x = c(300, 100), max.y = c(600, 600), path = c("~/path_to_image1/image1.jpg", "~/path_to_image1/image2.j` `onset = c(1, 4000), offset = c(4000, 6000)`
`filename`	Name of path where the .gif is saved
`save.gif`	If TRUE, save the created .gif file under the name specified in the filename parameter
`gif.dpi`	Resolution in dpi if .gif is saved. Lower values give smaller files.
`gazepoint.size`	Size of marker representing fixation coordinates.
`n.loops`	Specify the number of times to play the plotted sequence. Default is 1. If n.loops is 0, the .gif will play in an eternal loop
`show.legend`	If TRUE, show values of the variable id in legend
`id_color_map`	A character vector with HEX color codes for each id. If `NA`, a color map with unique colors for each id is created by the function. You can create a specific color map for your data using the following code: new_color_map <- `c("#FB61D7", "#00C094")` names(new_color_map) <- `c("Id1", "Id2")`
`resolution.scaling`	Scaling of the original images and gaze data. Default is 0.5. Decreasing the size of the images can make the function quicker. This can be useful if you want to assign specific colors for different groups
`framerate`	Frames per seconds of the returned animation. Default 10
`show.progress`	If TRUE, show progression of the function in the prompt

Value

a magick animation of raw and fixated values plotted on the y axis and sample number on the x axis

---

`aoi_test` *Test whether a gaze coordinates are within or outside a rectangular or elliptical AOI. The aois df must contain the variables x0, x1, y0 and y1. x0 is the minimum x value, y0 the minimum y value. x1 the maximum x value. y1 the maximum y value and type where rect means that the AOI is a rectangle and circle that the AOI is a circle or ellipse If a column called name is present, the output for each AOI will be labelled accordingly. Otherwise, the output will be labelled according to the order of the AOI in the data frame. The df 'gaze' must contain the variables onset, duration, x, and y. Latency will be defined as the value in onset of the first detected gaze coordinate in the AOI Make sure that the timestamps are correct! The function can be used with gaze data either fixations, saccades, or single samples. Note that the output variables are not equally relevant for all types of gaze data. For example, both total duration and latency are relevant in many analyses focusing on fixations, but total duration may be less relevant in analyses of saccades.*

Description

Test whether a gaze coordinates are within or outside a rectangular or elliptical AOI. The aois df must contain the variables x0, x1, y0 and y1. x0 is the minimum x value, y0 the minimum y value. x1 the maximum x value. y1 the maximum y value and type where rect means that the AOI is a rectangle and circle that the AOI is a circle or ellipse If a column called name is present, the output for each AOI will be labelled accordingly. Otherwise, the output will be labelled according to the order of the AOI in the data frame. The df 'gaze' must contain the variables onset, duration, x, and y. Latency will be defined as the value in onset of the first detected gaze coordinate in the AOI Make sure that the timestamps are correct! The function can be used with gaze data either fixations, saccades, or single samples. Note that the output variables are not equally relevant for all types of gaze data. For example, both total duration and latency are relevant in many analyses focusing on fixations, but total duration may be less relevant in analyses of saccades.

Usage

```
aoi_test(gaze, aois, outside = FALSE)
```

Arguments

`gaze`	data frame with each row representing a gaze coordinate from a fixation, saccade, or sample. Must include the variables x, y, duration, and onset. Onset zero should typically be trial onset
`aois`	data frame with AOIs.
`outside`	If FALSE, summarize data within AOIs. If TRUE, summarize data outside AOIs.

Value

data frame with total duration, number of occurrences and latency to first detected gaze coordinates for each AOI. Data are in long format.

`cluster2m` *Fixation detection by two-means clustering*

Description

Identify fixations in a gaze matrix using identification by two-means clustering. The algorithm is based on Hessels et al 2017. Behavior research methods, 49, 1802-1823. Data from the left and right eye are not processed separately. Adjust your analysis scripts to include this steps if you want the algorithm to include this step, as in Hessels et al 2017. Input data must be a data frame with the variables timestamp, x.raw and y.raw as variables. Other variables can be included but will be ignored. This function does not perform pre-processing in the form of interpolation or smoothing. Use the function process.gaze for this. Timestamps are assumed to be in milliseconds. Default settings assume that x and y coordinates are in pixels. The output data is a list with two data frames: fixations includes all detected fixations with coordinates, duration and a number of other metrics, filt.gaze is a sample-by-sample data frame with time stamps, raw and filtered gaze coordinates for fixations. If the input downsampling.factors is not empty, transition weights will be calculated based on the data in the original sampling rate and data at all sampling rate specified in this variable. According to Hessels et al 2017, this step makes the analysis less vulnerable to noise in the data.

Usage

```
cluster2m(
  gaze_raw,
  windowlength.ms = 200,
  distance.threshold = 0.7,
  one_degree = 40,
  window.step.size = 6,
  min.fixation.duration = 40,
  weight.threshold = 2,
  xcol = "x.raw",
  ycol = "y.raw",
  merge.ms.threshold = 40,
  downsampling.factors = NA,
  missing.samples.threshold = 0.5
)
```

Arguments

`gaze_raw`	Data frame with unfiltered gaze data. Include the variable timestamp with timing in ms and columns with raw x and y data as specified by the parameters xcol and ycol or their default values
`windowlength.ms`	Length of the moving analysis windows

`distance.threshold`	Subsequent fixations occurring withing this distance are merged. Set to 0 if you do not want to merge fixations.
`one_degree`	One degree of the visual field in the unit of the raw x and y coordinates, typically pixels
`window.step.size`	Distance between starting points of subsequent analysis windows in samples
`min.fixation.duration`	Minimum duration of accepted fixations. Shorter fixations are discarded
`weight.threshold`	Samples with a transition weight exceeding it are candidates for fixation detection.
`xcol`	Name of the column where raw x values are stored. Default: x.raw
`ycol`	Name of the column where raw y values are stored. Default: y.raw
`merge.ms.threshold`	Only fixations occurring within this time window in milliseconds are merged
`downsampling.factors`	Factors to downsample the data by in calculating fixation weights. If downsampling.factors has the values `c(10, 2)`, transition weights will be calculated base on data in the original sampling rate as well as the two donwsampled data sets.
`missing.samples.threshold`	Remove fixations with a higher proportion of missing samples. Range 0 to 1.

Value

list including separate data frames for fixations and sample-by-sample data including filtered and unfiltered data. The "fixations" data frame gives onset, offset, x, y, RMSD and missing samples of each fixation.

Examples

```
gaze <- cluster2m(sample.data.processed)
```

`downsample_gaze` *Downsample gaze*

Description

This function downs-samples gaze by a specified factor. Data are down-sampled by splitting the data in bins and calculating the mean of each bin.

Usage

```
downsample_gaze(data_in, ds.factor, xcol = "x", ycol = "y")
```

Arguments

`data_in`	Data frame which must contain the variables specified by the parameters xcol and ycol.
`ds.factor`	The factor to down-sample by. For example, setting ds.factor to 10 down-samples data recorded at 1000 HZ to 100 HZ.
`xcol`	Name of the column where raw x values are stored. Default: x
`ycol`	Name of the column where raw y values are stored. Default: y

Value

Data frame with downsampled gaze data. The output variables are x, y, and the numbers of the first and last samples of the original data frame included in the bin.

`draw_aois` *Draw one or more areas of interest, AOIs, on a stimulus image and save to the R prompt. The input is the path to a 2D image. Supported file formats: JPEG, BMP, PNG. The function returns a data frame with all saved AOIs. By default, AOIs are drawn in a coordinate system where y is 0 in the lower extreme of the image, e.g., an ascending y axis. Tobii eye trackers use a coordinate system with a descending y-axis, e.g., x and y are 0 in the upper left corner of the image. Make sure that your AOIS match the coordinate system of your eye tracker output. By setting the parameter reverse.y.axis to TRUE, the saved AOIs will be reformatted to fit a coordinate system with a descending y-axis. All AOIS have the variables x0, x1, y0 and y1. x0 is the minimum x value, y0 the minimum y value. x1 the maximum x value. y1 the maximum y value*

Description

Draw one or more areas of interest, AOIs, on a stimulus image and save to the R prompt. The input is the path to a 2D image. Supported file formats: JPEG, BMP, PNG. The function returns a data frame with all saved AOIs. By default, AOIs are drawn in a coordinate system where y is 0 in the lower extreme of the image, e.g., an ascending y axis. Tobii eye trackers use a coordinate system with a descending y-axis, e.g., x and y are 0 in the upper left corner of the image. Make sure that your AOIS match the coordinate system of your eye tracker output. By setting the parameter reverse.y.axis to TRUE, the saved AOIs will be reformatted to fit a coordinate system with a descending y-axis. All AOIS have the variables x0, x1, y0 and y1. x0 is the minimum x value, y0 the minimum y value. x1 the maximum x value. y1 the maximum y value

Usage

```
draw_aois(image.path, reverse.y.axis = FALSE)
```

Arguments

`image.path`	path to a valid image file with the suffix .jpeg, .jpg, .png or .bmp
`reverse.y.axis`	If TRUE, save AOIs positioned on a reverse Y-axis where y is 0 in the upper end of the image. Note that AOIs will be converted to fit a reversed Y axis before printed in the R prompt and saved, but will be shown in the original coordinate system when plotted inside the function.

Value

data frame with type and coordinates of drawn AOIs

`filt_plot_2d` *Plot fixation filtered vs. raw or unfiltered gaze coordinates in 2D space.*

Description

This function plots and returns a ggplot2 figure showing fixation filtered and raw gaze coordinates plotted against time. The interval to plot can be defined as a proportion of the data frame or by sample numbers. This function uses one data.frame with fixations and one with sample-by-sample raw data

Usage

```
filt_plot_2d(
  raw.data,
  filtered.data,
  plot.window = c(NA, NA),
  raw.columns = c("x.raw", "y.raw"),
  filt.columns = c("x", "y"),
  fixation.radius = 40,
  xres = 1920,
  yres = 1080,
  verbose = TRUE
)
```

Arguments

`raw.data`	gaze matrix which must include columns for filtered and unfiltered data as specified in the raw.columns parameter
`filtered.data`	Data frame with fixation data which must include columns for filtered x and y data as specified in the raw.columns parameter as well as the variable onset which indicates the onset of the fixation. Make sure the onset varables match the timing in the raw.data df

`plot.window`	vector defining the time window to plot. If left empty, the 50-65 <0, they are assumed to be proportions, e.g., `plot.window = c(0.3,0.35)` plots the 30-35 percent of max.length interval of the data. Numbers >1 are assumed to refer to sample order in the data
`raw.columns`	Names of variable containing raw data. Default x.raw and y.raw
`filt.columns`	Names of variable containing filtered data. Default x and y
`fixation.radius`	Radius of circles showing fixations.
`xres`	horizontal resolution of the screen or area to plot on. Default 1920
`yres`	vertical resolution of the screen or area to plot on. Default 1080
`verbose`	if TRUE, print the resulting plot

Value

a ggplot of raw and fixated values plotted on the y axis and sample number on the x axis

---

`filt_plot_temporal` *Plot fixation filtered vs. raw gaze coordinates*

Description

This function plots and returns a ggplot2 figure showing two time series of gaze coordinates plotted against time. The interval to plot can be defined as a proportion of the data frame or by sample numbers. Use this function to plot data before and after processing or filtering to examine their effects. For example, unprocessed x or y coordinates can be plotted against x and y coordinates following pre-processing and/or a fixation filter. Either the x or the y vector is plotted

Usage

```
filt_plot_temporal(
  data_in,
  plot.window = c(NA, NA),
  var1 = "x.raw",
  var2 = "x",
  verbose = TRUE
)
```

Arguments

`data_in`	gaze matrix which must include columns for filtered and unfiltered data as specified in the var1 and var2 paramters
`plot.window`	vector defining the time window to plot. If left empty, the 50-65 <0, they are assumed to be proportions, e.g., `plot.window = c(0.3,0.35)` plots the 30-35 in the data
`var1`	Name of the first variable to plot. Default "x.raw"
`var2`	Name of the second variable to plot (overlayed on var1) Default: "x"
`verbose`	If TRUE, print the resulting plot

Value

a ggplot with gaze coordinates plotted on the y axis and sample number on the x axis

Examples

```
new.plot <- filt_plot_temporal(sample.data.filtered, plot.window = c(1000, 2000))
```

`find.transition.weights` *Find transition weights for each sample in a gaze matrix.*

Description

This function is used internally by the function cluster2m

Usage

```
find.transition.weights(data_in, window.step.size = 6, windowsize)
```

Arguments

`data_in`	Input data
`window.step.size`	Step size
`windowsize`	Window size

Value

transition weights.

`idt_filter` *Dispersion-based fixation detection algorithm* `(I-DT)`

Description

Apply a dispersion-based fixation `(I-DT)` filter to the eye tracking data. The algorithm identifies fixations as samples clustering within a spatial area. The procedure is described in Blignaut 2009 Input data must be a data frame with the variables timestamp, x.raw and y.raw as variables. Other variables can be included but will be ignored. This function does not perform pre-processing in the form of interpolation or smoothing. Use the function process.gaze for this. Timestamps are assumed to be in milliseconds. Default settings assume that x and y coordinates are in pixels. The output data is a list with two data frames: fixations includes all detected fixations with coordinates, duration and a number of other metrics, filt.gaze is a sample-by-sample data frame with time stamps, raw and filtered gaze coordinates. The function can be slow for long recordings and/or data recorded at high sampling rates.

Usage

```
idt_filter(
  gaze_raw,
  one_degree = 40,
  dispersion.threshold = 1,
  min.duration = 50,
  xcol = "x.raw",
  ycol = "y.raw",
  distance.threshold = 0.7,
  merge.ms.threshold = 75,
  missing.samples.threshold = 0.5
)
```

Arguments

`gaze_raw`	Data frame with unfiltered gaze data. Include the variable timestamp with timing in ms and columns with raw x and y data as specified by the paramerers xcol and ycol or their default values
`one_degree`	One degree of the visual field in the unit of the raw x and y coordinates, typically pixels
`dispersion.threshold`	Maximum radius of fixation candidates. Samples clustering within a circle of this limit will be classified as a fixation if the duration is long enough.
`min.duration`	Minimum duration of fixations in milliseconds
`xcol`	Name of the column where raw x values are stored. Default: x.raw
`ycol`	Name of the column where raw y values are stored. Default: y.raw
`distance.threshold`	Subsequent fixations occurring withing this distance are merged. Set to 0 if you don't want to merge fixations.
`merge.ms.threshold`	Only subsequent fixations occurring within this time window are merged
`missing.samples.threshold`	Remove fixations with a higher proportion of missing samples. Range 0-1

Value

list including separate data frames for fixations and sample-by-sample data including filtered and unfiltered data. The fixations data frame includes onset, offset, x, y, RMSD and missing samples of each fixation.

Examples

```
idt_data <- idt_filter(sample.data.processed)
```

`interpolate_with_margin` *Interpolate over gaps (subsequent NAs) in vector.*

Description

Interpolate over gaps (subsequent NAs) in vector.

Usage

```
interpolate_with_margin(data_in, marg, max_gap)
```

Arguments

`data_in`	Vector to interpolate in
`marg`	Margin in samples before and after gap to use for interpolation
`max_gap`	Maximum length of gaps in sample

Value

vector with interpolated gaps

`ivt_filter` *I-VT algorithm for fixation and saccade detection*

Description

Apply an `I-VT` filter to the eye tracking data. The algorithm identifies saccades as periods with sample-to-sample velocity above a threshold and fixations as periods between saccades. See Salvucci and Goldberg 2000. Identifying fixations and saccades in eye tracking protocols. Proc. 2000 symposium on Eye tracking research and applications for a description.

Input data must be a data frame with the variables timestamp, x.raw and y.raw as variables. Other variables can be included but will be ignored. This function does not perform pre-processing in the form of interpolation or smoothing. Use the function process.gaze for this. Timestamps are assumed to be in milliseconds. Default settings assume that x and y coordinates are in pixels. The output data is a list with three data frames: fixations includes all detected fixations with coordinates, duration and a number of other metrics, saccades includes data for saccades, filt.gaze is a sample-by-sample data frame with time stamps, raw and filtered gaze coordinates for fixations. The function has a number of parameters for removing potentially invalid fixations and saccades. The parameter min.fixation.duration can be used to remove unlikely short fixations. If the parameter missing.samples threshold is set to a value lower than 1, fixations with a higher proportion of missing raw samples are removed.

Usage

```
ivt_filter(
  gaze_raw,
  velocity.filter.ms = 20,
  velocity.threshold = 35,
  min.saccade.duration = 10,
  min.fixation.duration = 40,
  one_degree = 40,
  save.velocity.profiles = FALSE,
  xcol = ”x.raw”,
  ycol = ”y.raw”,
  distance.threshold = 0.7,
  merge.ms.threshold = 75,
  missing.samples.threshold = 0.5
)
```

Arguments

`gaze_raw`	Data frame with unfiltered gaze data. Include the variable timestamp with timing in ms and columns with raw x and y data as specified by the parameters xcol and ycol or their default values
`velocity.filter.ms`	Window in milliseconds for moving average window used for smoothing the sample to sample velocity vector.
`velocity.threshold`	Velocity threshold for saccade detection in degrees/second
`min.saccade.duration`	Minimum duration of saccades in milliseconds
`min.fixation.duration`	Minimum duration of fixations in milliseconds
`one_degree`	One degree of the visual field in the unit of the raw x and y coordinates, typically pixels
`save.velocity.profiles`	If TRUE, return velocity profiles of each detected saccade as a variable in the saccades data frame
`xcol`	Name of the column where raw x values are stored. Default: x.raw
`ycol`	Name of the column where raw y values are stored. Default: y.raw
`distance.threshold`	Subsequent fixations occurring withing this distance are merged. Set to 0 if you don’t want to merge fixations.
`merge.ms.threshold`	Subsequent fixations occuring within this time window and distance specified by distance.threshold are merged. Set to 0 if you don’t want to merge fixations.
`missing.samples.threshold`	Remove fixations with a higher proportion of missing samples. Range 0 to 1.

Value

list including separate data frames for fixations and sample-by-sample data including filtered and unfiltered data. The fixations data frame gives onset, offset, x, y, RMSD and missing samples of each fixation.

Examples

```
ivt_data <- ivt_filter(sample.data.processed, velocity.threshold = 30, min.fixation.duration = 40)
```

`merge_adjacent_fixations`

Merge adjacent fixations

Description

Merge fixations which appear close in space and time. This function is called by other functions and typically not used outside them

Usage

```
merge_adjacent_fixations(
  fixations,
  gaze_raw,
  distance.threshold = 0.5,
  ms.threshold = 75,
  one_degree = 40
)
```

Arguments

`fixations`	Data frame with fixations
`gaze_raw`	Data matrix with raw data. See description of the ivt_filter function
`distance.threshold`	Subsequent fixations occurring withing this distance are merged. Set to 0 if you don't want to merge fixations.
`ms.threshold`	Maximum time elapsed between fixations to be merged.
`one_degree`	One degree of the visual field in the scale of the x and y coordinates. Typically pixels

Value

A new data frame with fixations

`plot_filter_results` *Plot validity measures from one or more fixation detection algorithms*

Description

This function visualizes validity measures of fixations detected with one or more fixation detection algorithms. The function is tested for fixation data frames generated with kollaR event detection algorithms. By default, the function can plot Root Mean Square Deviations of detected fixations, fixation duration and the proportion of missing raw samples. The output data is a ggplot which can be modified further outside the function. If you want to use this function to compare more than one fixation detection algorithms, combine them using the function rbind in base R. For example, `rbind(my_data1[["fixations"]], my_data2[["fixations"]])` would generate a combined data frame with the fixations detected by two event classification procedures.

Usage

```
plot_filter_results(data_in, plot.variable = "rmsd")
```

Arguments

`data_in`	Data frame with fixations to plot
`plot.variable`	Variable to plot. If left empty, RMSD of fixations are plotted. Alternatives are "rmsd", "duration", "missing.samples"

Value

A ggplot with visualizations of the selected validity measure

Examples

```
plot_filter_results(data_in = sample.data.fixations, plot.variable = "rmsd")
```

`plot_sample_velocity` *Plot the sample-to-sample velocity of eye tracking data.*

Description

This function visualizes the sample-to-sample velocity in a period of eye tracking data. This can be helpful when determining a suitable velocity threshold for saccade detection Input data must be a data frame with the variables timestamp, x.raw and y.raw as variables. Other variables can be included but will be ignored. This function does not perform pre-processing in the form of interpolation or smoothing. Use the function process.gaze for this. Timestamps are assumed to be in milliseconds. Default settings assume that x and y coordinates are in pixels. The output data is a plot of sample-to-sample velocity in the selected interval.

Usage

```
plot_sample_velocity(
  data_in,
  velocity.filter.ms = 20,
  plot.window = c(NA, NA),
  xcol = "x.raw",
  ycol = "y.raw",
  threshold.line = NA,
  one_degree = 40,
  verbose = TRUE
)
```

Arguments

`data_in`	Data frame with gaze data to plot. Include the variable timestamp with timing in ms and columns with raw x and y data as specified by the paramerers xcol and ycol or their default values
`velocity.filter.ms`	Window in milliseconds for moving average window used for smoothing the sample-to-sample velocity vector.
`plot.window`	vector defining the time window to plot. If left empty, the 50-65 <0, they are assumed to be proportions, e.g., plot.window = c(0.3,0.35) plots the 30-35 percent of max.length interval of the data. Numbers >1 are assumed to refer to sample order in the data
`xcol`	Name of the column where raw x values are stored. Default: "x.raw"
`ycol`	Name of the column where raw y values are stored. Default: "y.raw"
`threshold.line`	Can be specified to add a line showing a potential velocity threshold for saccade detection. No threshold is shown if this parameter is NA
`one_degree`	One degree of the visual field in the unit of the raw x and y coordinates, typically pixels
`verbose`	If TRUE, print the resulting plot

Value

A ggplot showing the sample-to-sample velocity of the selected data interval

Examples

```
plot_sample_velocity(data_in = sample.data.processed, threshold.line = 35)
```

`plot_velocity_profiles` *Create ggplot of saccade velocity profiles*

Description

This function plots and returns a ggplot showing velocity profiles of saccades plotted against time. Saccades should be generated with the ivt.filter functions with the save.velosity.profiles parameter set to TRUE. The interval to plot is defined by saccade number as they appear in the sacccades data frame.

Usage

```
plot_velocity_profiles(saccades, onset = NA, offset = NA, verbose = TRUE)
```

Arguments

`saccades`	data frame including saccades. Each saccade must have a list with a vector of the veloctity profiles
`onset`	first saccade to plot. The value must correspond to a number in the variable "number" in the saccades data frame. If left empty, all saccades are plotted
`offset`	last saccade to plot. The value must correspond to a number in the variable "number" in the saccades data frame.
`verbose`	If TRUE, print the resulting plot

Value

ggplot with velocity profiles

Examples

```
new.plot <- plot_velocity_profiles(sample.data.saccades, onset = 10, offset = 20)
```

`process_gaze` *Interpolation and smoothing of gaze-vector*

Description

Preprocessing of gaze vector Interpolate over gaps in data and smooth the x and y vectors using a moving average filter. The gaze vector must contain the variables timestamp, and variables containing unfiltered x and y coordinates. Default names: x.raw and y.raw. Timestamps are assumed to be in milliseconds. The unprocessed x and y variables are kept under the names x.unprocessed and y.unprocessed for comparison. The function will add the variable timestamp.t to the data frame before returning. This is a theoretical timestamp based on the detected median sample-to-sample timestamp difference as compared to the actual registered time stamps in the data. This can be useful in some validation analyses.

Usage

```
process_gaze(
  gaze_raw,
  max_gap_ms = 75,
  marg_ms = 30,
  filter_ms = 15,
  xcol = "x.raw",
  ycol = "y.raw"
)
```

Arguments

`gaze_raw`	Data frame containing unfiltered timestamp, x.raw and y.raw vectors.
`max_gap_ms`	The maximum gaps defined as subsequent NAs in the data to interpolate over in milliseconds. Default 75 ms
`marg_ms`	The margin in milliseconds before and after the gap to use as basis for interpolation.
`filter_ms`	The size of the moving average window to use in smoothing. Default 15 ms
`xcol`	Name of column containing unprocessed x coordinates
`ycol`	Name of column containing unprocessed y coordinates

Value

data frame with gaze data after interpolation and filtering

Examples

```
processed_gaze <- process_gaze(sample.data.unprocessed)
```

`sample.data.filtered` *Fixation-filtered sample-by-sample example data*

Description

This dataset contains data from 1 individuals during a free viewing tasks after pre-processing. Data were recorded at 1200 Hz using a Tobii Pro Spectrum eye tracker

Usage

```
sample.data.filtered
```

Format

A data frame

timestamp timestamp in ms recorded by the eye tracker

x.raw unfiltered gaze position x

y.raw unfiltered gaze position y

x fixation filtered gaze position x

y fixation filtered gaze position y

Source

The dataset was stored in the package at 'data/example_data.RData'

`sample.data.fixation1` *Fixations from 1 individual*

Description

This dataset contains fixation data from 1 individuals during a free viewing tasks. Data were recorded at 1200 Hz using a Tobii Pro Spectrum eye tracker

Usage

```
sample.data.fixation1
```

Format

A data frame

x fixation filtered gaze position x

y fixation filtered gaze position y

duration duration of fixation in milliseconds

onset onset of fixation in milliseconds

offset offset of fixation in milliseconds

rmsd Root mean square deviation of included samples from the centroid of the fixation

fixation.filter Name of the fixation filter algorithm

threshold Threshold setting for the fixation filter algorithm

id Participant id

Source

The dataset was stored in the package at 'data/example_data.RData'

`sample.data.fixations` *Fixations from 7 individuals*

Description

This dataset contains fixation data from 7 individuals during a free viewing tasks. Data were recorded at 1200 Hz using a Tobii Pro Spectrum eye tracker

Usage

```
sample.data.fixations
```

Format

A data frame

x	fixation filtered gaze position x
y	fixation filtered gaze position y
duration	duration of fixation in milliseconds
onset	onset of fixation in milliseconds
offset	offset of fixation in milliseconds
rmsd	Root mean square deviation of included samples from the centroid of the fixation
fixation.filter	Name of the fixation filter algorithm
threshold	Threshold setting for the fixation filter algorithm
id	Participant id

Source

The dataset was stored in the package at 'data/example_data.RData'

`sample.data.processed` *Pre-processed sample-by-sample example data*

Description

This dataset contains data from 1 individuals during a free viewing tasks after pre-processing. Data were recorded at 1200 Hz using a Tobii Pro Spectrum eye tracker

Usage

```
sample.data.processed
```

Format

A data frame

id participant number
timestamp timestamp in ms recorded by the eye tracker
x.raw gaze position x
y.raw gaze position y
timestamp.t "'Theoretical timestamp' for comparison."
sample sample nr in recording

Source

The dataset was stored in the package at 'data/example_data.RData'

`sample.data.saccades` *Saccades from 3 individuals*

Description

This dataset contains saccade data from 3 individuals during a free viewing tasks. Data were recorded at 1200 Hz using a Tobii Pro Spectrum eye tracker

Usage

```
sample.data.saccades
```

Format

A data frame

onset onset of the saccade in ms
x.onset gaze position x at onset
y.onset gaze position y at onset
offset offset of the saccade in ms
x.offset gaze position x at offset
y.offset gaze position y at offset
duration duration of saccade in ms
amplitude amplitude of saccade in degrees
peak.velocity peak velocity of saccade
velocity.profile velocity profile
id participant number

Source

The dataset was stored in the package at 'data/example_data.RData'

`sample.data.unprocessed` *Unprocessed sample-by-sample example data*

Description

This dataset contains data from 3 individuals during a free viewing tasks before pre-processing. Data were recorded at 1200 Hz using a Tobii Pro Spectrum eye tracker

Usage

```
sample.data.unprocessed
```

Format

A data frame

id	participant number
timestamp	timestamp in ms recorded by the eye tracker
x.raw	gaze position x
y.raw	gaze position y

Source

The dataset was stored in the package at 'data/example_data.RData'

`static_plot` *Plot fixations in 2D space overlaied on a stimulus image*

Description

This function plots and returns a ggplot2 figure showing fixations on a background with one or multiple images, typically the stimuli. Data can represent one or multiple participants The interval to plot is defined by sample numbers. Fixations must have the variables x, y, and onset. The function is tested with .jpg-images. If paths to multiple images are given, all will be displayed. Fixations are shown on a white background if no background images are defined

Usage

```
static_plot(
  gazedata,
  xres = 1920,
  yres = 1080,
  plot.onset,
  plot.offset,
  background.images = NA,
  show.legend = TRUE,
  group.by = NA,
  gazepoint.size = 4,
  id_color_map = NA,
  connect.lines = TRUE,
  verbose = TRUE
)
```

Arguments

`gazedata`	Data frame with fixation data which must include columns for x and y coordinates as well as the variable onset which indicates the onset of the fixation. If the categorical or factor variable id is included, separate colors will represent each participant. Make sure the onset variables match the timing the plot.onset and plot.offset input.
`xres`	horizontal resolution of the screen or area to plot on. Default 1920
`yres`	vertical resolution of the screen or area to plot on. Default 1080
`plot.onset`	Onset of the interval in the gaze_data$onset variable to plot in the same unit, typically milliseconds
`plot.offset`	Offset of the interval in the gaze_data$onset variable to plot in the same unit, typically milliseconds
`background.images`	data frame with background images to use as background. The data frame must include the variables min.x, min.y, max.x, and max.y variables representing where the images should be placed on the background and the variable path specifying a full file path. #Example: `background.images <- data.frame( path = "my_image.jpg", min.x = 1, min.y = 1, max.x = 200. max.y = 200)`
`show.legend`	If TRUE, show values in "id" in legend
`group.by`	If not NA, plot each level in the variable in a separate panel. For example group.by = "group" returns a separate panel for each group and group.by ="id" returns a separate panel for each id.
`gazepoint.size`	Size of the circle illustrating the point of gaze
`id_color_map`	A ggplot color map specifying a color to plot for each id. ids should match the variable id'in the gazedata matrix. Set to NA to assign values automatically.
`connect.lines`	If TRUE, gaze coordinates are connected with lines
`verbose`	If TRUE, the resulting figure is displayed automatically

Value

a ggplot of raw and fixated values plotted on the y axis and sample number on the x axis

Index