Matlab interface to Eyelink EDF files.
Version 1.0.1

Alexander Pastukhov

December 17, 2015

1 What is it for?

This library provides a simple interface to import contents of the EDF files generated by Eyelink eye-tracker into Matlab (if you don't know what are EDF files generated by Eyelink - you don't need this library). It imports events and/or samples, automatically parsing them into separate trials. In addition to that several post-processing functions can be used to extract selected events (fixations, saccades and blinks), variable values (TRIAL_VAR events) and microsaccades.

Library and updates can be downloaded from our web-site homepage at
http://kobi. nat.uni-magdeburg.de/edfImport.

If you find any bugs, please email me at Pastukhov.Alexander@gmail.com.

2 Prerequisite

This library relies on EDF API provided by SR-Research on their support web-site (www.sr-support.com). This library was tested to work with version 2.5 of EDF API.

For details on structures and fields please consult EDF API manual. Note, that there are some discrepancies between the manual and the actual library implementation. For example in the FSAMPLE structure the manual lists *status* field, which is, however, not defined in the library itself.

Another important note: although MATLAB creates any value as double by default, it is very stringent on how it handles the types when they are defined. So, for example, if you will subtract time fields you will never get a negative value, as they are defined as *unsigned*. To override it just use *double(variable)* to convert it to double.

3 Known limitations

While *edfExtractVariables* function can automatically extract variable values, it assumes that they were recorded using *TRIAL_VAR* message. Combination of *TRIAL_VAR_LABELS* and *TRIALS_VAR_DATA* currently is not supported.

4 Version changes

- Version 1.0.1 added performance improvement (removed the bug) and a progress bar, to make things look neat.
- Version 1.0.2 added truncation of samples to exclude empty elements in Samples fields.
- Version 1.0.3-1.0.5 bug fixes

5 Compilation

Precompiled mex files (Windows and Mac OS X, 32 bit Matlab only) are included but you can compile them yourself by running *edfCompile.m* script. Please check paths to include files and libraries. Default locations are as follows.

- For Windows (32 bit):
  - includes: `C:\Program Files\SR Research\EyeLink\EDF_Access_API\Example`
  - library: `C:\Program Files\SR Research\EyeLink\EDF_Access_API\lib\win32`
- For Windows (64 bit):
  - includes: `C:\Program Files (x86)\SR Research\EyeLink\EDF_Access_API\Example`
  - library: `C:\Program Files (x86)\SR Research\EyeLink\EDF_Access_API\lib\win32`
- For Mac OS X:
  - includes: `/Library/Frameworks/edfapi.framework/Headers/`
  - library: `/Library/Frameworks/edfapi.framework/`

I have never tested it on Linux, but compilation should be straightforward. Contact me if you need help with it.

6 How to use

To import data from the EDF file call *edfImport* function, e.g.:

```
Trials= edfImport('test.edf', [1 1 1], '');
```

This will import all events and samples into vector Trials structures (see section 8 for details). Next you can call *edfExtractInterestingEvents* to extract fixations, saccades, blinks and button presses recorded by the eye tracker; *edfExtractMicrosaccades* to extract microsaccades based on raw eyes positions; *edfExtractVariable* for variables and values stored during recording.

7 Functions list

7.1 edfImport

Imports events and/or samples from the EDF file.

Syntax

```
Trials= edfImport(Filename, Options, SampleFields,
    TrimSamplesToCorrectNumber)
Trials= edfImport(Filename, Options, SampleFields)
Trials= edfImport(Filename, Options)
Trials= edfImport(Filename)
[Trials, Preamble]= edfImport(...)
```

Description

Trials= edfImport(Filename, Options, SampleFields, TrimSamplesToCorrectNumber) imports events and/or samples from the file *Filename*. *Options* argument is a vector with following flags *[consistency load_events load_samples]*. Where

- consistency
  - 0: no consistency check
  - 1: check consistency and report (default)
  - 2: check consistency and fix
- load_events
  - 0: do not load events
  - 1: load events (default)
- load_samples
  - 0: do not load samples (default)
  - 1: load samples

SampleFields is a space-separated list of *FSAMPLE* structure fields to be imported. If *load_samples* is 0 this argument is ignored. To import all fields omit *SampleField* argument or pass an empty string.

TrimSamplesToCorrectNumber (default: true): truncates samples to the real number of imported samples. Arrays may be longer as they are pre-allocated by computing number of samples based on recording duration and sampling frequency. Typically, you may get one (last) empty sample.

Trials= edfImport(Filename, Options) if *load_samples* is 1 - imports all *FSAMPLE* fields.

Trials= edfImport(Filename) uses default *Options*= [1 1 0].

[Trials, Preamble]= edfImport(...) additionally imports *Preamble* of the EDF file (see EDF API manual for details).

Examples

Importing events only:

```
Trials= edfImport('test.edf');
```

Importing everything:

```
Trials= edfImport('test.edf', [1 1 1]);
```

Importing selected *FSAMPLE* fields

```
Trials= edfImport('test.edf', [1 1 1], 'time gx gy rx ry');
```

7.2 edfExtractInterestingEvents

Extract fixations, saccades, blinks and button presses from events and places them into new fields within *Trial* structure.

Syntax

```
Trials= edfExtractInterestingEvents(Trials)
```

Description *Trials= edfExtractInterestingEvents(Trials)* extact fixations, saccades and blinks from events and places them, respectively, into *Fixations*, *Saccades* and *Blinks* field in the updated *Trials* structure. New substructures contain the following fields (see details in EDF API manual):

- Fixations: eye, sttime, entime, time, gavx, gavy, PixInDegX, PixInDegY
- Saccades: eye, sttime, entime, time, gstx, gsty, genx, geny, avel, pvel, ampl, phi, PixInDegX, PixInDegY
- Blinks: eye, sttime, entime, time
- Buttons: ID, Pressed, time

7.3 edfExtractMicrosaccades

Extracts microsaccades from the raw eye positions data using algorithm described in Engbert & Kliegl (2003). If you use this function, please cite *Engbert, R. and R. Kliegl (2003). "Microsaccades uncover the orientation of covert attention." Vision Res 43(9): 1035-45.*

Syntax

```
Trials= edfExtractMicrosaccadesForTrial(Trials, VelocityBraketMS,
    VelocityThreshold, MinimalDurationMS, MinimalSeparationMS)
Trials= edfExtractMicrosaccadesForTrial(Trials)
```

Description

Extracts microsaccades using raw eye positions. This requires that *Samples* field exists (i.e. samples were imported) and the following fields are present: *time*, *gx*, *gy*. Refresh rate of the camera for recording is taken from *Trials().Header.rec.sample_rate* field. Additional options and their default values (used if the option is omitted), for more details consult Engbert & Kliegl (2003).

- VelocityBraketMS: time span around current sample with which to compute the velocity in milliseconds. *Default: 20 ms.*
- VelocityThreshold: velocity threshold for microsaccade detection in medians. *Default: 6.*
- MinimalDurationMS: minimal microsaccade duration in milliseconds. *Default: 12 ms.*
- MinimalSeparationMS: minimal time in milliseconds allowed between two microsaccades, otherwise they are merged. *Default: 12 ms.*

For information on the appended *Microsaccades* structure see 8.

7.4 edfExtractVariables

Extracts variables and their values from *TRIAL_VAR* messages. Note *TRIAL_VAR_LABELS* and *TRIAL_VAR_DATA* are currently not supported.

Syntax

```
Trials= edfExtractVariables(Trials)
```

Description

Extracts variables and their values from *TRIAL_VAR* messages (in *Events* structure) and stores them into a new field *Variables* with each variable being a new subfield. During recording variables can be saved using either space-separated format *'TRIAL_VAR VarName VarValue'* or using a MATLAB-compatible syntax *'TRIAL_VAR VarName=VarValue'*. In the latter case you can use any MATLAB-legal expression for *VarValue*.

7.5 edfExtractKeyEventsTiming

Extracts time of the events recorded with *KEY_EVENT* message.

Syntax

```
Trials= edfExtractKeyEventsTiming(Trials)
```

Description

This is a non-standard function. In order to have precise timing of experimental events in eye-tracker time units, I record them using *eyemsg_printf("KEY_EVENT SomeImportantEvent")*. This way I know exactly when it happened and later use current function to extract the timing. Information is stored in the new field *KeyEvents* with each event name being a new subfield and timing in eye tracker time units stored as a value. For example "KEY_EVENT MaskOnset" becomes *Trials(iT).KeyEvents.MaskOnset= T;* (there T is the time of the event). Note, you can use event codes like *"KEY_EVENT FrameOnset(1)"* and *"KEY_EVENT FrameOnset(2)"*, in this case you will get a two element vector *KeyEvents.FrameOnset= [T1 T2]*.

7.6 edfCheckFixationStability

Verifies that all fixations happen within a safe radius around the fixation.

Syntax

```
Trials= edfCheckFixationStability(Trials, Fixation, ValidRadius)
```

Description

Checks fixations for each trial to verify that they are within a safe radius away from the fixation. *Fixation* is a 2 element vector for the fixation position on the screen **in pixels**: [x, y]. *ValidRadius* - maximum tolerable deviation from fixation **in degress of visual angle**. Each trial is appended with a new *Valid* field, which is 1 if trial is valid and 0, if some fixations fell outside of the safe radius.

8 Trial structure

Each trial structure could contain following fields/substructures. Which fields are available depends on what kind of processing was done.

- Header: this is a *TRIAL* structure, for details see EDF API manual.
- Events: this is a *FEVENT* structure, for details see EDF API manual.
- Samples: this is a *FSAMPLE* structure with an additional field *RealNumberOfSamples*, for details see EDF API manual. Note, only fields which you chose to import, when calling *edfImport*, will be present plus *RealNumberOfSamples*. Latter contains real number of imported samples and is used by *edfImport* function to truncate fields to eliminate empty samples (see *edfImport* for details).
- StartTime: time of the first sample recording (the trial itself starts earlier). Added by *edfExtractInterestingEvents* function.
- Fixations: information about all fixations, as detected by the eye tracker. Fields indclude: eye, sttime, entime, time, gavx, gavy, PixInDegX, PixInDegY. Added by *edfExtractInterestingEvents* function, see 7.2.
- Saccades: information about all saccades, as detected by the eye tracker. Fields indclude: eye, sttime, entime, time, gstx, gsty, genx, geny, avel, pvel, ampl, phi, PixInDegX, PixInDegY. Added by *edfExtractInterestingEvents* function, see 7.2.
- Blinks: information about all blinks, as detected by the eye tracker. Fields include: eye, sttime, entime, time. Added by *edfExtractInterestingEvents* function, see 7.2.
- Microsaccades: information about extracted microsaccades. Subfields:
  - Start: index of the microsaccade start in Samples.
  - End: index of the microsaccade end in Samples.
  - StartTune: start of the microsaccade in eye tracker time.
  - EndTime: end of the microsaccade in eye tracker time.
  - DeltaX: horizontal component.

- DeltaY: vertical component.
- vPeak: peak velocity.
- Amplitude: amplitude (traveled distance).
- Phi: direction in degrees (not radians).
- Duration: in milliseconds.

Added by *edfExtractMicrosaccades*, see 7.3.

- Variables: contains all variables and values stored during recording with *TRIAL_VAR* eye tracker command. Added by *edfExtractVariables* function, see 7.4.
- KeyEvents: contains names (as fields) and time (as values) of key events recorded with a custom *KEY_EVENT* message. Added by *edfExtractKeyEventsTiming*, see 7.5.
- Valid: generated by *edfCheckFixationStability* (see 7.6), 1 if trial is valid, 0 if some fixation occured outside of the safe radius.

9 Service functions

9.1 edfMexImport

Imports information from EDF file. Don't call it directly, please use wrapper *edfImport* which has safety checks for passed arguments.

9.2 edfSelectSampleFields

Converts list of space-separated names of *FSAMPLE* structure fields into a boolean array of flags, later passed to *edfMexImport* function. Used by *edfImport*.

9.3 edfCompile

OS-sensitive script for library compilation. Modify it to suite your environment.

9.4 edfFindTrialRecordingStart

Locates beginning of samples recording, used by *edfExtractMicrosaccades*.