Package 'eyetools'

October 28, 2024

Type Package

Title Analyse Eye Data

Version 0.7.2

Description Enables the automation of actions across the pipeline, including initial steps of transforming binocular data and gap repair to event-based processing such as fixations, saccades, and entry/duration in Areas of Interest (AOIs). It also offers visualisation of eye movement and AOI entries. These tools take relatively raw (trial, time, x, and y form) data and can be used to return fixations, saccades, and AOI entries and time spent in AOIs. As the tools rely on this basic data format, the functions can work with data from any eye tracking device. Implements fixation and saccade detection using methods proposed by Salvucci and Goldberg (2000) <doi:10.1145/355017.355028>.

License GPL-3

URL https://tombeesley.github.io/eyetools/

BugReports https://github.com/tombeesley/eyetools/issues

Depends R (>= 2.10)

Imports ggforce, ggplot2, glue, hdf5r, lifecycle, magick, pbapply, rdist, rlang, stats, utils, zoo

Suggests knitr, rmarkdown, testthat (>= 3.0.0),

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Language en-GB

NeedsCompilation no

Author Beesley Tom [aut, cre], Ivory Matthew [aut]

Maintainer Beesley Tom <t.beesley@lancaster.ac.uk>

Repository CRAN

Date/Publication 2024-10-28 12:00:55 UTC

Contents

`AOI_seq` *Sequence analysis of area of interest entries*

Description

Analyses the sequence of entries into defined AOI regions across trials. Can only be used with fixation data with a "fix_n" column denoting fixation events.

Usage

```
AOI_seq(
  data,
  AOIs,
  AOI_names = NULL,
  sample_rate = NULL,
  long = TRUE,
  participant_ID = "participant_ID"
)
```

Arguments

data	A dataframe with fixation data (from fixation_dispersion). Either single or multi participant data
AOIs	A dataframe of areas of interest (AOIs), with one row per AOI (x, y, width_radius, height).
AOI_names	An optional vector of AOI names to replace the default "AOI_1", "AOI_2", etc.
sample_rate	Optional sample rate of the eye-tracker (Hz) for use with raw_data. If not supplied, the sample rate will be estimated from the time column and the number of samples.
long	Whether to return the AOI fixations in long or wide format. Defaults to long
participant_ID	the variable that determines the participant identifier. If no column present, assumes a single participant

Value

a dataframe containing the sequence of entries into AOIs on each trial.

If long is TRUE, then each AOI entry is returned on a new row, if FALSE, then a row per trial is returned with all AOI entries in one character string

Examples

```
data <- combine_eyes(HCL)
fix_d <- fixation_dispersion(data, participant_ID = "pNum")

AOI_seq(fix_d, AOIs = HCL_AOIs, participant_ID = "pNum")
```

AOI_time *Time analysis of area of interest entries*

Description

Analyses total time on defined AOI regions across trials. Works with fixation and raw data as the input (must use one or the other, not both).

Usage

```
AOI_time(
  data,
  data_type = NULL,
  AOIs,
  AOI_names = NULL,
  sample_rate = NULL,
  as_prop = FALSE,
  trial_time = NULL,
  participant_ID = "participant_ID"
)
```

Arguments

data	A dataframe of either fixation data (from fix_dispersion) or raw data
data_type	Whether data is a fixation ("fix") or raw data ("raw")
AOIs	A dataframe of areas of interest (AOIs), with one row per AOI (x, y, width_radius, height).
AOI_names	An optional vector of AOI names to replace the default "AOI_1", "AOI_2", etc.
sample_rate	Optional sample rate of the eye-tracker (Hz) for use with data. If not supplied, the sample rate will be estimated from the time column and the number of samples.
as_prop	whether to return time in AOI as a proportion of the total time of trial
trial_time	a vector of the time taken in each trial. Equal to the length of x trials by y participants in the dataset
participant_ID	the variable that determines the participant identifier. If no column present, assumes a single participant

Details

AOI_time can take either single participant data or multiple participants where there is a variable for unique participant identification. The function looks for an identifier named `participant_ID` by default and will treat this as multiple-participant data as default, if not it is handled as single participant data, or the participant_ID needs to be specified

Value

a dataframe containing the time on the passed AOIs for each trial. One column for each AOI separated by trial.

Examples

```
data <- combine_eyes(HCL)
fix_d <- fixation_dispersion(data, participant_ID = "pNum")

# fixation data
AOI_time(data = fix_d, data_type = "fix", AOIs = HCL_AOIs, participant_ID = "pNum")

#raw data
AOI_time(data = data, data_type = "raw", AOIs = HCL_AOIs,
         sample_rate = 120, participant_ID = "pNum")
```

`combine_eyes` *Combine binocular data into single X/Y coordinate pairs*

Description

Combines the data from binocular samples into X/Y coordinate pairs. Two methods can be used: "average" or "best_eye". For "average", the result is based on the average of the two eyes for each sample, or for samples where there is data from only a single eye, that eye is used. For "best_eye", a summary of the proportion of missing samples is computed, and the eye with the fewest missing samples is used.

Usage

```
combine_eyes(data, method = "average")
```

Arguments

`data`	raw data with columns time, left_x, left_y, right_x, right_y, and trial
`method`	either "average" or "best_eye" - see description.

Value

a dataframe of x-2 variables (with left_x and right_x condensed to x, and left_y and right_y condensed to y) and the same number of observations as the input data

Examples

```
combine_eyes(HCL, method = "average")
```

`compare_algorithms` *A battery of metrics and plots to compare the two algorithms (dispersion and VTI)*

Description

A tool for comparing the two different algorithms present in this package. This function is useful for assessing the data as well as exploring which algorithm is likely to fit data more appropriately. The raw data is run through both algorithms (using the same specified dispersion tolerances, etc.) before making comparisons of the underlying data. Can only be used for single participant data.

Usage

```
compare_algorithms(
  data,
  plot_fixations = TRUE,
  print_summary = TRUE,
  sample_rate = NULL,
  threshold = 100,
  min_dur = 150,
  min_dur_sac = 20,
  disp_tol = 100,
  NA_tol = 0.25,
  run_interp = TRUE,
  smooth = FALSE
)
```

Arguments

data	A dataframe with raw data (time, x, y, trial) for one participant
plot_fixations	Whether to plot the detected fixations. default as TRUE
print_summary	Whether to print the summary table. default as TRUE
sample_rate	sample rate of the eye-tracker. If default of NULL, then it will be computed from the timestamp data and the number of samples. Supplied to the VTI algorithm
threshold	velocity threshold (degrees of VA / sec) to be used for identifying saccades. Supplied to the VTI algorithm
min_dur	Minimum duration (in milliseconds) of period over which fixations are assessed. Supplied to both algorithms.
min_dur_sac	Minimum duration (in milliseconds) for saccades to be determined. Supplied to the VTI algorithm
disp_tol	Maximum tolerance (in pixels) for the dispersion of values allowed over fixation period. Supplied to both algorithms
NA_tol	the proportion of NAs tolerated within any window of samples that is evaluated as a fixation. Supplied to the dispersion algorithm
run_interp	include a call to eyetools::interpolate on each trial. Supplied to the VTI algorithm
smooth	include a call to eyetools::smoother on each trial. Supplied to the VTI algorithm

Value

a list of the fixation data, correlation output, and data used for plotting

Examples

```
data <- combine_eyes(HCL)
compare_algorithms(data[data$pNum == 118,])
```

conditional_transform *conditional_transform*

Description

A function to perform conditional transformations of the x/y raw data. The function takes the dataframe and performs a single axis flip based on the values specified in the cond_column. The primary use of this function is to correct or normalise the data when counterbalancing stimulus placement within experiments (e.g., having a target stimulus appear on the left and right equally often)

Usage

```
conditional_transform(
  data,
  flip = c("x", "y"),
  cond_column,
  cond_values,
  resolution_x = 1920,
  resolution_y = 1080,
  message = TRUE
)
```

Arguments

data	a dataframe that includes columns x and y and the column specified in cond_column. Can be raw, fixation, or saccade data.
flip	either "x", to flip across vertical midline, or "y" to flip across horizontal midline
cond_column	a column name, on which the flips are conditional
cond_values	a single value or vector stating which values in con_column result in a flip
resolution_x	screen size in pixels for the x axis
resolution_y	screen size in pixels for the y axis
message	whether to output messages during function. Useful to turn off when using in a vectorised fashion where it is running multiple times

Value

a dataframe of the equivalent format as the input data

Examples

```
data <- combine_eyes(HCL)
data <- merge(data, HCL_behavioural)
conditional_transform(data, flip = "x",
                      cond_column = "cue_order",
                      cond_values = 2)
```

`dist_to_visual_angle` *Compute visual angle from distance metrics*

Description

Takes a single value or vector of distances and returns the visual angle equivalent.

Usage

```
dist_to_visual_angle(
  vector,
  dist_type = "cm",
  view_dist_cm = 60,
  screen_width_cm = 51,
  screen_width_pixels = 1920
)
```

Arguments

`vector`	vector of distances (or single distance)
`dist_type`	default is "cm". Specify "pixel" for conversion from pixel values.
`view_dist_cm`	viewing distance in cm. Default of 60cm.
`screen_width_cm`	used in conversion of pixel values. Default is 51 cm (24" monitor).
`screen_width_pixels`	used in conversion of pixel values. Default is 1920 pixels.

Value

an equivalent-sized object to the input

Examples

```
# calculate visual angle for stimulus of 5cm
dist_to_visual_angle(5)

# calculate visual angle of stimuli 2 and 10cm width at 50 cm viewing angle
dist_to_visual_angle(c(2,10), view_dist_cm = 50)

# calculate visual angle of 150 pixel wide
dist_to_visual_angle(150, dist_type = "pixels")
```

`fixation_dispersion` *Fixation detection using a dispersion method*

Description

Detects fixations by assessing dispersion of the eye position, using a method that is similar to that proposed by Salvucci & Goldberg (1996). Evaluates the maximum dispersion (distance) between x/y coordinates across a window of data. Looks for sufficient periods in which this maximum dispersion is below the specified dispersion tolerance. NAs are considered breaks in the data and are not permitted within a valid fixation period. Runs the interpolation algorithm by default to fix small breaks in the data.

Usage

```
fixation_dispersion(
  data,
  min_dur = 150,
  disp_tol = 100,
  run_interp = TRUE,
  NA_tol = 0.25,
  progress = TRUE,
  participant_ID = "participant_ID"
)
```

Arguments

`data`	A dataframe with raw data (time, x, y, trial) for one participant (the standardised raw data form for eyetools)
`min_dur`	Minimum duration (in milliseconds) of period over which fixations are assessed
`disp_tol`	Maximum tolerance (in pixels) for the dispersion of values allowed over fixation period
`run_interp`	include a call to eyetools::interpolate on each trial
`NA_tol`	the proportion of NAs tolerated within any window of samples that is evaluated as a fixation
`progress`	Display a progress bar
`participant_ID`	the variable that determines the participant identifier. If no column present, assumes a single participant

Details

It can take either single participant data or multiple participants where there is a variable for unique participant identification. The function looks for an identifier named `participant_ID` by default and will treat this as multiple-participant data as default, if not it is handled as single participant data, or the participant_ID needs to be specified

Value

a dataframe containing each detected fixation by trial, with mean x/y position in pixel, start and end times, and duration.

References

Salvucci, D. D., & Goldberg, J. H. (2000). Identifying fixations and saccades in eye-tracking protocols. Proceedings of the Symposium on Eye Tracking Research & Applications - ETRA '00, 71–78.

Examples

```
data <- combine_eyes(HCL)
fixation_dispersion(data, participant_ID = "pNum")
```

---

fixation_VTI *Fixation detection using a velocity threshold identification method*

Description

Determine fixations by assessing the velocity of eye-movements, using a method that is similar to that proposed by Salvucci & Goldberg (1996). Applies the algorithm used in VTI_saccade and removes the identified saccades before assessing whether separated fixations are outside of the dispersion tolerance. If they are outside of this tolerance, the fixation is treated as a new fixation regardless of the length of saccade separating them. Compared to fixation_dispersion(), fixation_VTI() is more conservative in determining a fixation as smaller saccades are discounted and the resulting data is treated as a continued fixation (assuming it is within the pixel tolerance set by disp_tol). Returns a summary of the fixations found per trial, including start and end coordinates, timing, duration, mean velocity, and peak velocity.

Usage

```
fixation_VTI(
  data,
  sample_rate = NULL,
  threshold = 100,
  min_dur = 150,
  min_dur_sac = 20,
  disp_tol = 100,
  run_interp = TRUE,
  smooth = FALSE,
  progress = TRUE,
  participant_ID = "participant_ID"
)
```

Arguments

data	A dataframe with raw data (time, x, y, trial) for one participant
sample_rate	sample rate of the eye-tracker. If default of NULL, then it will be computed from the timestamp data and the number of samples
threshold	velocity threshold (degrees of VA / sec) to be used for identifying saccades.
min_dur	Minimum duration (in milliseconds) of period over which fixations are assessed
min_dur_sac	Minimum duration (in milliseconds) for saccades to be determined
disp_tol	Maximum tolerance (in pixels) for the dispersion of values allowed over fixation period
run_interp	include a call to eyetools::interpolate on each trial.
smooth	include a call to eyetools::smoother on each trial
progress	Display a progress bar
participant_ID	the variable that determines the participant identifier. If no column present, assumes a single participant

Details

It can take either single participant data or multiple participants where there is a variable for unique participant identification. The function looks for an identifier named `participant_ID` by default and will treat this as multiple-participant data as default, if not it is handled as single participant data, or the participant_ID needs to be specified

Value

a dataframe containing each detected fixation by trial, with mean x/y position in pixel, start and end times, and duration.

References

Salvucci, D. D., & Goldberg, J. H. (2000). Identifying fixations and saccades in eye-tracking protocols. Proceedings of the Symposium on Eye Tracking Research & Applications - ETRA '00, 71–78.

Examples

```
data <- combine_eyes(HCL)
fixation_VTI(data, participant_ID = "pNum")
```

HCL *Example dataset from that contains binocular eye data from two participants from a simple contingency learning task (the data are from Beesley, Nguyen, Pearson, & Le Pelley, 2015). In this task there are two stimuli that appear simultaneously on each trial (to the left and right of the screen). Participants look at these cues and then make a decision by selecting an "outcome response" button.*

Description

The dataset contains data from two participants and the first six trials of the study.

Usage

```
HCL
```

Format

A dataframe of 31,041 observations and seven variables

pNum participant number

time timestamp of the sample (milliseconds)

left_x x coordinate of the left eye

left_y y coordinate of the left eye

right_x x coordinate of the right eye

right_y y coordinate of the right eye

trial trial number ...

HCL_AOIs *Example AOIs for use with HCL*

Description

This dataframe contains three rectangular areas of interest (AOIs), set out for use with the HCL dataset. Values are in pixels.

Usage

```
HCL_AOIs
```

Format

A data frame with 3 rows and 4 variables:

x centred x coordinate of the AOI

y centred y coordinate of the AOI

width_radius either the width of the AOI, or the radius for circular AOIs

height the height of the AOI; should be NA for circular AOIs ...

HCL_behavioural *Example dataset of behavioural data to complement dataset HCL.*

Description

This contains information on stimuli (such as the side the predictive cue was presented on) as well as response data, including accuracy and response times

Usage

```
HCL_behavioural
```

Format

A dataframe of 12 observations and eight variables

pNum participant number

trial trial number

P_cue Are these necessary columns?

NP_cue Are these necessary columns?

cue_order whether the predictive cue os presented on the left (1) or the right (2)

correct_out NAre these necessary columns?

accuracy response accuracy

RT response time in milliseconds ...

hdf5_to_csv *hdf5_to_csv*

Description

A function to convert TOBII-generated hdf5 files to csv

Usage

```
hdf5_to_csv(filename)
```

Arguments

filename	the hdf5 file generated from TOBII

Value

A list of csv files collected from the eyetracker content, if only one eyetracking event is present, return this as a csv file

Examples

```
## Not run:
raw_data <- hdf5_to_csv("example_TOBII.hdf5")

## End(Not run)
```

interpolate *Interpolation of missing data (NAs)*

Description

Extends the zoo::na.approx and zoo::na.spline functions to include a report which provides the proportion of missing data before and after the interpolation process. This is handy for evaluating the effectiveness of the repair.

Usage

```
interpolate(
  data,
  maxgap = 25,
  method = "approx",
  report = FALSE,
  participant_ID = "participant_ID"
)
```

Arguments

data	dataframe with columns time, x, y, trial (the standardised raw data form for eyeproc)
maxgap	maximum number of consecutive NAs to fill. Any longer gaps will be left unchanged (see zoo package)
method	"approx" for linear interpolation or "spline" for cubic spline interpolation
report	default is FALSE. If TRUE, then the return value is a list containing the returned data frame and the report.
participant_ID	the variable that determines the participant identifier. If no column present, assumes a single participant

Details

It can take either single participant data or multiple participants where there is a variable for unique participant identification. The function looks for an identifier named `participant_ID` by default and will treat this as multiple-participant data as default, if not it is handled as single participant data, or the participant_ID needs to be specified

Value

a dataframe of the same shape of the input data

Examples

```
data <- combine_eyes(HCL)
interpolate(data, maxgap = 20, participant_ID = "pNum")
```

---

plot_seq *Plot of raw data over time*

Description

A tool for visualising the timecourse of raw data over a single trial. If data from multiple trials are present, then a single trial will be sampled at random. Alternatively, the trial_number can be specified. Data can be plotted across the whole trial, or can be split into bins to present distinct plots for each time window.

Usage

```
plot_seq(
  data = NULL,
  trial_number = NULL,
  AOIs = NULL,
  bg_image = NULL,
  res = c(0, 1920, 0, 1080),
```

```
  flip_y = FALSE,
  plot_header = FALSE,
  bin_time = NULL,
  bin_range = NULL
)
```

Arguments

data	A dataframe with raw data. If multiple trials are used, then one trial is sampled at random.
trial_number	can be used to select a particular trial within the data
AOIs	A dataframe of areas of interest (AOIs), with one row per AOI (x, y, width_radius, height).
bg_image	The filepath of an image to be added to the plot, for example to show a screenshot of the task.
res	resolution of the display to be shown, as a vector (xmin, xmax, ymin, ymax)
flip_y	reverse the y axis coordinates (useful if origin is top of the screen)
plot_header	display the header title text which explains graphical features of the plot.
bin_time	if wanting to split data into bins, the time (in ms) for each bin of data to be displayed
bin_range	if wanting to split data into bins, the first and last bin to be display, e.g., c(1,5)

Value

a plot of the raw data representing changes over time

Examples

```
data <- combine_eyes(HCL)

# plot the raw data
plot_seq(data = data[data$pNum == 118,])

# with AOIs
plot_seq(data = data[data$pNum == 118,], AOIs = HCL_AOIs)

# plot raw data with bins
plot_seq(data = data[data$pNum == 118,], bin_time = 500)
```

plot_spatial *Plot raw data and fixations*

Description

A tool for visualising raw eye-data, processed fixations, and saccades. Can use all three data types together and independently. Fixations can be labeled in the order they were made. Can overlay areas of interest (AOIs) and customise the resolution.

Usage

```
plot_spatial(
  raw_data = NULL,
  fix_data = NULL,
  sac_data = NULL,
  AOIs = NULL,
  trial_number = NULL,
  bg_image = NULL,
  res = c(0, 1920, 0, 1080),
  flip_y = FALSE,
  show_fix_order = TRUE,
  plot_header = FALSE
)
```

Arguments

raw_data	data in standard raw data form (time, x, y, trial)
fix_data	data output from fixation function
sac_data	data output from saccade function
AOIs	A dataframe of areas of interest (AOIs), with one row per AOI (x, y, width_radius, height). If using circular AOIs, then the 3rd column is used for the radius and the height should be set to NA.
trial_number	can be used to select particular trials within the data
bg_image	The filepath of an image to be added to the plot, for example to show a screen-shot of the task.
res	resolution of the display to be shown, as a vector (xmin, xmax, ymin, ymax)
flip_y	reverse the y axis coordinates (useful if origin is top of the screen)
show_fix_order	label the fixations in the order they were made
plot_header	display the header title text which explains graphical features of the plot.

Value

a plot of the raw data

Examples

```
data <- combine_eyes(HCL)
data <- data[data$pNum == 118,]
# plot the raw data
plot_spatial(raw_data = data[data$pNum == 118,])

# plot both raw and fixation data together
plot_spatial(raw_data = data, fix_data = fixation_dispersion(data))

#plot one trial
plot_spatial(raw_data = data, fix_data = fixation_dispersion(data), trial_number = 1)
```

---

saccade_VTI *Velocity threshold identification of saccades*

Description

Use the velocity threshold algorithm from Salvucci & Goldberg (1996) to determine saccadic eye movements. Returns a summary of the saccades found per trial, including start and end coordinates, timing, duration, mean velocity, and peak velocity.

Usage

```
saccade_VTI(
  data,
  sample_rate = NULL,
  threshold = 150,
  min_dur = 20,
  participant_ID = "participant_ID"
)
```

Arguments

data	A dataframe with raw data (time, x, y, trial) for one participant
sample_rate	sample rate of the eye-tracker. If default of NULL, then it will be computed from the timestamp data and the number of samples
threshold	velocity threshold (degrees of VA / sec) to be used for identifying saccades
min_dur	minimum duration (ms) expected for saccades. This helps to avoid identification of very short saccades occurring at the boundary of velocity threshold
participant_ID	the variable that determines the participant identifier. If no column present, assumes a single participant

Details

It can take either single participant data or multiple participants where there is a variable for unique participant identification. The function looks for an identifier named `participant_ID` by default and will treat this as multiple-participant data as default, if not it is handled as single participant data, or the participant_ID needs to be specified

Value

a data frame giving the saccades found by trial

Examples

```
data <- combine_eyes(HCL)
saccade_VTI(data, participant_ID = "pNum")
```

`smoother` *Smoothing of raw data*

Description

A wrapper for the stats::loess function, with default parameters suitable for smoothing raw eye data

Usage

```
smoother(data, span = 0.1, plot = FALSE, participant_ID = "participant_ID")
```

Arguments

`data`	A dataframe with raw data (time, x, y, trial) for one participant
`span`	From stats::loess. The parameter alpha which controls the degree of smoothing.
`plot`	whether to plot the raw and smoothed plot for inspection
`participant_ID`	the variable that determines the participant identifier. If no column present, assumes a single participant

Details

It can take either single participant data or multiple participants where there is a variable for unique participant identification. The function looks for an identifier named `participant_ID` by default and will treat this as multiple-participant data as default, if not it is handled as single participant data, or the participant_ID needs to be specified

Value

a dataframe of the same shape as the input data

Examples

```
data <- combine_eyes(HCL)

smoother(data, participant_ID = "pNum")

#with an inspection plot
smoother(data, span = .02, participant_ID = "pNum", plot = TRUE)
```

Index