INDEX OF THIS DOC

- WHAT IS T2T?
- WRAPPED BY SWIG
- SUPPORTED PLATFORMS
- NOTES ON THE INSTALLATION
- COMMANDS DOCUMENTATION
- T2T VERSIONS HISTORY

WHAT IS T2T?

T2T is a small package which provides Tobii Eye Trackers (www.tobii.com) communication support in OS X and Linux. It's based on talk2tobii (which is based on libtobii). Although talk2tobii has been developed as Matlab Toolbox, t2t is completely independent from Matlab. T2T also brings talk2tobii to Linux platforms as Matlab and Octave Toolbox. T2T permits any C/C++/Python/Perl applications to interface with a Tobii Eye Tracker.

To know more about talk2tobii visit:
http://www.cbcd.bbk.ac.uk/people/affiliated/fani/talk2tobii

To know more about libtobii visit:
http://andrewd.ces.clemson.edu/tobii/

T2T is composed of:

- libtobii*: a modified version of the original Duchowski TET library port (not open source)
- talk2tobii*: a modified version of the Fani Deligianni's original talk2tobii module
- libt2t: a library which makes of talk2tobii (Matlab executable) a C/C++ linkable library
- t2tio: a sample terminal application which lets you send/receive data to/from a Tobii machine
- a Perl wrapper to libt2t: provides access to libt2t from Perl
- a Python wrapper to libt2t : provides access to libt2t from Python
- some Python and Perl sample code that shows how to use the library:
    + followEyes: tracks the eyes movement of the subject and reproduces them on the screen
    + calibration: performs a tobii calibration displaying a set of pictures on the screen
    + cmdTest: performs a t2t test test running several commands
- a built mex extension of talk2tobii* for Matlab and Octave

Package content:

t2t/doc	-- documentation
t2t/bin	-- binary executables
t2t/examples	-- python and perl examples
t2t/inc	-- header files for C/C++ applications
t2t/lib	-- binary libraries

WRAPPED BY SWIG

Thanks to SWIG (http://www.swig.org/) your Perl and Python applications can benefit of libt2t.

SUPPORTED PLATFORMS

- OS X
- Linux

NOTES ON THE INSTALLATION

Remember to set properly your DYLD_LIBRARY_PATH if you make use of the dynamic libraries. If you use python or perl, do tell to the interpreter where to find the necessary modules. For Python you need to set the environment var PYTHONPATH, whereas for perl set PERL5LIB.

Example (for Mac OS X):

```
#> cd t2t
#> make
#> export DYLD_LIBRARY_PATH=$DYLD_LIBRARY_PATH:`pwd`/lib:`pwd`/python
#> export DYLD_LIBRARY_PATH= $DYLD_LIBRARY_PATH: `pwd`/lib/perl5/darwin-2level/auto/t2tsw
#> export PYTHONPATH=$PYTHONPATH:`pwd`/lib/python
#> export PERL5LIB=$PERL5LIB:`pwd`/lib/perl5/darwin-2level:`pwd`/examples/perl
```

For Linux do take care of changing DYLD_LIBRARY_PATH with LD_LIBRARY_PATH and to refer properly to the corrisponding path.

There is a utility script which suggests automatically how to do: utils/envset.sh. Just run it.

The examples

To run the sample code in the examples dir you need to install some modules for Perl and for Python (if do not have them already installed)

For Perl:

    Time::HiRes
    Term::ReadKey
    Tk::CursorControl
    Tk::JPEG
    Tk::PNG
    Tk

Have a look at

    http://www.cpan.org/misc/cpan-faq.html#How_install_Perl_modules
    http://www.cpan.org/modules/INSTALL.html

to know how to install a perl module

For Python:

    argparse
    pygame
    turtle

Have a look at

    http://docs.python.org/install/

to know how to install a python module.

COMMANDS DOCUMENTATION

As basic reference the original talk2tobii manual Matlab and Octave is still valid in general, but there are some noticeable differences for the following commands: START_CALIBRATION, SAVE_DATA, GET_SAMPLE_EXT, GET_SAMPLE, CLEAR_DATA, GET_STATUS, EVENT. Also there are 5 new commands GET_GAZES_DATA and GET_EVENTS_DATA, START_AUTO_SYNC, STOP_AUTO_SYNC, REMOVE_CALIBRATION_SAMPLES, TIMESTAMP (void command). You can find the new specifications in talk2tobii.m, but briefly:

1. The START_CALIBRATION has been changed quite a lot, offering a new set of parameters to perform an optimal calibration, both from the point of view of the procedure and the results. It also provides the possibility to load a previously saved calibration. Basically, the calibration process implies to define a sequence of calibration points, the number of samples the TET should collect per each one of these points and, optionally, the name of the file where to save the result of the calibration. Another mandatory parameter, is the special boolean flag "clear calibration", which should be set to true to clear a previous calibration and start the new one from scratch; on the other hand, when it's false, the calibration will add the newly collected samples to the set gathered from the previous calibration (either loaded or performed): this is useful when, after the analysis of the calibration results, one might decide to repeat the calibration only for some of the original calibration points, or simply to add new calibration points. In repeating the calibration, START_CALIBRATION should be used in combination with REMOVE_CALIBRATION_SAMPLES. Note that, removing samples does not involve recalculating and setting the calibration: that's performed issuing again a START_CALIBRATION command.
2. The GET_SAMPLE_EXT and GET_SAMPLE commands return an array with an extra element that is the remote TET timestamp converted into local timestamp.
3. The SAVE_DATA command, if the events file name and the gaze data file name are the same, now returns the 2 data types, events & gazes, combined on a single line and printed on the same file ordered by timestamps (all in milliseconds).
4. The GET_STATUS command returns extra information bits, appended at the end of the status array: the 13th values (index starting from 1) reflects the status of the last auto-synchronization operation performed by the background autosync thread (this bit is to check if the autosync procedure work properly); the 14th value is 1 if there's a calibration samples removal ongoing; finally, the 15th value is 1 after a calibration point has been successfully added by the TET (that requires a variable amount of time since and ADD_CALIBRATION_POINT command has been issued, depending on the number of samples per point to collect), and is reset to 0 when the user issues a DREW_POINT command.
5. The EVENT command takes no longer a start_time parameter: this value is set automatically by talk2tobii at the moment at which the command is processed, using the internal local timer (*) and it's returned as output parameter.
6. CLEAR_DATA has 2 optional parameters which permit to delete partially the data collected, up to a specific index. This, in combination with SAVE_DATA, permits to perform an incremental saving.
7. The TIMESTAMP command returns a talk2tobii local timestamp: to issue such a command you do not have actually to specify its name; this command requires passing no parameters to talk2tobii, just call: talk2tobii(). The value returned is expressed in seconds.

8. GET_GAZES_DATA and GET_EVENTS_DATA commands return respectively the gazes data and the events data collected since a specific sample index/event index.
9. START_AUTO_SYNC/STOP_AUTO_SYNC take advantage of the UDP based synchronization service supported only since this talk2tobii version (previously only synchronization over TCP was implemented): the first command starts a background thread which takes care to keep the local clock & the remote TET server clock synchronized; the second command stop the thread. You can any time issue also a SYNCHRONIZE command: will still work and will use the new UDP service if available.
10. REMOVE_CALIBRATION_SAMPLES has been added to adjust an ongoing calibration. This command removes the samples points (collected by the TET server during a calibration process) which reside within a screen area described by a specified central point and a radius. Successively, the calibration can be recalculated, set and saved using again START_CALIBRATION, adding or without adding new calibration points.

Other commands keep the original syntax but they have been improved: for example the calibration process is better synchronized and can be interrupted without causing deadlock.

Better CPU usage
Talk2tobii thread doesn't suck any more. In the previous implementation, the TET command forwarding thread in talk2tobii, was performing a continuos loop which engaged the CPU actively all the time and most of it in doing useless operations. Now, the thread sleeps all the time and it's awaken only when it's necessary to carry out an operation.

Time references (*)
In the previous version of talk2tobii, remote timestamps from the TET server were not converted to local timestamps on the base of the synchronization result. Talk2tobii used its proper timestamps source (which was completely unrelated to libtobii's one and to remote time values), to mark both incoming data from the TET and outgoing data (user events). This source was however the same as used in the psychtoolbox (http://psychtoolbox.org/), for which talk2tobii was originally designed to work with: by this way one could compare directly timestamps taken with the GetSecs() function of the psychtoolbox and times taken within talk2tobii. In the new version, in order to extend the usability of the toolbox, relate local & TET remote timestamps and unify time management, things have been changed and this is no more true. Timestamps are now all generated inside talk2tobii or converted from remote timestamps (TET timestamps) to local ones with special routines which compensate for clock asynchrony. The more precise way to establish a relation between an external time source and the talk2tobii time source is to issue a TIMESTAMP command, which is a special syntax command (see above).

The libt2t API is in t2t.h where you find some comments.
To get help in using t2tio, just type 't2t -h' at the shell prompt.
To understand how to reference libt2t from Python or Perl, look at the examples directory where you can find sample code which covers all the possible commands.

<u>Multithreading Environment</u>

T2T is thread safe.

T2T VERSIONS HISTORY

v1.0
First T2T version, very Beta.

v1.1
Various bugs fixed.
Samples applications improved.
Introduced tracking during calibration.

v1.2
Added Mexmaci files to support the 32 bit matlab version on Mac OS X.

Luca Filippin - January 2011 - luca.filippin@gmail.com
Subscribe to http://groups.google.com/group/t2t-pkg to get more info and post questions