Package 'eyelinker'

July 22, 2025

Type Package

Title Import ASC Files from EyeLink Eye Trackers

Version 0.2.1

Author Simon Barthelme <simon.barthelme@gipsa-lab.fr>

Maintainer Austin Hurst <ajhurst@uwaterloo.ca>

Description Imports plain-text ASC data files from EyeLink eye trackers into (relatively) tidy data frames for analysis and visualization.

License GPL-3 | file LICENCE

URL https://github.com/a-hurst/eyelinker

BugReports https://github.com/a-hurst/eyelinker/issues

Depends R (>= 3.2)

Imports stringi, stringr, tibble, readr, intervals

Suggests testthat, knitr, rmarkdown, dplyr, ggplot2, tidyr, covr

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2021-06-03 17:20:02 UTC

Contents

eyelinker *Import ASC Files from EyeLink Eye Trackers*

Description

Dealing with unprocessed ASC files from EyeLink eye trackers can be a pain. This package aims to make importing and working with these files as fast and easy as possible.

Details

For documentation of the structure of the returned data, see the "format" vignette:

```
vignette("format", package = "eyelinker")
```

For worked examples illustrating the package in action, see the "basics" vignette:

```
vignette("basics", package = "eyelinker")
```

read.asc *Read EyeLink ASC Files*

Description

Imports data from EyeLink ASC files into (relatively) tidy data frames for analysis and visualization. Event data and/or raw sample data from the files can be imported, along with information about the tracker hardware and configuration. All data is divided into numbered blocks using the "START" and "END" messages in the ASC file.

Usage

```
read.asc(fname, samples = TRUE, events = TRUE, parse_all = FALSE)

read_asc(fname, samples = TRUE, events = TRUE, parse_all = FALSE)
```

Arguments

fname	character vector indicating the name of the .asc file to import.
samples	logical indicating whether raw sample data should be imported. Defaults to TRUE.
events	logical indicating whether event data (e.g. saccades, blinks, messages, etc.) should be imported. Defaults to TRUE.
parse_all	logical indicating whether samples/events not within START/END blocks should be parsed. Defaults to FALSE.

Details

ASC files can contain anywhere between 125 to 2000 rows of samples for every second of recording, meaning that the resulting files can be very large (1.2 million rows of samples for 20 minutes at 1000Hz). As a result, importing some ASC files can be slow, and the resulting data frames can take up 100's of MB of memory. To speed up import and greatly reduce memory load, you can choose to ignore raw samples and only import events by setting the samples parameter to `FALSE`.

This function returns a list containing the following possible data frames:

`raw` Raw sample data

`sacc` Saccade end events

`fix` Fixation end events

`blinks` Blink end events

`msg` Messages sent or received by the tracker

`input` Input port (TTL) events

`button` Button box / gamepad events

`info` Tracker settings/configuration metadata

The names of the columns in these data frames correspond to column names given in the ASC section of the EyeLink 1000 User's Guide.

Note that this function cannot import EDFs directly; they must be converted to plain-text ASC using the edf2asc utility before importing.

Value

A `list` of `tibble`s containing data from the .asc file.

Author(s)

Simon Barthelme & Austin Hurst

Examples

```
# Example file from SR research that ships with the package
fpath <- system.file("extdata/mono500.asc.gz", package = "eyelinker")
dat <- read.asc(fpath)
plot(dat$raw$time, dat$raw$xp, xlab = "Time (ms)", ylab = "Eye position along x-axis (pix)")
```

`whichInterval` *From a set of intervals, find which interval values belong to*

Description

Returns which interval (if any) each number in a vector belongs to, given a set of user-defined intervals. Intervals can be specified using either two-column matrices or `Intervals` objects from the `intervals` package.

Usage

```
whichInterval(x, Intv)

which_interval(x, Intv)
```

Arguments

`x`	A vector of numeric values
`Intv`	A two-column matrix or an object of class Intervals

Value

For each value in x: if x[i] is in the set of intervals, the index of the corresponding interval(s), NA if no interval contains x[i]

Author(s)

Simon Barthelme

See Also

%In%

Examples

```
start <- c(0, 1, 2)
end <- c(.5, 1.3, 3)
intv <- cbind(start, end) # The first interval is 0-0.5, second is 1-1.3, etc.
whichInterval(seq(0, 3, l = 10), intv)
```

`%In%` *Find if value belongs to a set of intervals*

Description

Returns whether numeric values on the left-hand side of the operator fall within any of the specified intervals on the right-hand side. Intervals can be specified using either two-column matrices or `Intervals` objects from the `intervals` package.

Usage

```
x %In% Intv

x %within% Intv
```

Arguments

`x`	A vector of numeric values
`Intv`	A set of intervals, defined by a two-column matrix of endpoints or an Intervals object

Value

A vector of logicals, which are true if x[i] belongs to any of the intervals in the set.

Author(s)

Simon Barthelme

Examples

```
start <- c(0, 1, 2)
end <- c(.5, 1.3, 3)
intv <- cbind(start, end) # The first interval is 0-0.5, second is 1-1.3, etc.
c(0, .6, 1.5, 3) %In% intv
```

Index