Multimedia **A**uthoring and **M**anagement using your **E**yes and **M**ind

H2020-ICT-2014 - 644780

D4.3 - Initial implementation of MAMEM's middleware and Interaction SDK

Dissemination level:	Public (PU)
Contractual date of delivery:	M18, 31/10/2016
Actual date of delivery:	M16, 31/08/2016 (draft) M19, 05/12/2016 (final)
Workpackage:	WP4 Middleware for Interaction through Eyes and Mind
Task:	T4.2 - Middleware implementation and Interaction SDK T4.3 - Algorithms integration and technical verification
Type:	OTHER
Approval Status:	Final
Version:	0.7
Number of pages:	39
Filename:	D4.3_ Initial_Implementation_MAMEM_Middleware_InteractionSDK_final.docx

Abstract: The goal of D4.3 is to deliver the documentation of the software implementing the first version of MAMEM's middleware and interaction SDK. The software is delivered along with this document in the form of an installation package and handles most of the work required for the integration of all the system subcomponents. The document presents the current development status of the platform, step by step installation instructions and detailed instructions on how to modify or extend the functionalities of the system.

co-funded by the European Union

Copyright

History

Version	Date	Reason	Revised by
v0.1 (alpha)	28/7/2016	Included the table of contents.	Walter Nistico
v0.2	20/8/2016	First version to be delivered for internal review.	Walter Nistico, Georgios Liaros
v0.3 (beta)	26/8/2016	Beta version of the document.	Walter Nistico, Georgios Liaros
v0.4	29/8/2016	Elaborated version incorporating the comments of the internal reviewers.	Walter Nistico, Korok Sengupta, Spiros Nikolopoulos, Georgios Liaros
V0.5 (final - draft)	31/8/2016	Final review	Markus Plank, Spiros Nikolopoulos
v0.6	02/12/2016	Pre-final version	Markus Plank, George Liaros
v0.7	05/12/2016	Final editing and submission	Spiros Nikolopoulos

Author list

Organization	Name	Contact Information
SMI	Walter Nistico	walter.nistico@smi.de
CERTH	Georgios Liaros	geoliaros@iti.gr
CERTH	Spiros Nikolopoulos	nikolopo@iti.gr
UNI KO-LD	Korok Sengupta	koroksengupta@uni-koblenz.de
UNI KO-LD	Raphael Menges	raphaelmenges@uni-koblenz.de
SMI	Markus Plank	markus.plank@smi.de

Executive Summary

The purpose of this document is to deliver the documentation of the software implementing the first version of MAMEM's middleware and interaction SDK that incorporates the first-phase results of WP2 and WP3. The software is delivered along with this document in the form of an installation package.

We begin by presenting a short overview of the MAMEM platform discussing the current development status and the offered functionalities. Specifically, we mention the supported sensor devices (EEG, Eye-tracking and GSR) for the base level of the platform, the capabilities of the Interaction SDK and the status of GazeTheWeb browser. Following, we present the system requirements, the contents of the installation package and step by step installation instructions. Finally, we provide instructions on how to get the source code of the project and modify it, if needed. Particular emphasis has been placed on the correct ways to extend the system with additional functionality.

In concluding this document, we discuss our future plans and actions that will be taken for improving the functionality of the MAMEM platform.

Abbreviations and Acronyms

API	Application Programming Interface
BCI	Brain Computer Interface
EEG	ElectroEncephaloGram
GPU	Graphics Processing Unit
GSR	Galvanic Skin Response
GUI	Graphical User Interface
LSL	LabStreamingLayer
SDK	Software Development Kit
SSVEP	Steady State Visually Evoked Potentials

Table of Contents

List of Figures

List of Tables

1 Introduction

This document includes the documentation for the initial implementation of MAMEM's middleware and interaction SDK, from now on referred to as "**MAMEM platform**".

The document is structured as follows: Section 2 presents the current status and capabilities of the platform. Section 3 provides the necessary instructions for installing the platform and Section 4 is addressed to developers who want to modify or extend the functionality of the system.

2 Overview of the MAMEM platform

2.1 Sensors

The MAMEM platform includes the necessary components for connecting sensor devices to the system. Since each sensor may require its own SDK for collecting data and communication with the host operating system, we use *plugins* for compatibility purposes. At this point, plugins for the following four sensors have been implemented:

- **Emotiv EPOC+**, a wireless, easy to use 14-channel EEG headset ("lightweight" configuration).
- **BE Plus LTM**, EBNEuro's wireless, high-density, research-grade EEG headset ("heavyweight" configuration).
- **Shimmer3 GSR+,** a wireless device for monitoring GSR and optical heart rate.
- **SMI iViewRED** for all screen-mounted, remote eye-tracking devices in SMIs product portfolio ("heavyweight" configuration).

Each plugin is associated with an executable file which has the role of connecting a sensor with the Middleware of the system (LSL). For more information on plugins please refer to Section 3.4

2.2 Interaction SDK

The Interaction SDK includes the necessary signal processing and machine learning algorithms for translating the data streams generated from the sensors into commands for the end-user applications. At this point, our Interaction SDK is based on MATLAB, which is widely used in the scientific community. However, as MATLAB is a commercial platform it is planned to migrate to OpenVIBE in the future in order to avoid the inclusion of commercial software into the system.

There are two main uses for the Interaction SDK. The first one is to perform off-line experiments with pre-recorded data in order to evaluate the performance of different algorithms and select the optimal configuration and the second one is to apply the selected configuration in an on-line setting. For the off-line scenario, MATLAB needs to be purchased and installed. By contrast, for the on-line scenario only the MATLAB Compiler Runtime (v9.0.1, 64-bit) must be installed into the system, which is offered for free and has been included in the provided MAMEM installer.

2.3 GazeTheWeb

GazeTheWeb, the web browser that can be operated using gaze, is included in the MAMEM installer. Gaze-based functionalities that are supported by GazeTheWeb at this time are as following (for an in-depth description, refer to Section 2.2 of D3.1 [8]):

- Easy to use gaze-based keyboard for typing text
- Navigating to a web page by typing the URL
- Back/Forward buttons for navigation
- Adding bookmarks
- Zooming and scrolling through a web page

- Click emulation
- Pausing/Resuming the eye-tracker input that can be used for e.g. reading a large amount of text
- Multiple tabs for browsing multiple web pages at the same time
- Easy selection of text input fields inside a web page.

3 Installing the MAMEM platform

3.1 System requirements

The MAMEM system is composed of several components, each with different requirements. In the following, we provide details for a superset of the system requirements and recommended PC configurations that cover all different components composing the MAMEM system. Although not recommended, the MAMEM system can be installed and run on any PC of the user's choice. However, users must be aware that the quality of the recording cannot be guaranteed on devices which do not fulfil specific requirements. In this section we provide guidelines and hardware requirements for using the MAMEM system on a PC. Nevertheless, we cannot guarantee that a particular Laptop or Desktop PC running Microsoft Windows 7™ or 8™ will be sufficient. It is possible that other hardware components or software on a particular PC may interfere with the proper functioning of the MAMEM software. The following instructions are to be considered as guidelines only on how to set up a PC. All information about the system requirements and recommended configuration is provided in the following table.

Requirements	
Operating system	Windows 7™, Windows 8™, Windows 8.1™, Windows 10™
CPU Architecture	64-bit
Disk Space	2GB free space is required to install the MAMEM platform and all necessary components. Additional disk space is required for data recording and analysis.
CPU	Quad-Core Intel Core i5 or i7 of 3rd and 4th generation (Core 3xxx and 4xxx series).
USB	Native USB 3.0 or USB 3.0 Add-On Card with support for SuperSpeed USB 3.0 capability. It is possible that PCs do not support USB 3.0 or the USB chipset on the PC may not be compatible with the Eye Tracker. A USB 3.0 Add-On card may add SuperSpeed USB 3.0 capability to the PC.
Bluetooth	Bluetooth is required for connecting the Shimmer device with the system (GSR and Optical heart rate measurements).
GPU	OpenGL 3.3 support is required
Recommended PC Configuration	
User Account Rights:	Administrator rights are required
Standby Mode	Disable the Standby Mode
Screen Saver	Disable the screen saver
Power Options	Disable any function that will power-down the PC after a period of inactivity

Display	Set to never turn off
Sleep	Set to never put the computer to sleep
Power Plan	Set to High Performance
Closing the Lid:	Set to “Do nothing”
Windows Updates	During the recording sessions the update should be set to “Never check for updates”
Antivirus Software	Add iViewRED to exception list. Contact SMI for Antivirus Software recommendations. During recording sessions only. Turn off scheduled updates. Deactivate virus scanning.
USB Devices	Do not use any other USB devices on the same USB controller as the iViewRED eye tracker during the recording.
Other Software / Applications	Do not use additional applications that consume CPU or USB resources on the PC, since they might have an impact on the eye tracker performance
Firewall and Networking	Add an exception to the firewall when asked for the following applications • GazeTheWeb • EPOC2LSL • ShimmerDevice • EBNeuro BePlusLTM Amplifier • iViewRED • SSVEPDemo
Write Access:	Ensure the operator has “Write” access to the repository directory. Do not try to write to a folder for which you do not have “Write” access

Table 1. System requirements for the MAMEM platform

3.2 Installation package contents

The installation package contains all the necessary components for sensor data acquisition, signal processing, translation of signals into commands as well as a number of applications that can take advantage of the system and provide useful functionality for end-users.

Specifically, the package includes

- 4 plugins for data acquisition from a) Emotiv EPOC EEG headset, b) EBNeuro’s EGI BE Plus LTM EEG device, c) Shimmer3 GSR+ sensor, and d) SMI iViewRED eye tracker.
- The InteractionSDK that processes the data acquired from the sensors and translates them into commands.
- GazeTheWeb, a fully functional web browser that can be operated only with SMI eye trackers.

- GazeTheWeb – Tweet, a customised Twitter interface, optimised for eye-tracker usage.
- GazeTheWeb – SSVEP, an augmented version of GazeTheWeb that also utilizes EEG signals for interacting with the computer.

3.3 Installing the system

Download the setup application from [5] and install the system following the instructions of the installation wizard.

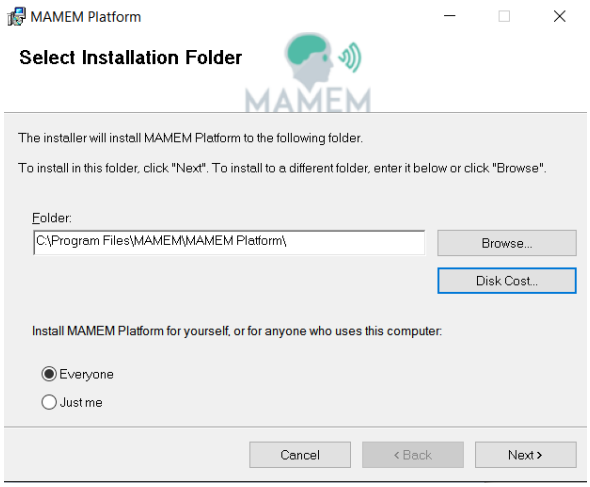


Figure 1. Confirming the destination folder

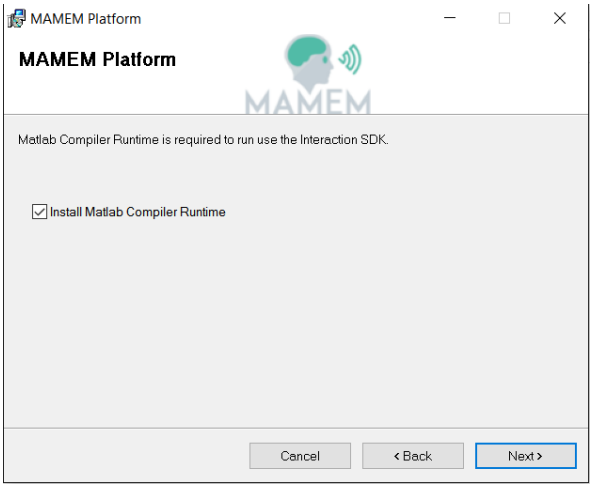


Figure 2. Select if the MATLAB Compiler Runtime should be installed (recommended).

After finishing the installation, the applications and the demos may be accessed via the Start menu under the entry “MAMEM”.

3.4 System setup

3.4.1 Connecting EPOC with LSL

In order to connect the EPOC device with LSL, simply connect the USB Bluetooth dongle into a USB port, power up the device and launch the “EPOC” application. A command line window then will pop-up which prints a line such as “Updated:4” each time a chunk of samples (size=4) is pushed into the LSL stream. This will generate an LSL stream named “EmotivStream” containing the EEG data for all 14 channels. The streaming can be stopped at any time by simply closing the window.

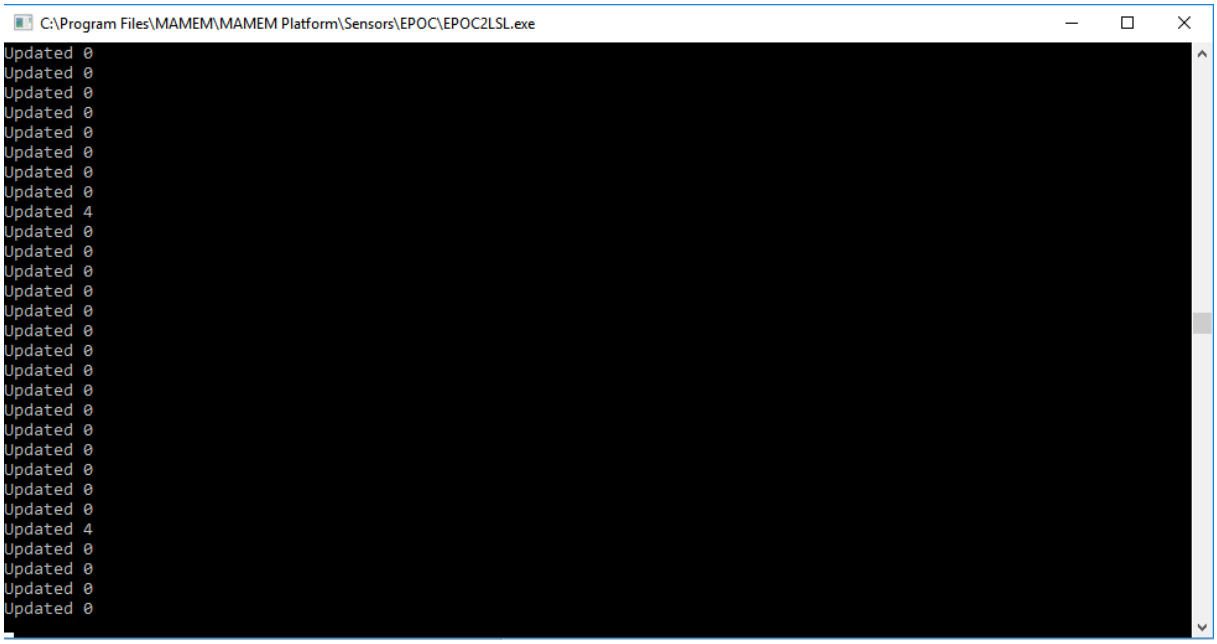


Figure 3. The EPOC plugin

3.4.2 Connecting BE Plus LTM with LSL

Launch the BEPlusLTM application of the MAMEM package. A window such as the following will be opened.

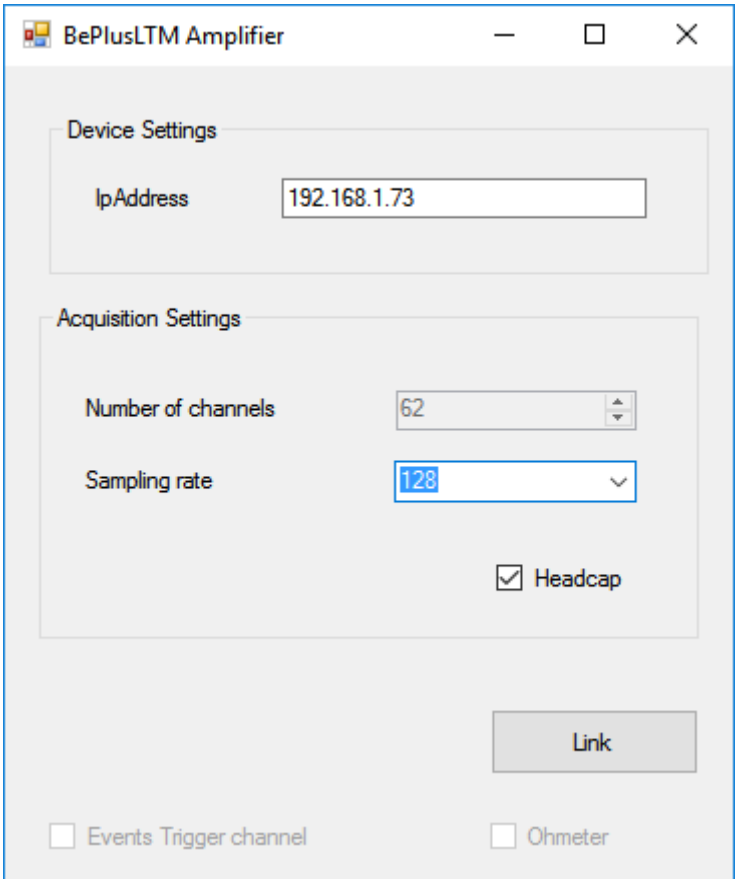


Figure 4. The interface for connecting the BE Plus LTM with LSL.

Fill the IP address of the device in the first input field as it is printed on the back of the device near the power connector ports. In the number of channels field, type “62”, ensure that the “Headcap” checkbox is selected and press “Link”. The LSL stream is now generated with the name “EBNeuro_BePLusLTM_[Device IP Address]”.

3.4.3 Connecting the SMI RED eye tracker with LSL

First connect the eye-tracker to a USB 3.0 port and launch the IViewRed application (provided by SMI), in order to power up the device. To connect it with LSL, launch the “SMI IViewX Connector” application that is provided by the MAMEM installation package. The predefined settings should work in most cases. If you use a dual-monitor setup, ensure that the resolution specified by this window is the one which corresponds to the correct monitor. Press the “Link” button to initialize the LSL stream. A stream named “IViewX” should appear in the LSL network.

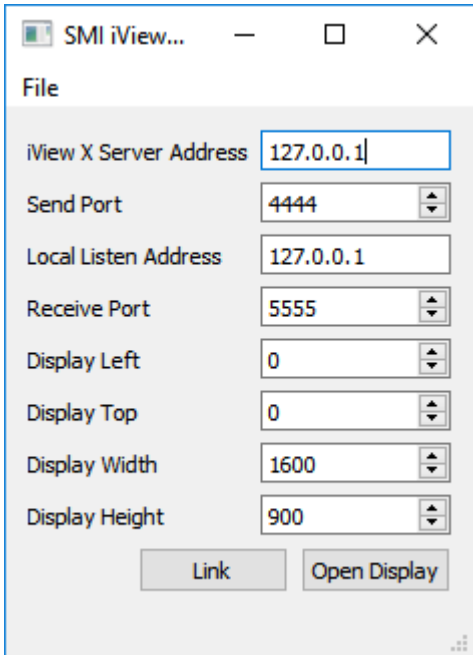


Figure 5. The interface for connecting SMI IViewRed eye-tracker with LSL

3.4.4 Connecting the Shimmer 3+ device with LSL

Before using the shimmer device, you must ensure that the proper firmware is installed. In order to check the firmware, you may use the “Consensys” application which is provided by the Shimmer. The firmware type or version can be checked in the home screen of the application after connecting the Shimmer device to its dock.

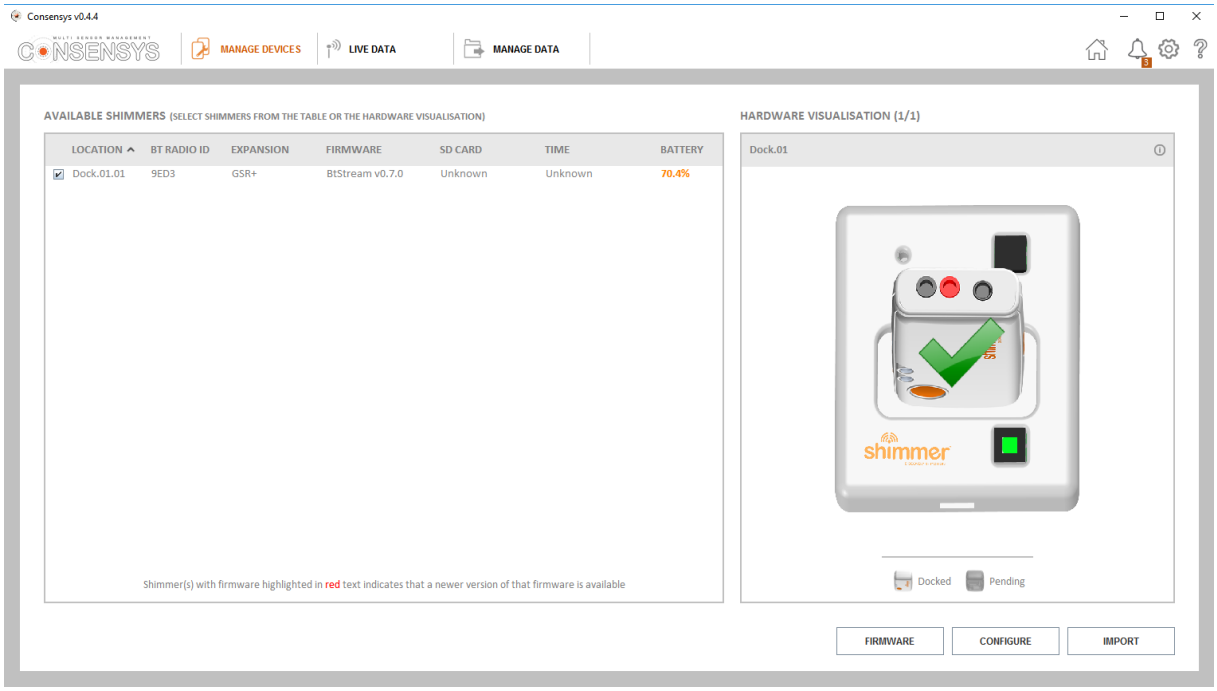


Figure 6. Consensys software suite for configuring the Shimmer device (provided by Shimmer)

It is recommended to program your Shimmer with the "BtStream v0.7.0" firmware. This can be done by selecting the "FIRMWARE" button on the bottom-right of the screen, then select "BtStream_Shimmer3_v0.7.0" and then "PROGRAM". Your Shimmer should now be able to be paired by Bluetooth with your computer by using the password "1234" (default password).

After pairing the Shimmer with the computer via Bluetooth, you may proceed with connecting it with LSL by launching the "Shimmer Device" application, provided by the MAMEM installation package. The appropriate COM Port which corresponds to the Shimmer Bluetooth connection should be initially selected in the first input field. Since there is not a reliable way to detect which of all ports corresponds to the Shimmer automatically, the fastest way is to try and select all the available ports until it works (no error window appears). Finally, click the Link button to generate the LSL stream. A stream with name "Shimmer_COM[x]", should appear in the network containing two channels of data. The first one contains the optical heart beat rate data (beats per minute) and the second one the GSR measured in kOhms.

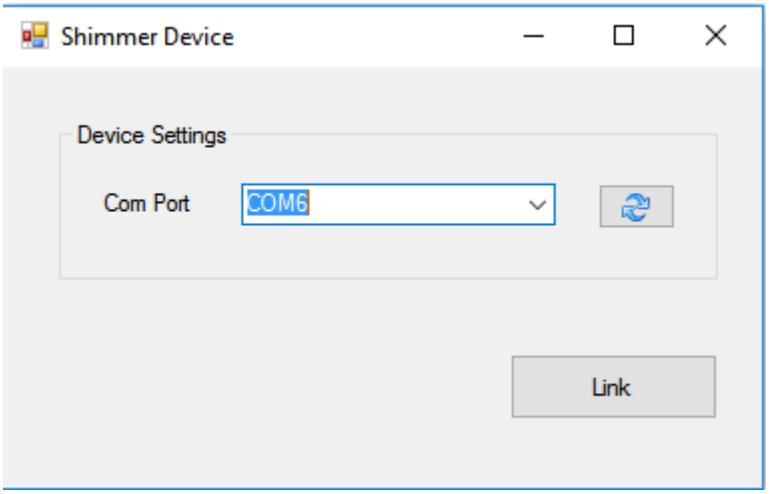


Figure 7. Interface for connecting the Shimmer device with LSL.

3.4.5 Stress Detection service (beta)

The Stress Detection service is a tool that can be used to detect the stress levels of a user in real-time using the GSR measurements from a Shimmer device. Before using this service, a calibration recording must be performed using the Shimmer device of a (recommended) 1-hour duration (this should only be done once, if the same user is operating in more or less the same conditions). The easiest way to perform the calibration recording is by using the LabRecorder application in conjunction with the Interaction-SDK (eeg-processing-toolbox). Another requirement for the calibration step is the LSL Matlab library and the Matlab software.

The calibration can be done with the following steps;

- First connect the Shimmer device with LSL using the instructions provided in Section 3.4.4
- Then launch the LabRecorder application and record your data in an .xdf file.
- When the recording is finished, launch Matlab and after including the LSL library and the eeg-processing-toolbox in your path, execute the following code.

```
sd = eegtoolkit.services.StressDetection;
sd.trainThresholdsFromXdf(xdfFilename);
```

- After the training is finished, save the “sd” object in a .mat file.

When the calibration is finished, start the “Stress Detection” application that was included in the MAMEM installation package. Copy the saved .mat file that was produced in the calibration step into the same folder of the Stress Detection service (should be inside the Program Files folder). Then type the filename in the “Service filename” input field and click the “Start” button. If everything was OK, the status indicator should turn green. Since the stress detection algorithm requires a few amount of samples in order to work, there is a delay of about 90 seconds until a result is produced. After this small delay, a new LSL stream should appear in the network, named “Stress levels” containing the calculated stress levels which is a number between 1 (low stress) and 5 (high stress). These numbers are calculated based on the GSR samples of the past 90 seconds and indicate the average stress level during that period of time.

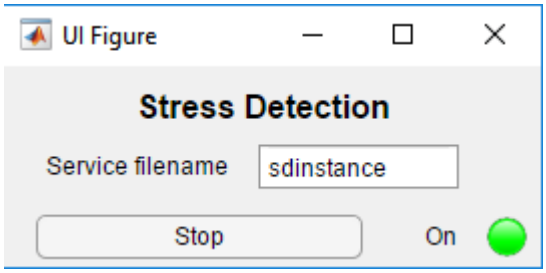


Figure 8. The Stress detection service interface.

3.4.6 Using LabRecorder to record data from the sensors

LabRecorder [6] (included in the MAMEM installation package) is an open-source software written in Python that can be used for recording any type of stream existing in the LSL network. The timestamping of the streams is handled by this software, which ensures the proper synchronisation of the signals.

The file format used by the LabRecorder is XDF. This is a new open general-purpose format that was designed concurrently with LSL and supports all features of LSL streams. To run LabRecorder, you must have installed Python-2.6 or 2.7 in your system.

The interface of the LabRecorder application looks like the following screenshot

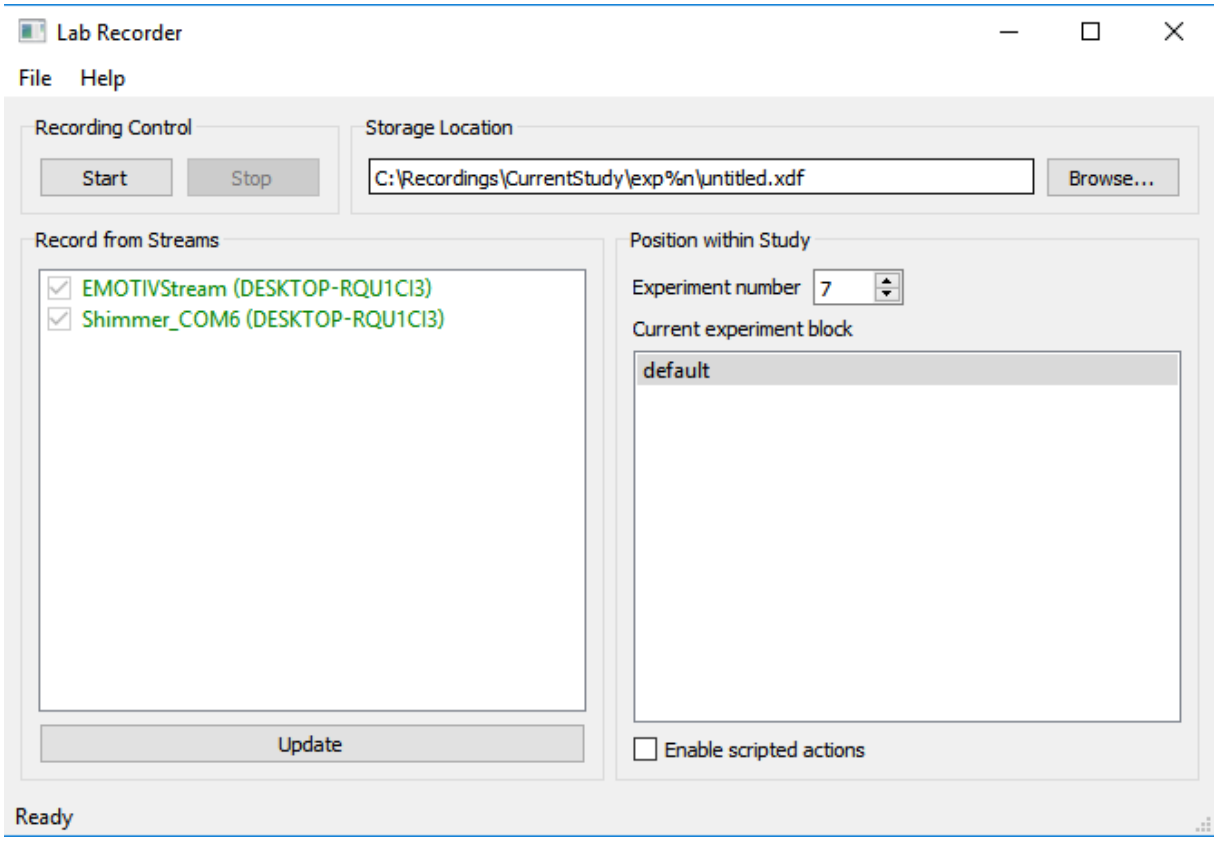


Figure 9. LabRecorder interface for recording LSL streams.

To initiate a recording, simply select the streams that you wish to record, select a filename and click the Start button. The resulting xdf file is supported by the eeg-processing-toolbox

(Interaction SDK) by the "eegtoolkit.util.load_xdf" function and also by the popular EEG toolbox "EEGLAB".

3.5 MAMEM Control Panel

At the current stage of development, the MAMEM Control Panel can be used for monitoring the signals that are captured by the sensors and ensure that they have been connected properly with the system as described in the previous section. Specifically, by using the Control Panel, you can monitor an EEG signal from a specific channel, view the coordinates of the user's gaze that are captured from the eye tracker in real-time, monitor his heart beat rate through the optical sensor and his/her stress levels as they are being calculated in real-time through the GSR sensor. To run the Control Panel, it is required to at least connect an EEG device (either the EPOC or the BEPlus LTM) and the eye-tracker with LSL by completing the steps that were described in the previous section. Optionally, you can also connect the GSR sensor, in order to monitor the Optical Heartbeat Rate and the stress levels of the user.

After launching the Control Panel, you may monitor the signals by selecting the appropriate streams in the "EEG Stream", "Eye tracker stream" fields and pressing the button "View streams".

If you wish to also monitor the stress levels and heartbeat rate of the user, you will have to also connect the Shimmer device with the system, as well as run the Stress Detection service.

After completing all these steps, the Control Panel will look like Figure 10.

On the left half of the screen the selected EEG channel is visualised in a two-dimensional plot, where the *y* dimension corresponds to the electrode's amplitude in μV and the *x* dimension corresponds to the time elapsed in seconds. On the right side of the screen the eye-tracker's gaze coordinates are visualized in a scatter plot where each axis, corresponds to the screen coordinates in pixels. The colour of the scatter plot points indicates the age of the samples (blue are the older points and green are the most recent). Finally, on the bottom of the screen there are widgets that provide on-line information about the detected stress-level and the heartbeat rate of the user.

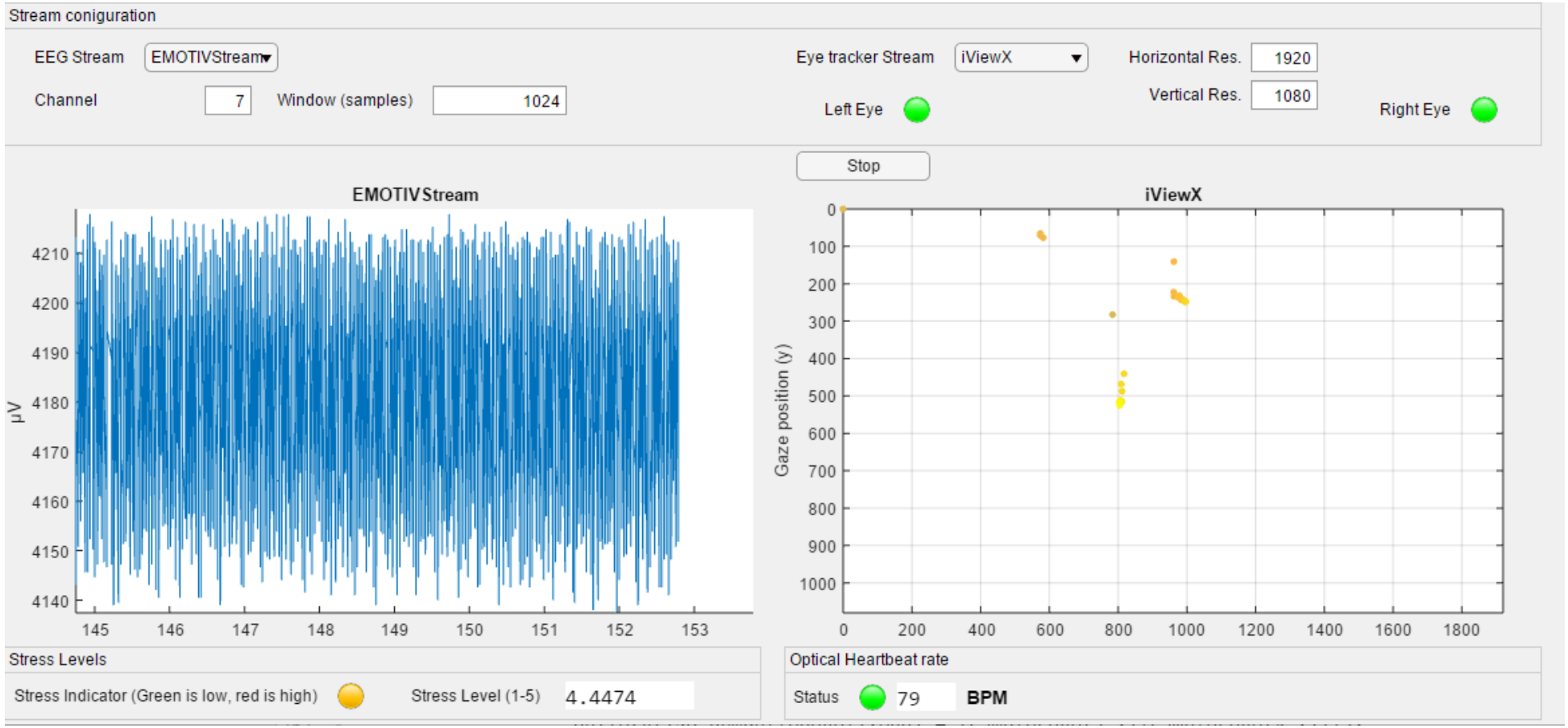


Figure 10. MAMEM Control panel

3.6 Example demos

The demos are pre-configured to use SMI RED eye trackers by default, so the iViewRED server application must be running [4]. In addition, demos that require EEG input will require the EPOC device to be connected. In order to verify that the EPOC device is properly connected, please run the EMOTIV Xavier Testbench application and verify that any EEG signal is shown in the screen as depicted in Figure 11.

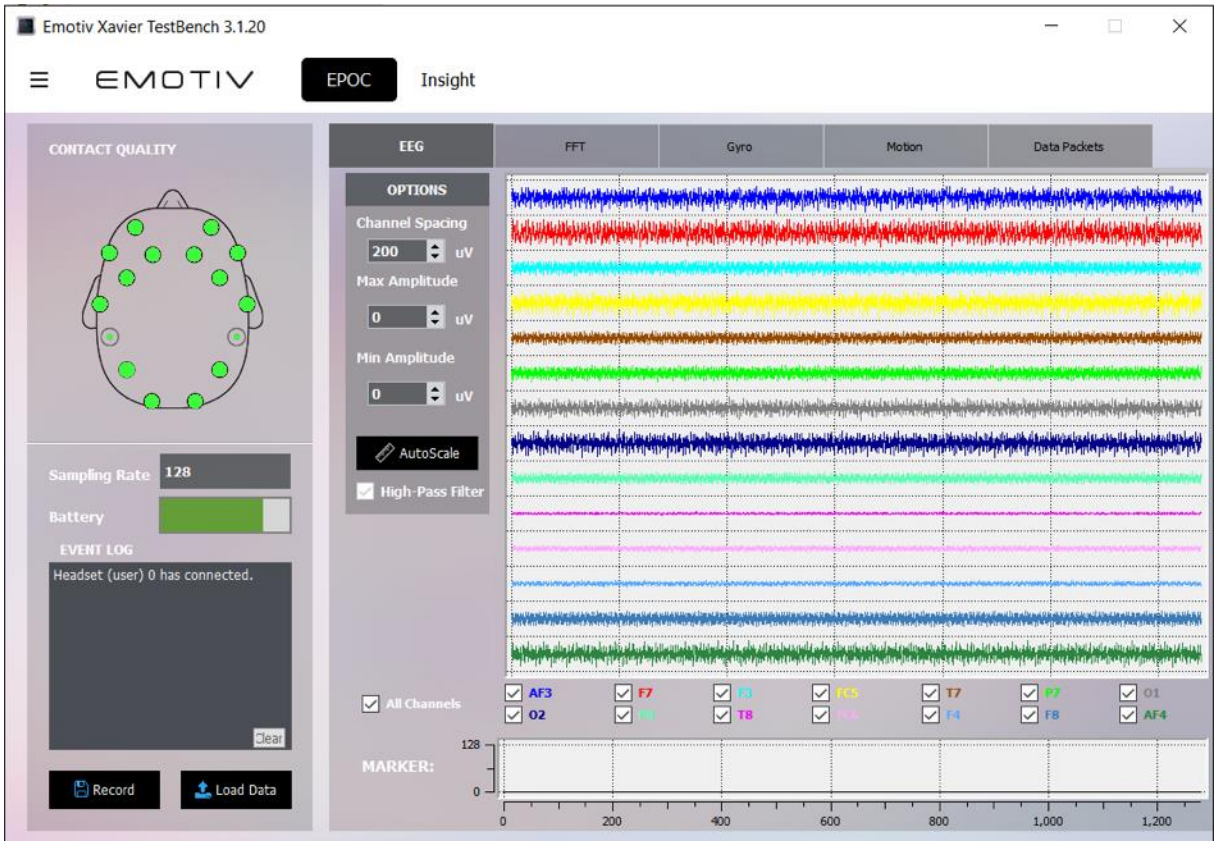


Figure 11. Verifying that EPOC is connected properly

3.6.1 GazeTheWeb – Tweet

GazeTheWeb – Tweet is an interface for Twitter that can be controlled entirely with an eye-tracker. It features the full functionality of Twitter, such as tweeting a message, browsing your feed, searching for trends, finding new people to follow and more. See screenshots below for some examples.

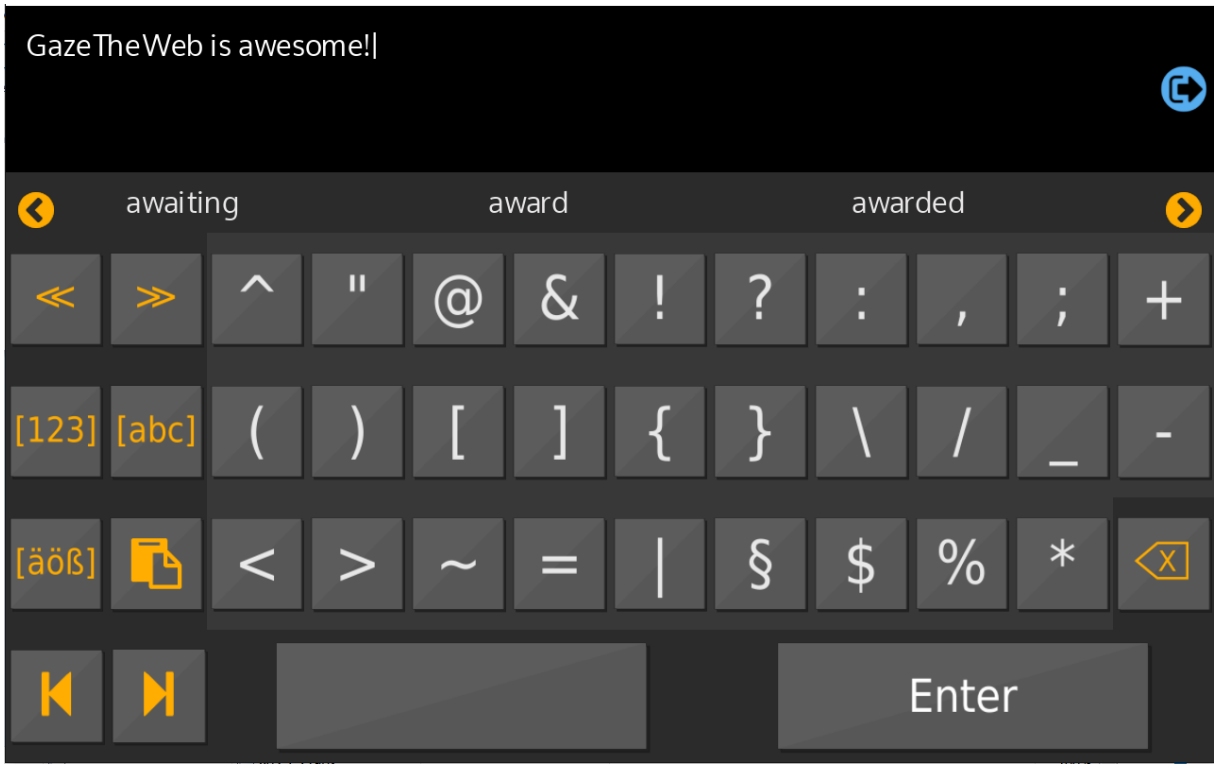


Figure 12. Tweeting a message using the gaze-controlled keyboard

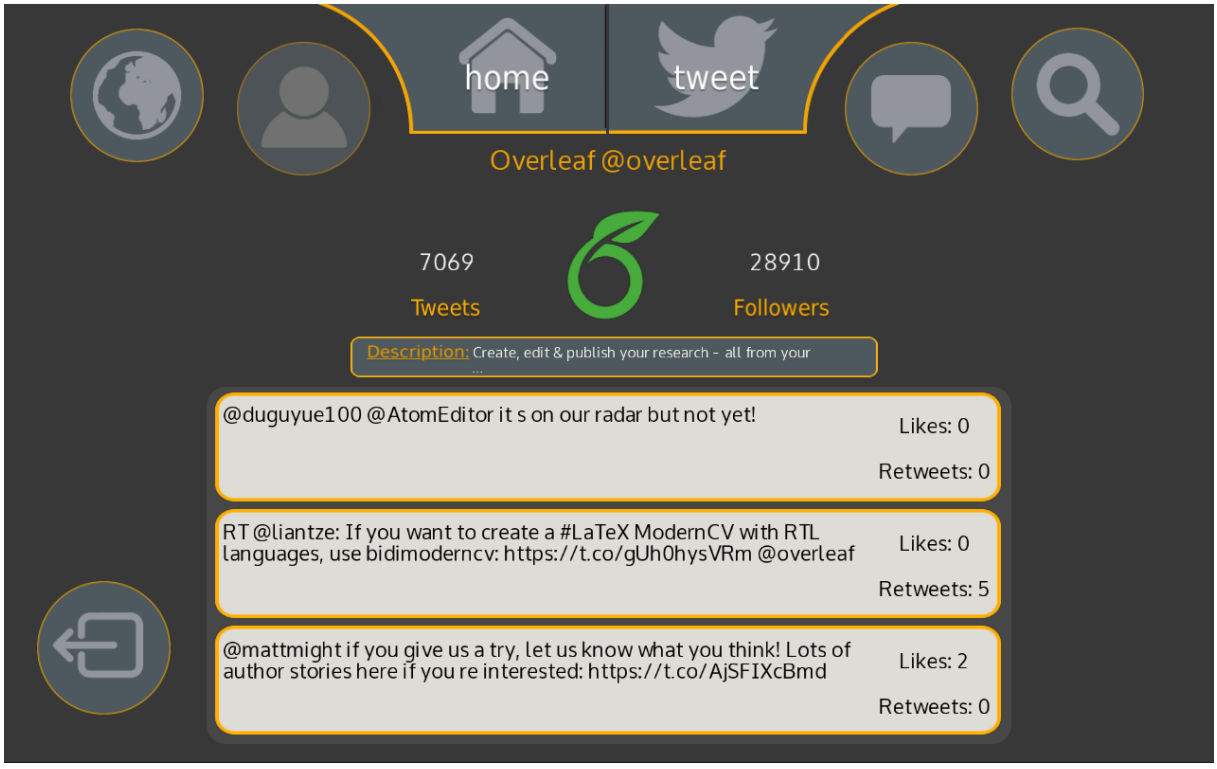


Figure 13. Viewing a Twitter profile

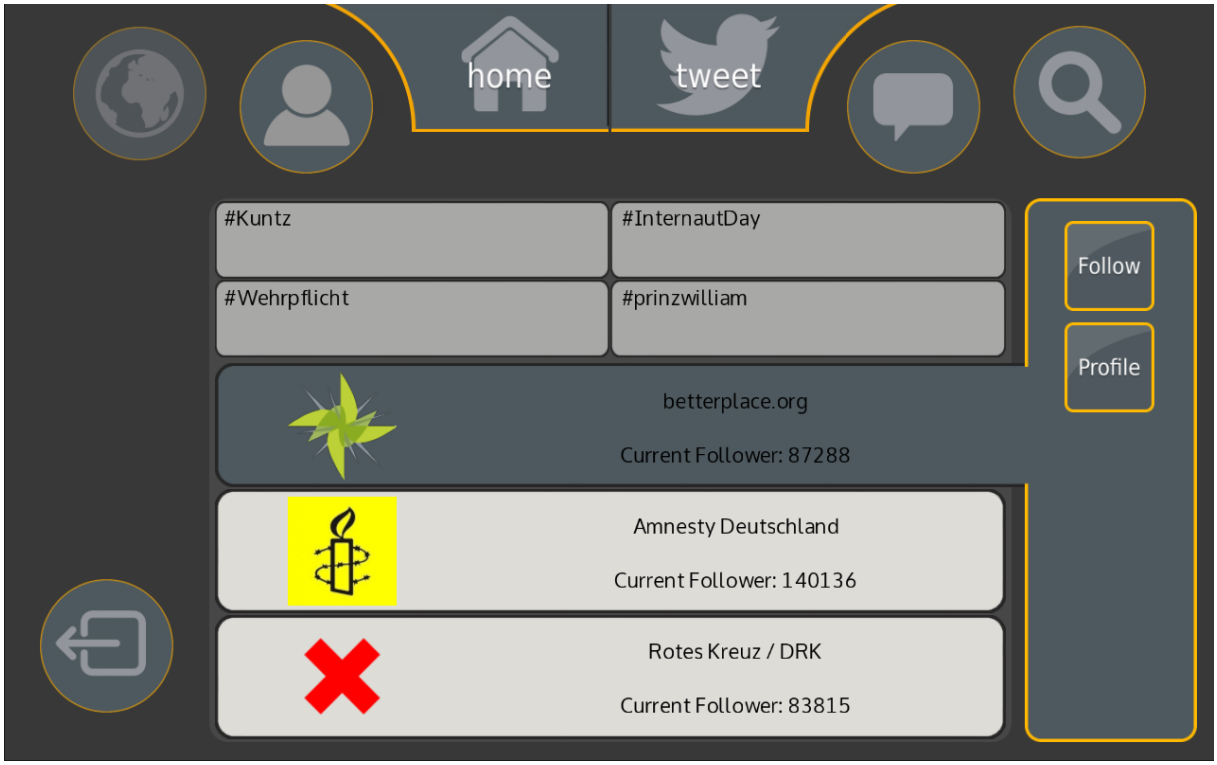


Figure 14. Searching for new people to follow

3.6.2 SSVEP recognition via GazeTheWeb

This example demonstrates how EEG signal can be used as input for controlling a website. To run the demo, first verify that the Emotiv EPOC and the SMI REDn Scientific are turned on and working properly and then launch the “SSVEP Demo” application from the start menu. The demo launches GazeTheWeb browser and navigates you to the homepage of the MAMEM website. In order to trigger the SSVEP recognition screen, first you have to open the side menu from the right by clicking on it and select either the option “The Project” or “Results”. Then the screen shown in Figure 17 is presented, in which you will have to direct your visual attention to the box that corresponds to the desired menu option for 5 seconds. After 5 seconds, the EEG signal captured from the EPOC device is sent to the Interaction SDK and translated into a command. GazeTheWeb interprets the command received back from the InteractionSDK and then transfers the user to the selected subpage of the MAMEM website.

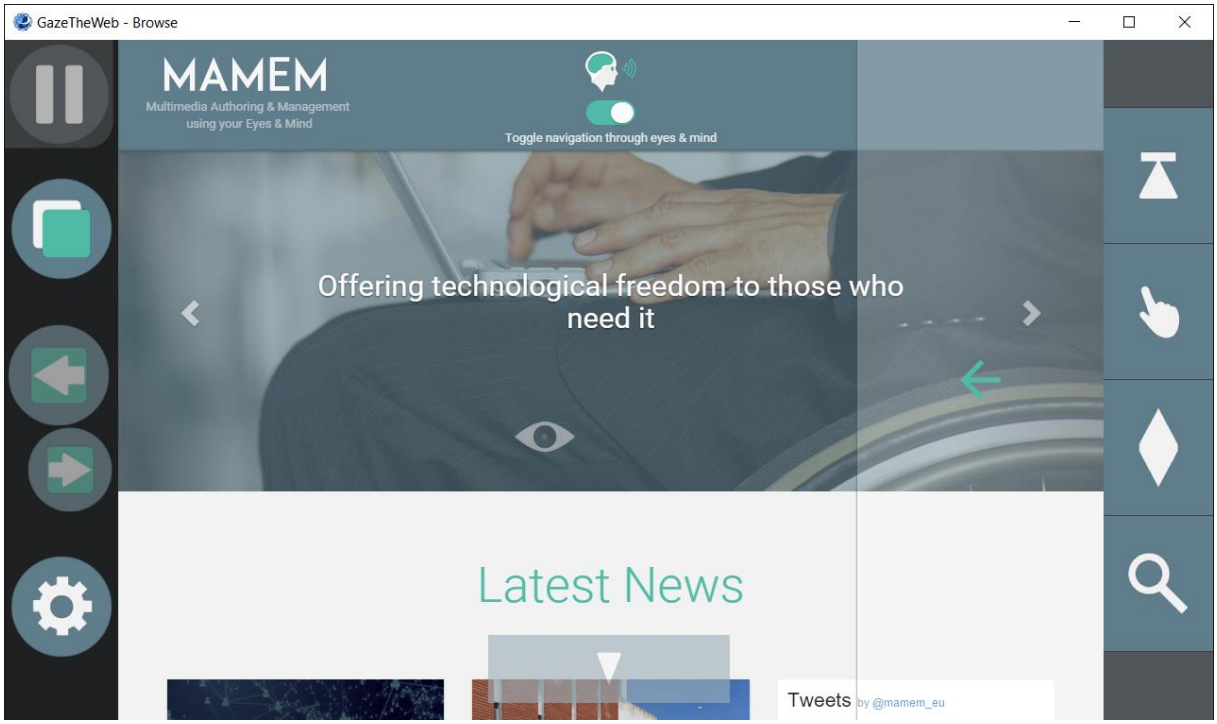


Figure 15. MAMEM website, as seen via GazeTheWeb.

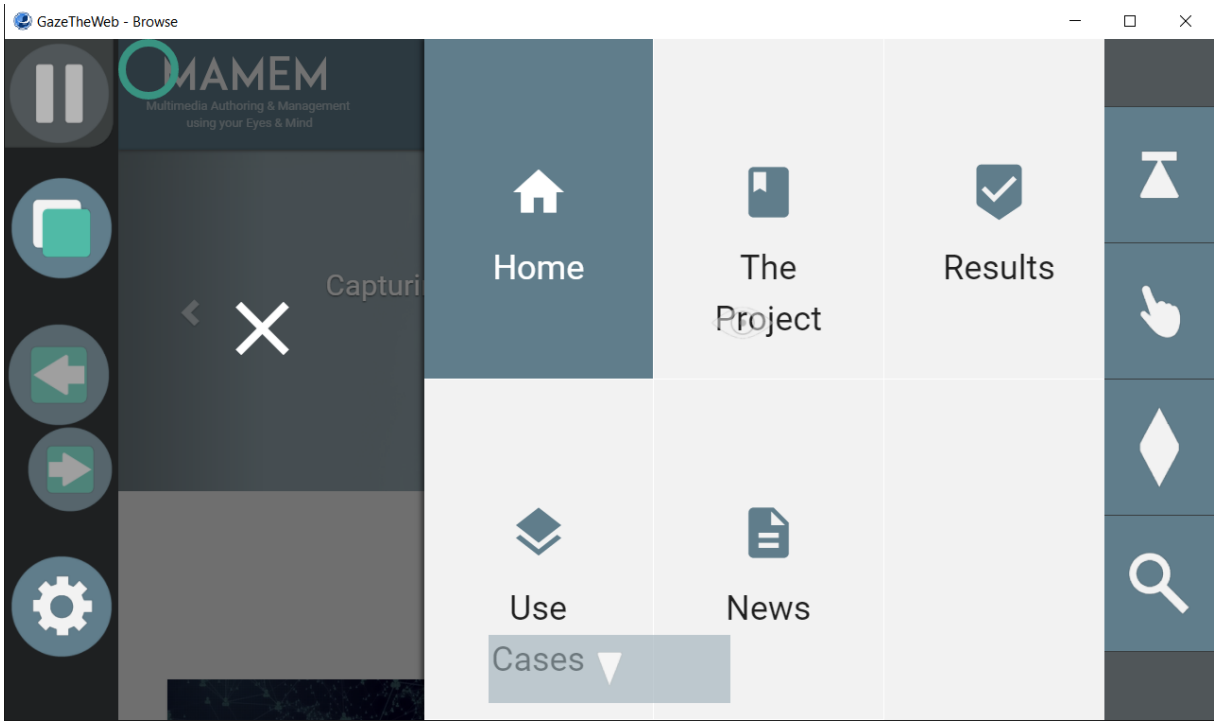


Figure 16. Selecting “The Project” or “Results” will pop up a SSVEP interface and trigger the SSVEP recognition.

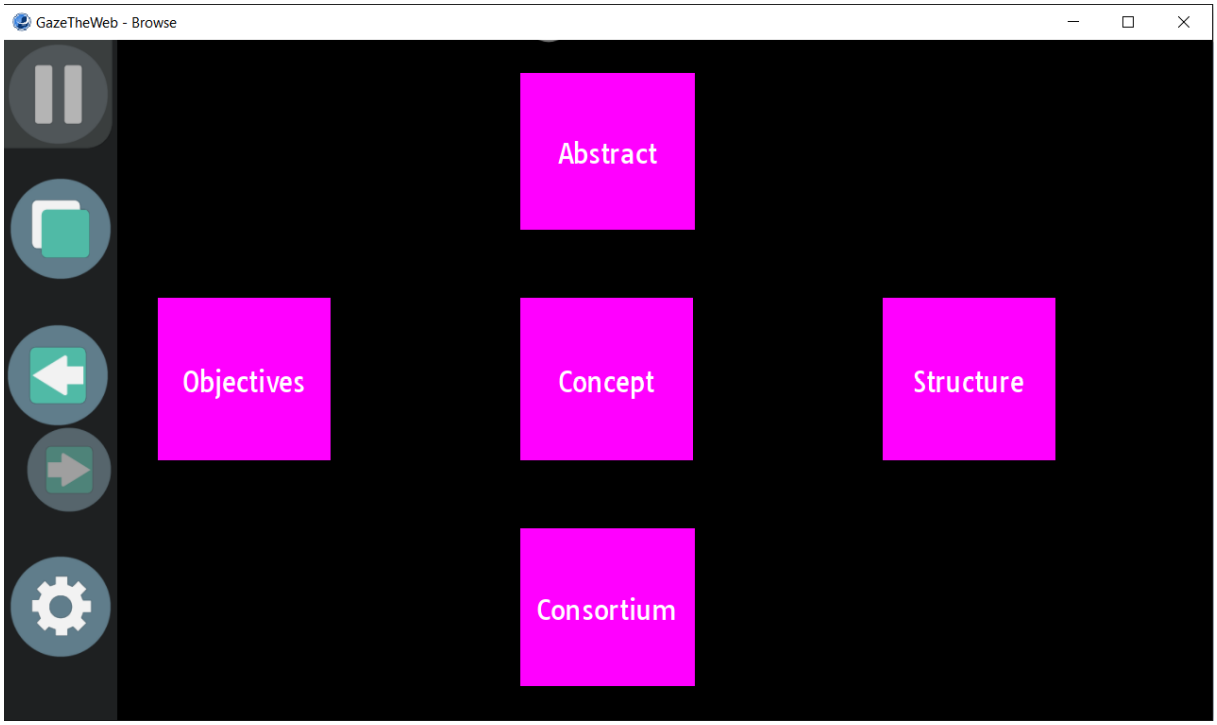


Figure 17. Select the desired menu option by directing your visual attention to the corresponding box. After 5 seconds of EEG signal capturing you will be navigated to the desired page.

4 Extending the system

In the previous Section we have described how the MAMEM platform can be easily installed using a user-friendly installation package. On the contrary this section will cover topics that are related to modifying or extending the source code and is mainly addressed to software developers and interfaces designers.

4.1 Git instructions and code organisation

The MAMEM platform is hosted on GitHub as an open source project and is organised in submodules that are maintained and developed independently by each partner. The easiest way to download the code of every single component is by cloning the master project [1] and using the following git instructions for fetching the submodules:

```
git clone https://github.com/MAMEM/mamem-platform.git
cd mamem-platform
git submodule update --init --recursive
```

Once the projects have been downloaded, the following folders and files will be generated in your file system:

- The Application-networking folder which contains tools for data communication between applications and the core of the system.
- The Applications folder which includes the end-user applications of the system such as the GazeTheWeb browser.
- The Interaction SDK folder which contains the code of the data processing algorithms.
- The Middleware folder which contains the code responsible for the data stream generation and synchronization.
- The sensors folder which contains the code for building the plugins that acquire the data from the sensors.
- A setup script that should be run in order to prepare your system for building the source code. This is achieved by downloading the necessary dependencies and declaring environment libraries.
- A build script that can be run for generating the binaries of the project.

Currently, the source code of the following software can be downloaded:

- The EPOC plugin for communication with LSL.
- The InteractionSDK (eeg-processing-toolbox).
- GazeTheWeb browser
- Various networking tools that have been used during development

4.2 Building the source code

4.2.1 Sensors and InteractionSDK

The source code of the MAMEM platform can be built only under the Windows operating system (Windows 7 or later). Further, the following packages must be installed on your system

- Microsoft Visual Studio 2015
- MATLAB (for compiling the Interaction SDK)
- Emotiv Premium SDK

Each individual component may be built separately, but it is way faster to use the included batch script files. To build the project using the script files, first run “setup.bat” in order to download all the necessary dependencies into your system and then run script “build.bat”. Once the build script has been executed successfully, the generated products and demos will be available into a generated folder named “Build” at the root path of the project.

4.2.2 GazeTheWeb

GazeTheWeb employs the Chromium Embedded Framework (CEF) for embedding a Web browser engine based on the Chromium core. CEF offers a convenient way to add web browser control and implement a visual layout GUI (e.g., eyeGUI [2]) in the application. More specifically CEF provides the infrastructure for HTML rendering and JavaScript to a C++ project.

With the CEF framework, GazeTheWeb application can control resource loading, navigation, context menus and more, while taking advantage of the same performance and HTML5 technologies available in the Google ChromeWeb browser. One of the other essential components of GazeTheWeb is the interface layout specifically designed for eye-based interaction via eyeGUI framework.

Architecture

Figure 18 shows GazeTheWeb architecture, which is primarily structured in two major components, i.e., CEFImplementation that implements the necessary functions of Chromium Embedded Framework for browser functionality, and Visual Browser to generate visualizations and customized objects for gaze interactions.

CEFImplementation The top part of the structural diagram in Figure 18 signifies the necessary functions for Chromium Embedded Frame to work. The basic communication works by sending the JavaScript code from main application to renderer process, and receiving messages from renderer process (like DOM nodes). A mediator class (CefMediator) abstracts all Web related classes (Tab, DOMNode) and serves as interface between CEF implementation and visual browser.

Visual Browser The visual browser consists of Web object which contains multiple Tab objects, where each tab has texture/image (called WebView) which is registered in CEF Implementation for drawing the web page. An active tab renders its texture/image to the screen with OpenGL. - Gives one full control over visualization, we can zoom into the webpage or darken certain areas or cut out stuff. Atomic actions like "zooming" etc. are implemented as actions and can be combined to pipeline. Figure 19 is an example of click emulation, where the ZoomAction delivers coordinates for ClickEmulation. Master owns all objects - Web organizes Tabs - Tab has one WebView filled by CEF Renderer at onPaint - DOMNode are filled by CefMediator - Triggers generated from information from DOMNodes - Triggers push back Pipelines

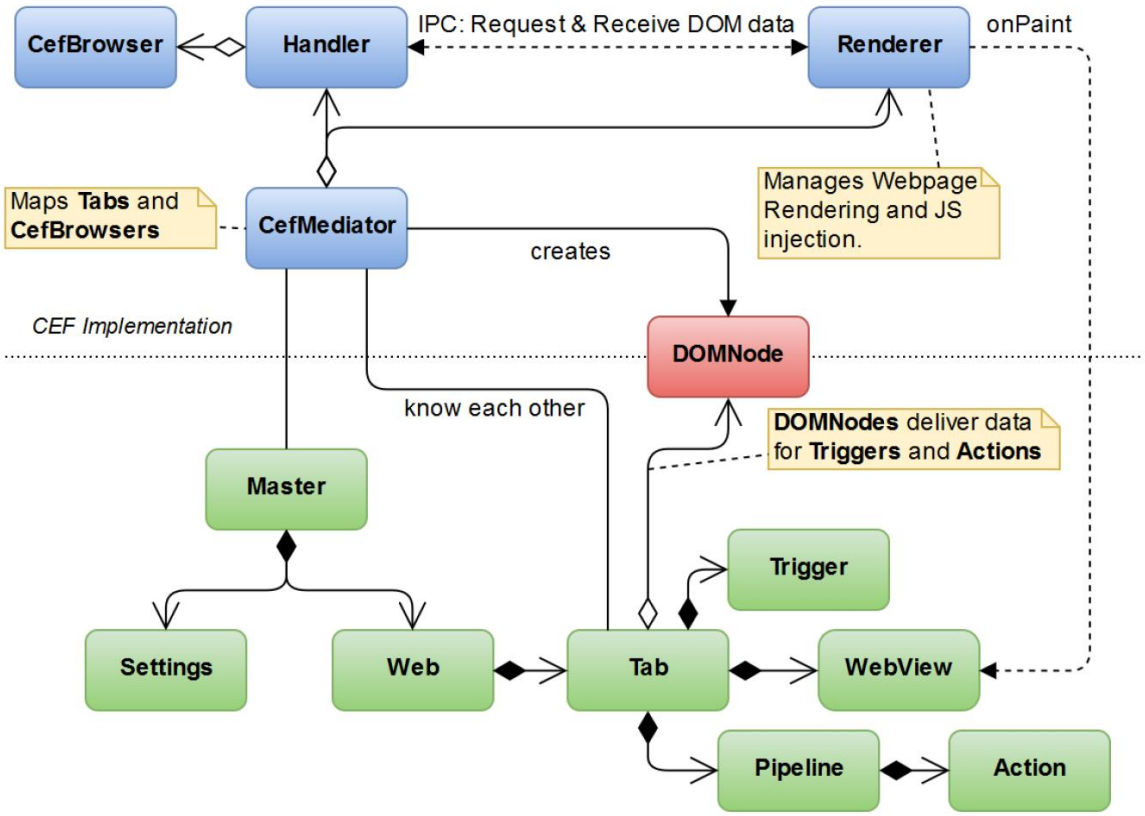


Figure 18. GazeTheWeb Architecture

Compilation

The software source can be compiled on either Windows with Visual Studio 2015 as 32bit project or on Linux with GCC 5.x as 64bit project. In addition, the graphics card must support OpenGL 3.3 or higher. Following are the steps for compilation

1. Clone GazeTheWeb repository from Github
2. Download Windows 32bit CEF 3.x binaries of branch 2454 at [7]. Extract the downloaded files and copy following content into the locally cloned repository:
    a. Include
    b. Libcef.dll
    c. Release
    d. Debug
    e. Resources
    f. README.txt
    g. LICENSE.txt
    h. DO NOT overwrite the provided CMakeLists.txt, otherwise the Prototype and Client is not found.
3. If prototype should be built too, one has to include its subdirectory in the main CMakeLists, line 532.
4. Create a build folder somewhere and execute CMake to create a project, which can be compiled.

a. *CLIENT_SMI_REDN_SUPPORT* defines, whether Client should compile with support for SMI iViewX
b. *CLIENT_TOBII_EYEX_SUPPORT* defines, whether Client should compile with support for Tobii EyeX SDK

Usage

The software usage like interface, startup, and input possibilities can be configured by editing the setup file provided at src/Setup.h, shows an example of these settings.

```
static const bool FULLSCREEN = false;
static const int INITIAL_WINDOW_WIDTH = 1280;
static const int INITIAL_WINDOW_HEIGHT = 720;
static const float DURATION_BEFORE_INPUT = 1.f; // wait one second before accepting input
static const bool PAUSED_AT_STARTUP = false;
static const bool ENABLE_WEBGL = false; // only on Windows
static const bool LOG_DEBUG_MESSAGES = false;
static const bool LOG_DEBUG_MESSAGES = true;
static const std::string LAB_STREAM_OUTPUT_NAME = "BrowserOutputStream";
static const std::string LAB_STREAM_OUTPUT_SOURCE_ID = "myuniquesourceid23443";
static const std::string LAB_STREAM_INPUT_NAME = "MiddlewareStream"; // may be set to same value as LAB_STREAM_O
```

Figure 19. GazeTheWeb usage configuration

For the validation purpose, there is a log file named “log.txt” created at first run. Take a look at that file if something behaves not as expected.

Following are the software used in GazeTheWeb:

- CEF3: **https://bitbucket.org/chromiumembedded/cef**
- GLM: **http://glm.g-truc.net/0.9.7/index.html** (MIT license chosen)
- GLFW3: **http://www.glfw.org**
- iViewX: Connection to the iViewX SDK, copyright SMI GmbH (**http://www.smivision.com)**
- TobiiEyeX: Connection to Tobii EyeX SDK (**http://developer.tobii.com/eyex-sdk)**
- eyeGUI: **https://github.com/raphaelmenges/eyeGUI**
- FreeType 2.6.1: **http://www.freetype.org** (FreeType license chosen)
- Spdlog: **https://github.com/gabime/spdlog**
- Liblsl: **https://github.com/sccn/labstreaminglayer**
- Boost: **https://github.com/boostorg/boost**
- Text-csv: **https://github.com/roman-kashitsyn/text-csv**
- TinyXML2 (used one from eyeGUI library): **https://github.com/leethomason/tinyxml2**

4.3 Incorporating a new sensor in the system

In order to add a new sensor to the system, a plugin must be created that will acquire data from the sensor and push the data to the MAMEM Middleware. The Middleware is based on the LabStreamingLayer library which handles the data networking/synchronisation and can

be downloaded from [3]. The LabStreamingLayer SDK supports most major programming languages, so any of these languages may be chosen in order to interface with the new sensor.

The following C++ code will demonstrate how to create a LabStreamingLayer stream and push samples through it:

- Start by including the C++ library header:

```
#include "lsl_cpp.h"
```

- Then, declare a stream info object which specifies that the sensor will be identified as "MyEEG" (here, EEG is sampled from 8 channels with 100 Hz, each channel carries one float (32 bit) value):

```
lsl::stream_info info("MyEEG","EEG",8,100,lsl::cf_float32,"myuid");
```

- Create a stream outlet object with the properties just specified in the info object

```
lsl::stream_outlet outlet(info);
```

- Now samples can be streamed using the outlet

```
// Declare a buffer for our sample
float sample[8];

// Stream the samples into the outlet
while (true) {

    // fictious function which returns a sample from my EEG device and
    // copies it into our sample buffer
    getSampleFromMySensor(sample);

    // Push the sample into the LSL outlet
    outlet.push_sample(sample);

}
```

4.4 Extending the Interaction SDK

The Interaction SDK is a MATLAB toolbox that contains a set of signal processing algorithms capable of translating the acquired sensor data into meaningful commands. Specifically, it covers all the necessary steps of the pipeline, such as the pre-processing of the signals, feature extraction/selection and classification. It is placed right in the middle of the architecture and works by reading the sensor data from the middleware, running the data through the processing pipeline and finally, transmitting the result to the end-user applications. For the time being, the toolbox is focused on recognizing SSVEP patterns from raw EEG signals but more recognition tasks are foreseen in the future, such as the recognition of motor imagery and Error-Related Potentials.

4.4.1 Compiling a custom SSVEP detection module

To get started with the Interaction SDK, it is recommended to view the existing example named "SSVEPDemo". The demo is pre-configured to work with the Emotiv EPOC and GazeTheWeb so as to browse MAMEM's website by using an SSVEP interface. The EEG

source stream is declared in the "datastream" variable and can be changed to another sensor if desired, by setting the name of the stream that is generated from the respective plugin. More properties can be configured by going through the code comments below. The recognition pipeline algorithms can also be changed by modifying the properties of the LSLWrapper class.

```
%Add dependencies to the Matlab path, LibLSL is required for this
example
%to work
addpath liblsl-Matlab\;
addpath liblsl-Matlab\bin\;
addpath liblsl-Matlab\mex\;
%Initialize LSL Wrapper class
lsl = eegtoolkit.util.LSLWrapper;
%Declare the name of the stream that will contain the EEG data
datastream = 'EMOTIVStream';
%Declare the name of the stream through which the events will be
%communicated
eventstream = 'MyEventStream';
%Size of signal that will be used for the recognition task
bufferSize = 5; %in seconds
%The event code that will trigger the recognition task
eventCode = 100;


% RECOGNITION ALGORITHM CONFIGURATION

%Indicate the number of stimuli (5) and their frequencies
stimulus_frequencies = [12 10 8.57 7.5 6.66];

%Indicate which channels of the data (different electrodes) will be used
ss = eegtoolkit.preprocessing.SampleSelection;
%We will use all EPOC channels for this example
channels = 1:14;
ss.channels = channels;
ss.sampleRange = [1,640];

%Sampling rate for the EPOC headset is 128Hz
samplingRate = 128;

%Another required parameter for the CCA algorithm
numberOfHarmonics = 4;
%Initialize the Canonical Correlation Analysis class for the stimuli
%recognition
cca =
eegtoolkit.featextraction.CCA(stimulus_frequencies,channels,samplingRate
,numberOfHarmonics);

%Simple classifier that uses the max value of the features to assign the
%label
maxC = eegtoolkit.classification.Max;

%Assign the algorithm configuration to the LSL Wrapper class
lsl.preprocessing = {ss};
lsl.featextraction = cca;
lsl.classification = maxC;
%Find the streams in the network
lsl.resolveStreams(datastream,bufferSize,eventstream);
```

```
%Pause for 5 seconds to allow the stream to gather some data
pause(bufferSize);
%Run the recognition task. The task runs indefinitely until is
specifically
%interrupted
lsl.runSSVEP(eventCode);
```

Finally, to package the app into an executable, the MATLAB "Application Compiler" should be used as seen in Figure 20. In the "Main File" field use the entry point script or function of the program. MATLAB should automatically detect whichever files to include in the package except any dynamic libraries that are called such as the LSL binaries (e.g. liblsl64.dll). To include these libraries into the package, you will have to manually add them into the "Files required for your application to run" field. When you are finished, click the "package" button to generate the executable. Note that the MATLAB compiler runtime itself should not be included in the package.

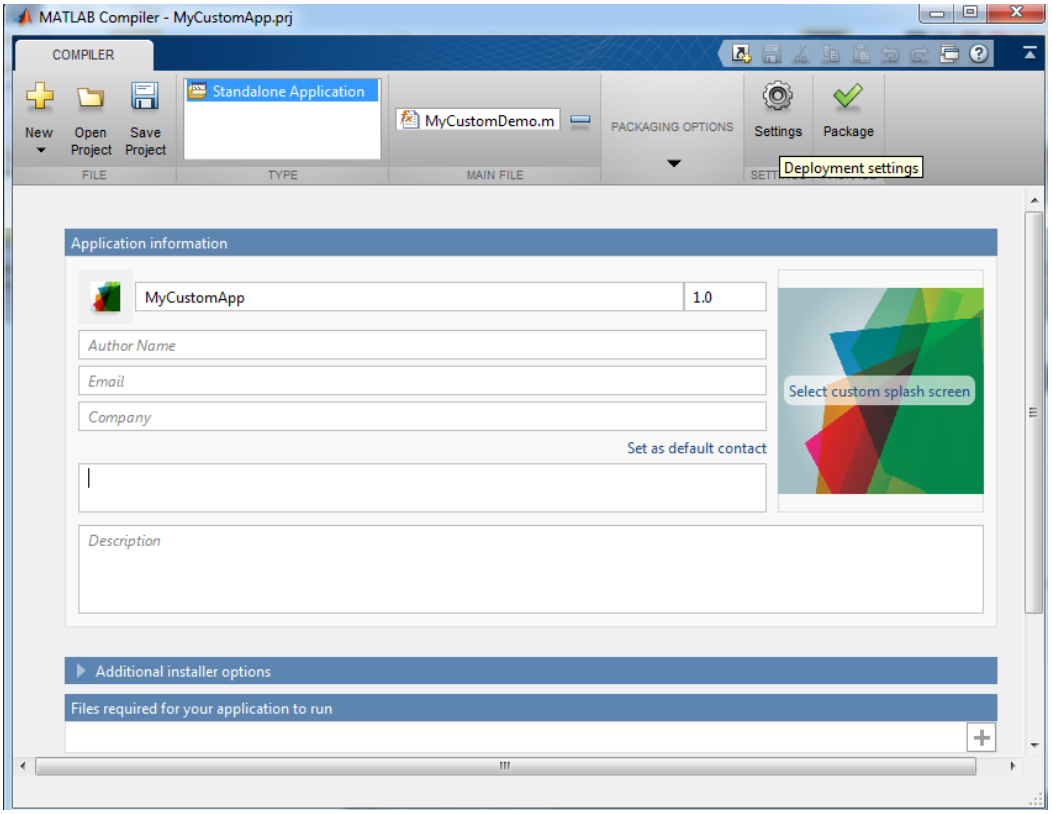


Figure 20. Packaging the Interaction SDK into an executable file.

4.4.2 Incorporating new algorithms into the toolbox

It is possible to extend the Interaction SDK with new algorithms by adding a new class in the appropriate package and extending the abstract class of it. The abstract class defines the methods that need to be implemented in order to be compatible with toolbox, as well as the appropriate inputs and outputs of these methods. The toolbox consists of the following extendable packages

- **Preprocessing:** Extend the “PreprocessingBase” class. The processing takes place inside the “process” method. It takes as input a set of “Trial” structs, containing the signals, and outputs a new set of “Trial” structs with the processed signals
- **FeatExctraction:** Extend the “FeatureExtractionBase” class and implement the “extract” method. It takes as input a set of “Trial” structs and outputs an “InstanceSet” object containing the extracted features and their labels (if available).
- **FeatSelection:** Extend the “FeatureSelectionBase” class and implement the “compute” method. It takes as input the aforementioned “InstanceSet” object and outputs a new “InstanceSet” after selecting the most important features.
- **Classification:** Extend the “ClassifierBase” class and implement the “build” and “classifyInstance” methods. The “build” method receives an “InstanceSet” object to be used for training the classifier, while the “classifyInstance” method uses the trained classifier to predict and output the labels of the test instances.

All the classes that are added to the toolbox should also include a method that prints their configuration parameters (getConfigInfo method), as well as the timestamp of their last executed operation (getTime method).

Extensive documentation of the eeg-processing-toolbox is provided by D3.2 [9] in Appendix A.

4.5 Incorporating custom eyeGUI layers in GazeTheWeb

Creating a new eyeGUI layer is a relatively simple task.

An .xeyegui file must be created inside the folder '**content/layouts/**' which resides in the application root folder. Also an .seyegui file that acts as a stylesheet must be created inside the folder '**content/stylesheets/**' too.

There are eyeGUI custom elements available based on the xml language for the purpose of building a custom layer. The available elements are documented online on the gitHub page of eyeGUI [2].

Once the layout is set up correctly, it must be declared inside the application source code in Tab.cpp by using the function “AddLayout()”. The stylesheet name and path is declared inside the .xeyegui so it does not need to be declared in the application.

Then the custom eyeGUI layer can be invoked and either enabled or disabled by using the “setVisibilityOfLayout()” function. There is also a “setValueOfStyleAttribute()” function to access the elements of the layer directly and make changes if needed.

Extensive documentation of the eyeGUI library is also provided by D3.2 [9] in Appendix B.

5 Troubleshooting and FAQ

5.1.1 GazeTheWeb

Application crashes at startup

- Does your graphics card support OpenGL 3.3 or higher?
- Have you followed the installation instructions in the readme?

Mouse cursor does not disappear after a while

- Is the eye tracker still active? Sometimes it crashes at connection with the application.

User interface does not respond

- Is the calibration of the eye tracker correct? Maybe recalibrate. If you wear glasses, try without and recalibrate if possible.
- Is pause mode activated? One can deactivate it with the button at the upper left corner or by hitting Tab on keyboard.

Application leaves fullscreen after startup

- Windows Firewall asks for permissions for the application. Just agree and go back to the application.

5.1.2 SSVEP Demo

GazeTheWeb browser does not start

- Run GazeTheWeb manually as an Administrator.

MATLAB crashes after 5 seconds of presenting the SSVEP layout interface.

- Make sure that the Emotiv EPOC is properly connected and pushing data. You can verify this by running the “EmotivXavierTestbench”.

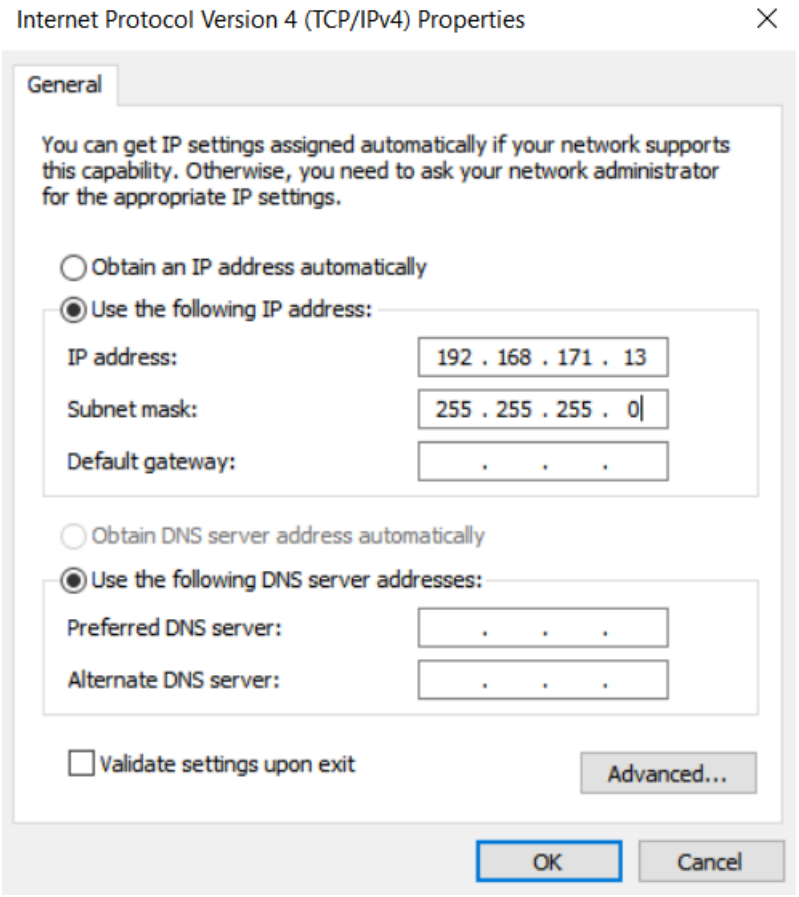
5.1.3 Sensors

EPOC plugin generates an empty stream

- Run the EmotivXavierTestbench application, that is provided by Emotiv and plug/unplug the USB receiver dongle until a signal is shown in the screen. Try also turning on/off the power of the device.
- If it still doesn’t work, then that probably means that you have an older version of EPOC which is not supported by our plugin. You may be able to fix that by replacing the edk.dll file inside the EPOC folder with your own, provided that you have purchased the Developer License of EPOC and downloaded the premium SDK.

BEPlus LTM is unable to connect with the device

Set up the network board with a static IP address of the same subnet of the device as shown in the screenshot below and then try again



6 Conclusions

In this document we have discussed the process of installing the MAMEM platform with the help of a single installation package, as well as how it is possible to modify or extend parts of the system with more functionality. Although the development status of the platform can be still considered as an early stage, the foundation has been established which is expected to assist greatly in further development. Our short term plans for the future are focused on improving the user experience and usability aspect of the platform by adding more graphical user interfaces and automated troubleshooting mechanisms, in order to assist inexperienced users. The long term development plan on the other hand is to focus on the extendibility of the platform in order to encourage more open source contributions on the system as well as the exclusion of the commercial software MATLAB from the system and replacing it with OpenVIBE for our Interaction SDK.

7 References

[1] https://github.com/MAMEM/mamem-platform, Source code of MAMEM platform

[2] https://github.com/raphaelmenges/eyeGUI/wiki, eyeGUI wiki page

[3] https://github.com/sccn/labstreaminglayer, LabStreamingLayer library

[4] http://www.smivision.com/en/gaze-and-eye-tracking-systems/support/software-download.html, iViewRED download page

[5] https://github.com/MAMEM/mamem-platform/releases/, MAMEM platform

[6] https://github.com/sccn/labstreaminglayer/wiki/LabRecorder.wiki, The LabRecorder Wiki page.

[7] https://cefbuilds.com/#branch_2454

[8] D3.1 - Eye-tracking-, EEG- and biofeedback-based control with meso level control paradigms, MAMEM Consortium, July 2016. url (requires authentication to MAMEM wiki): http://mklab.iti.gr/mamem/images/b/bb/D3.1_BCI_Interaction_final.pdf

[9] D3.2 - Multi-modal interaction with meso and high level control paradigm, MAMEM Consortium, November 2016, url (requires authentication to MAMEM wiki):

http://mklab.iti.gr/mamem/images/9/9c/D3.2_MultiModal_Interaction_With_Meso_and_High_Level_Control_Paradigms_Final.pdf