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

H2020-ICT-2014 - 644780

D3.2 - Multi-modal interaction with meso and high level control paradigms

Dissemination level:	Public (PU)
Contractual date of delivery:	Month 18, 31/10/2016
Actual date of delivery:	Month 16, 31/08/2016 (draft), M19, 05/12/2016 (Final)
Work package:	WP3 - Signal Processing for Interaction Control
Task:	T3.4 - Novel paradigms for multi-modal interaction
Type:	Prototype
Approval Status:	Final
Version:	0.8
Number of pages:	57
Filename:	MAMEM_deliverable_D3.2
Abstract: The goal of D3.2 is to understand the potential of multimodal algorithms and its implementation in the context of Eye tracking, Brain Computer Interaction (BCI) and bio signals and to investigate how we can leverage their co-existence in a system if combined together. Here, we also look into the various issues that have been encountered with multimodal interaction and specifically on the advantages and limitations of each modality in the context of a multimodal experience. In this respect, the goal of D3.2 is to facilitate the transition from single modal to multimodal experience.	
The information in this document reflects only the authors views and the European Community is not liable for any use that may be made of the information contained therein. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.	

Figure 1: co-funded by the European Union

Copyright

History

Version	Date	Reason	Revised by
v0.1(alpha)	15/07/2016	Table of Contents to be checked by the consortium	Chandan Kumar and Korok Sengupta
v0.2(beta)	20/08/2016	Added Content for internal review	Dario Comanducci, Markus Plank, Spiros Nikolopoulos, Chandan Kumar
v0.3	29/08/2016	Further sections modified for content	Elisavet Chatzilari, Chandan Kumar, Spiros Nikolopoulos, Korok Sengupta
v0.4	31/08/2016	Final version (for review) incorporating proof edits	Chandan Kumar, Korok Sengupta
v0.5	29/10/2016	Added new experiments and prototype details	Elisavet Chatzilari, Chandan Kumar, Raphael Menges, Spiros Nikolopoulos
v0.6	30/10/2016	Final version incorporating proof edits	Chandan Kumar, Spiros Nikolopoulos
v0.7	01/12/2016	Update of the draft version	Chandan Kumar, Elisavet Chatzilari
v0.8	05/12/2016	Final editing and submission	Spiros Nikolopoulos

Author List

Organization	Name	Contact Information
UNI KO-LD	Chandan Kumar	kumar@uni-koblenz.de
UNI KO-LD	Korok Sengupta	koroksengupta@uni-koblenz.de
UNI KO-LD	Raphael Menges	raphaelmenges@uni-koblenz.de
UNI KO-LD	Steffen Staab	staab@uni-koblenz.de
CERTH	Elisavet Chatzilari	ehatzi@iti.gr
CERTH	Spiros Nikolopoulos	nikolopo@iti.gr
CERTH	Vangelis Oikonomou	viknmu@iti.gr
CERTH	Kostas Georgiadis	kostas.georgiadis@iti.gr
CERTH	Georgios Liaros	geoliaros@iti.gr
CERTH	Katerina Adam	katadam@iti.gr
CERTH	Fotios Kalaganis	fkalaganis@iti.gr

Abbreviations and Acronyms

HCI	Human Computer Interface
MMI	Multi Modal Interface
SCR	Skin Conductance Response
API	Application Programming Interface
BCI	Brain Computer Interface
ECG	ElectroCardioGram
EEG	ElectroEncephaloGram
GSR	Galvanic Skin Response
MD	Muscular Disorder
SMR	Sensorymotor Rhythms
HR	Heart Rate
HRV	Heart Rate Variability

Executive Summary

The overall objective of WP3, and in extension of this deliverable, is to develop the necessary algorithms for translating bio-signals (i.e. eye-tracking based, EEG and GSR signals) into meaningful control. In that direction, this deliverable documents the progress of understanding individual signals and then bringing them together with fusion techniques as Multi Modal Interfaces. Our intention in this document is to investigate the different modalities, and combine them with primary mode of eye-based interaction (i.e. in GazeTheWeb framework). We review the related literature in multi modal interaction in Section 2 and further describe the specifications for motor impaired and MAMEM subjects. Section 3 looks into the modality of Eye and Mind in unison and how the two is brought together in MAMEM with their existing challenges. Section 4 talks about multimodal inputs and smarter user interfaces by using patterns from the psycho-physiological signals, thereby improving the user experience.

GazeTheWeb framework, which provides a path between eye tracker and the application interface, has been proposed as an interaction platform for MAMEM subjects. In Section 3, we briefly summarize GazeTheWeb functionalists and how the gaze modality is being used in MAMEM. We particularly discuss the current limitations of GazeTheWeb framework (i.e., eye based interaction) and how other modalities could assist in resolving the limitations. Section 3.2.1, 3.2.2 and 3.2.3 focuses on the different techniques of understanding and utilizing brain signals like SSVEP, SMR & ErrP, along with their limitations.

Optimizing gaze with other modalities is an important pathway for multimodal environment. In this direction, Section 4 presents our approach, design and preliminary experiments for multi modal optimization. Section 4.1 talks about optimizing gaze control with SMRs and incorporate a switch to improve the Midas Touch issue. Problems with ErrPs and their solutions have been investigated and experimented in Section 4.2. n integration approach of EEG sensor through ErrP has been experimented to provide correction mechanism for spellings. This experiment paves the way for future experiments, which utilizes the power of ErrPs and SMR to improve the MMI on GazeTheWeb Browser.

Contents

List of Figures

1 Introduction

Inherent interaction for human beings has always been multi-modal. Multiple senses work together in unison to process information from our environment to generate the action we need to do. Multimodal interaction in systems seeks to leverage the fact that human being can communicate via multiple senses, eye, sound, touch, movement, etc. These senses are then treated as inputs to the system that learns the commands of the user and works accordingly. MAMEM tries to bring forth this very technology involving the psycho-physiological signals (from the brain and the body) at disposal and convert them into input systems for multimedia interaction. Development of varied interaction methods for MAMEM is the natural progression towards building interfaces that take advantage of the devices that are at our exposure and try to integrate signals from multiple sources for better interaction of multimedia content.

MAMEM seeks to bring the accessibility of the computer systems to people who cannot use conventional modes of interaction like mouse, keyboard etc. Signals from the eye and the brain if used in unison would make the interaction smoother and more effectively than before. It will help people go beyond the conventional methods of input and expand the usability of these interaction systems to be beyond the parameters of experienced users. Since eye tracking provides a natural way to navigate computer systems, this has become the primary mode of interaction in MAMEM, and in this regard we have presented GazeTheWeb as a major interaction system and made it publicly available.

Furthermore, the electroencephalogram data gives us an insight into the working of the brain and also gives us an opportunity to understand stress and other neural artifacts that can be incorporated in the MAMEM framework. EEG and BCI are still in their native state of research. Signal generated from the brain are so unique and high dimensional that it would take us some time to really comprehend what exact stimuli causes those signals and from where it is being generated. Brain signals from alpha, beta frequency bands along with accuracy of eye tracking signals would give us a lead in providing more control to GazeTheWeb browser that we have built.

Initially, Steady State Visual Evoked Potentials (SSVEPs), which is known from the literature to provide the most accurate setting for EEG-based BCI applications, was investigated. MAMEM has publicly released two datasets consisting of EEG signals captured from 11 able-bodied subjects under three different experimental protocols. However, on one hand, for EEG-based interaction, SSVEPs provide state-of-the-art performance on the task of selecting a box among many, on the other hand, it can be considered as tiring and uncomfortable for the end user. And thus, different methods are being looked into for further research into how smooth the signal acquisition can be so as not to make the user tired.

Sensorimotor Rhythms(SMR) have been investigated as well as their main advantage over SSVEPs is that they do not depend on external stimuli for invoking brain signal patterns. Error Related Potentials(ErrP) was also investigated and found to be extremely quick in detection but it has relatively low accuracy and there is the challenge of flase positive signals. In MAMEM, we investigate the Sensorimotor Rhythms (SMRs) to understand signals that indicate state changes while the user navigates on the GazeTheWeb browser for reading or selection task. This "switching" task or state change task is being experimented to understand if this can be used in conjunction with signals from the eye tracking system to eradicate or reduce the most common problem of eye tracking : "Midas Touch" problem.

MAMEM's multimodal framework also seeks to include another modality : Galvanic Skin Response (GSR). In addition to the brain signals that can be used to understand the cognitive state, GSR gives us a fast and direct approach to understanding the stress component. The

proposed algorithm for GSR has the advantage that it does not require training, which can be a tedious process for the user but it only requires that the user wears the sensor for a calibration period. GSR will add on another modality that can be used as a relevance feedback to the GazeTheWeb Broswer that will be optimizing searches based on user preferences and reaction.

The Shimmer GSR+ hardware also gives us access to heart rate variability. As much as the GSR looks in to the stress related assessment, Heart Rate Variability is another metric that is being investigated to understand the human response to the designated use of the multiple devices for multimedia interaction.

This deliverable is structured as follows. First, we discuss the background literature review Section 2 of Multimodal Interaction in general, then with respect to motor impaired and finally how we use the knowledge gained from these two and use it for MAMEM. It is followed by interaction via eye tracking and EEG driven Control Paradigm in Section 3. Afterwards, we focus on the challenges and approach for Multimodal Interfaces in Section 4.

2 Background

2.1 Multi-modal Interaction

Today, we stand at the cross roads of research areas that include computer vision, psychology, artificial intelligence, and many others. Computer systems are becoming more ubiquitous and in that process, natural human behaviour and signals are being used as a part of the system to deliver effortless output based on human emotions, physical conditions and psychological conditions. Systems slowly are incorporating speech and body gestures like posture, hand gestures [28] to express emotion, mood, attitude, and attention there by joining the bandwagon of multi-modal systems [23].

A. Jaimes and N. Sebe [12] puts multimodal systems as *a multimodal HCI system is simply one that responds to inputs in more than one modality or communication channel (e.g., speech, gesture, writing, and others).* But multi-modal systems as we know it are in a stage where researchers are comfortable using multiple modalities separately and merge the results at the application stage. A possible reason as put by the authors of the survey [12] is the absence of a complete understanding of the roles of different modalities and their interplay. But there is a shift in the adaptation of natural behavior of human beings and thus "human centered interaction and interfaces" are being built. Our voice, hands, and entire body, once augmented by sensors such as microphones and cameras, are becoming the ultimate transparent and mobile multimodal input devices.

Apart from techical prowess, one of the biggest advantages of multimodal systems is its potential to expand the accessibility of the computing device to diverse and non specialist users and promote newer technology to people who were initially not aware of the progress of the technology. Multimodal systems or frameworks increase the accessibility irrespective of age, skill, cognitive style, sensory and motor impairments, native language dependency or temporary illness. The flexibility of multimodal systems permits the alternation of input modes thereby preventing overuse or physical damage of a particular modality during use over extended periods of time. Another reason of the surgence of multi-modal systems is the improvement in stability and robustness of the system. Multimodal systems give their users a more powerful interface for accessing and interacting with information having sophisticated visualization and multimedia outputs [26].

Some of the other potential advantages of multimodal systesms are enlisted below [27]:

- They permit the flexible use of input modes, including alternation and integrated use.
- They support improved efficiency, especially when manipulating graphical information.
- They support shorter and simpler speech utterances than a speech-only interface, which results in fewer disfluencies and more robust speech recognition.
- They support greater precision of spatial information than a speech-only interface, since pen input can be quite precise.
- They give users alternatives in their interaction techniques.
- They lead to enhanced error avoidance and ease of error resolution.
- They accommodate a wider range of users, tasks, and environmental situations.
- They adapt to during continuously changing environmental conditions.
- They accommodate individual differences, such as permanent or temporary handicaps.

- They help prevent overuse of any individual mode during extended computer usage.

Points to keep in mind while integrating multimodal interfaces was provided via a survey by Lalanne et.al [19] which used the 7 layered protocol of HCI of Neilsen [25]:

- Goal
- Task/Pragmatic
- Semantic
- Syntactic
- Lexical
- Alphabetical
- Physical

Given below is a table that showcases the sensory modalities relevant to multimodal human-computer interaction [4].

Modality	**Example**
Visual	Face Location
	Gaze
	Facial Expression
	Lip Reading
	Face Based Identity
	Gesture
	Sign Language
Auditory	Speech Input
	Non-Speech Audio
Touch	Pressure
	Location and Selection
	Gesture
Other Sensors	Sensor-based Motion Capture

2.2 Multimodal Interaction for motor impaired

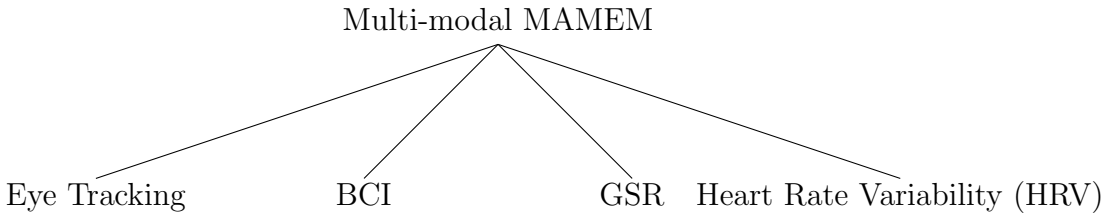
Motor impairment is broadly caused by [18, 30]:

- peripheral problems affecting muscles
- problems in the central nervous system affecting output to muscles
- sensory problems affecting muscles, movement and balance.

Importantly, motor impairments are present in a wide range of bodily and psychological diseases as classified by the World Health Organization (ICD-11) or the American Psychological Association (DSM-V). These relate to problems with movement, sitting, standing and the use of hands or the function of arms. Motor impairments vary from temporary (broken limbs) to permanent. Sometimes these are degenerative and sometimes these present a variable pattern of complaints. The disability is not always visible. In some cases the disability will be accompanied by painful symptoms. This usually results in reduced taxability.

It is because of these issues that its very important to think of different modalities and bring them together under one umbrella to help these people overcome their problems, without causing additive cognitive load to their existing conditions. Multimodal systems that involve auditory feedback and haptic feedback has been researched and experimented to provide better solutions to visually impaired [17]. Systems like HARMOINE (Hybrid Augmented Reality Multimodal Operation Neural Integration Environment)have been developed to lower patient's cognitive load by involving robotics in multimodal framework [15]. As mentioned in the Section 2.1, using the other senses for input into a computerized system could possibly provide a solution to the problems and thereby build a framework of different modalities to be accessed where ever required. Eyes could form one input in conjunction with brain signals, or voice commands, even movement of any limb to denote a particular designated signal that suits the person using this technology. Previous section and deliverable have shed light on the availability of different commercially and research applications covering individual modalities like just the eye, or just the voice or gesture or movement. Multimodal system are therefore an extension of these individual modalities accessing each of the individual psycho-physiological signals and making the interaction much better and comfortable for the user using it.

2.3 Multi-modal Interaction for MAMEM



MAMEM has been conceptualized with the aim to synchronize eye commands with other psycho-physiological signals. The Graph above gives a clear vision of the different modalities that could be and are involved in the framework that is being built for MAMEM.

Eye tracking for MAMEM is being done keeping in mind certain specifications a) binocular sampling rate of at least 60Hz, b) instant calibrationless setup, c) eyewear compatibility, d) gaze tracking range of 80 horizontal, 60 vertical, e) gaze position accuracy of 0.5 over all distances, f) live data streaming, g) lightweight, and h) standalone functionality of at least 2 hours. Commercially available eye trackers having high resolution give liberty to the patient to move their head freely. Saccade measurements and shortest latencies with fully remote, fiducial-free and contact-free setup are possible, even with less compliant subject groups. The fully automatic calibration takes only a few seconds and maintains drift-free accuracy throughout the experiment.

Additionally to eye tracking, we expect the BCI signal to indicate cognitive load associated with different tasks. Based on the research of the several research groups [5], [6], [21], [16] in BCI applications Emotiv's EPOC can be considered an optimal solution with respect to the relationship between quality and price. It is equiped with 14 saline-based wet-contact resistive electrodes fixed to flexible plastic arms of a wireless headset. Subjects only need to drop some saline solution (contact lenses protection solution) on the sponge in electrodes on the headset and directly wear it. Additionally, the EPOC headset also has a 2-axis gyroscope for measuring the head rotation. It can operate up to 12 hours long using a rechargeable polymer battery.

Heart rate Variability and Galvanic Skin Response are physiological indicators that also gives us some direction into the cognitive load of the individual using this system. The Shimmmer GSR3+ device that is being used for MAMEM can measure the GSR and the Heart Rate Variability.

Taking all these concepts together, GazeTheWeb framework has been developed for MAMEM, which now uses the eye and the brain signals to navigate and select across the information accessed on this platform. The primary navigation and selection is by eye tracking and the BCI deivice has now been added as a separate modality where selection of menu can be done by just capturing 5 seconds of EEG signal when one looks at the varried frequency of flickering of the menu tabs. On GazeTheWeb, all the modalities that are mentioned in this section are being researched and investigated for effortless integration. Signal processing of EEG signals have been taken into account and are looked into to understand the best applicability of such a high dimensional data generated from the BCI device. EEG signals are also being investigated to understand how relevance feedback can be implemented in the framework in order to provide better assistive search and navigation on GazeTheWeb. Signals from the eye have been filtered and are currently in use to select the most appropriate keys based on the distance of the point from the center of the key in focus. Algorithms that take in heart rate data from sensor and convert it into HRV has also been investigated.

All these individual modalities will be used in unision to develop products that can be used by patients.

3 Eye and EEG Driven Control Paradigm

MAMEM's main objective is to provide easy interaction control to users so as to make using the multimedia content as easy as possible even with motor impairment. Gaze or eye tracking modality not only gives us an idea over where our current visual attention is directed at but it also often precedes action, meaning we look at things before acting on them. But using the eye as an input has a lot of challenges as faced in MAMEM, the major being the "Midas Touch" issue. In this chapter we will focus on MAMEM's control paradigms involving eye tracking, BCI and GSR devices and investigate how, GazeTheWeb famework can be improved with SSVEP / SMR / ErrP and GSR while understanding their inherent limitations.

3.1 Gaze Input Modality

The eye tracker provides a natural way to interact and assist physically disabled people to communicate with computers. For this reason, we chose the analysis and realization of gaze input signals as the primary interaction modality between the user and the application platform GazeTheWeb. Our approach is to acquire a users eye gaze information in real-time by an eye tracking device that can be used to generate gaze events, and to analyze the data to deduce more high-level events. In this direction, we have presented a novel browser: (MAMEM deliverable 3.1[1]) GazeTheWeb browser which consists of a communication path between the eye tracker and the application interface.

3.1.1 GazeTheWeb

A major set back or problem faced by people with physical disability or motor impairment is the accessibility to the world of the web. Currently, systems are tuned to be used with multiple modalities that mainly involve motor control and visual control over projection of information from the internet or social media. Most of the information sources or social media intended for communication has not been designed keeping in mind the problem of this section of the population. And thereby, using devices that act as inputs for these environments like eye tracking, BCI devices, etc are not being used to their full potential. This is where GazeTheWeb framework comes into play. This framework gives the users to access the web by converting the modalities of hand and eye movements to only eye movement and command via eye tracking system.

GazeTheWeb browser is a web surfing system for adapted interface and functionality for eye tracking based input signals. In GazeTheWeb, the input events (which are typically composed of mouse and keyboard interactions in generic browsers) are revised to eye gestures. The framework is not only made to work for eye movements but has been designed in a specific way to be used by eye based interaction. This is because user interface components designed specifically for this purpose enhances the accuracy of eye tracking systems. Major functionalities of GazeTheWeb are selection, scrolling, link navigation, typing, etc. In addition to control paradigms, GazeTheWeb browser provides an overlay of self-explanatory buttons that goes beyond the conventional menu-based functionality to assist gaze-based interactions.

Efficient generation of gaze events depends on several eye-tracking parameters like saccade, micro saccade, fixation, latency, etc. Moreover gaze based interaction suffers from the "Midas Touch" problem which refers to the distinction between monitoring the environment with the eyes or using them to activate commands. Until now, we have proposed to mitigate these problems via accumulated dwell time selection, task specific threshold, object size, position, and several other interface optimization algorithms for eye tracking signals (we have conduct studies to analyze different types of selection methods with eye tracking, also taking target size

[1] http://mklab.iti.gr/mamem/images/b/bb/D3.1_BCI_Interaction_final.pdf

into account). However, in this deliverable our focus is to resolve these problems by combining gaze input with additional input sources like EEG and biosensor.

3.1.2 Limitations

GazeTheWeb has been improved technologically and with respect to design and workflows, in order to allow users suffering from motor disabilities a seamless transition from existing systems. However, there is a need of further optimization, and in that regard we present the usability parameters as following.

- Dwell time: The system is highly dependent on dwell time based selection (predefined gaze duration threshold to activate a command). All major activities (select, click, scroll, type etc.) are directly or indirectly related to the dwell time duration of users gaze. Hence the interaction experience could be tiring for expert users, and time-consuming for some real world tasks. We currently tackle this problem by providing optional key based selection for faster interaction, i.e., the user can look at the desired target in the screen and press a pre-defined hotkey (Enter key) for the desired action. In future we plan to replace the hotkey kind of selection with other modes such as brain command, head movement, body gestures, etc.

- Midas Touch: Another limitation of dwell time selection strategy is the "Midas Touch" issue of unintentional selection, e.g., clicking links while viewing the web page content. Currently we resolve this issue by providing mode switch buttons on GazeTheWeb layout. However in future we aim to build more natural interaction scheme to characterize users intention, hence the signals from other modalities would be very significant.

- Eye Tracker Accuracy: Even though with the evolution of advanced eye tracking devices and gaze tracking techniques, the eye trackers still suffer from the problem of eye drift, inherent jitters and calibration accuracy. Currently we resolve these problems with smoothing and filtering the gaze coordinates. However interaction with small targets still suffers from frequent errors, e.g., while typing with GazeTheWeb keyboard, user might select neighboring characters. Currently we try to resolve this problem with magnifying the attention area to make easier selections. However to tackle the issue of erroneous entries, we are working on error related potential (ErrP) of EEG signals.

- Rest the Eye: One major feedback from experiments have been the resting of the eye. Due to the Midas touch issue, where ever one looks acts as an input command to the system. Currently we try to optimize interface space to provide appropriate areas of the screen. However in future we aim to measure the cognitive workload of user (e.g. it can be inferred through the stress level detected through the GSR sensors), and possibly have a switch that goes on or off based on the cognitive load or when user needs to take rest.

- Head Movement: A common problem is the head movement post calibration. This throws the sensors out of its calibrated state for that particular user. A more robust approach is being investigated to solve this problem of head movement.

- Visual Search Cognition: Experiments with "GazeTheWeb-Twitter" as mentioned in the 3.1 deliverable shows us that people become so focused in searching and selecting letters from the virtual keyboard that they do not pay heed to the word suggestions that come up on the system. Currently, we are trying to overcome this limitation by highlight not only the words that are coming as suggestions in order to attract visual attention but also incorporate letter highlight and word prediction on letters, so it can help in reduction of key selections and there by increase the word speed.

  In this regard, Figure 2 shows the interface of newly designed keyboard that allows dwell time threshold to select the suggested word completely from the key there by reducing the

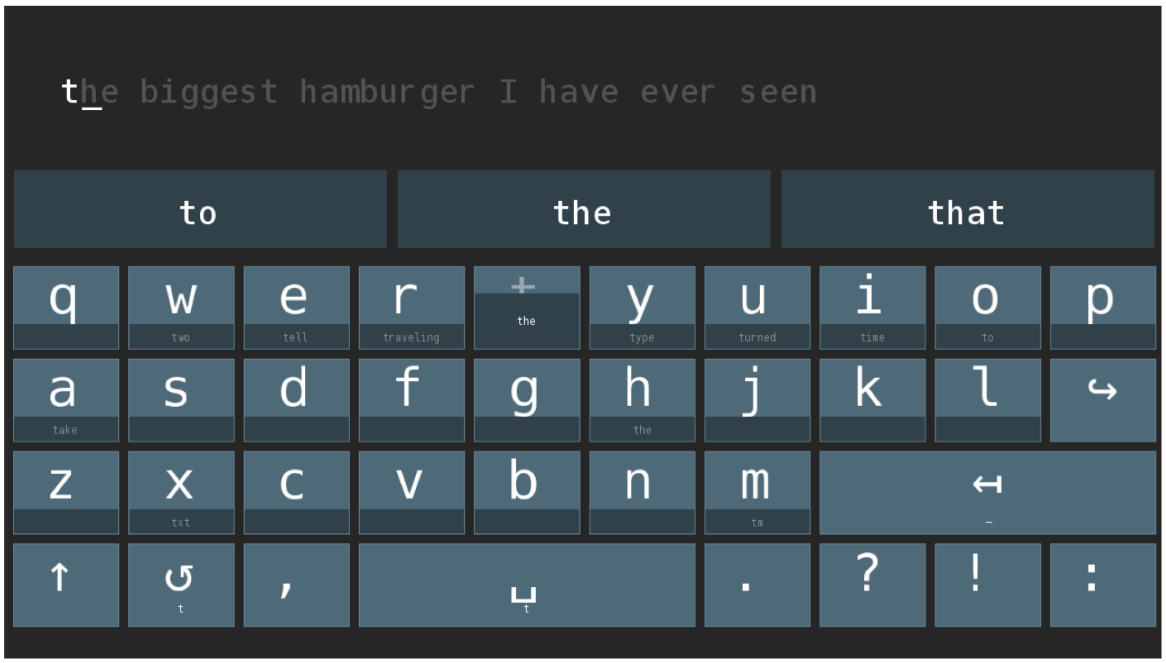


Figure 2: A new keyboard to improve auto text suggestions

time of visual search and selection of word suggestions. The new keyboard not only helps us in seeing what we have typed, but also what could become of the letter that we are typing to make it into a word. Another change that has been brought in the keyboard is to bring in key highlighting but with respect to predicted words, where in the next letter of the predicted words are highlighted on the keyboard. This simple highlighting of next letter brings in attention on the keys quicker. Word suggestion on the keyboard helps the user in not only looking at what he has already typed on that very letter he/she is focusing now, but also a view of the complete word prediction that may be the word that needs to be selected. In future we plan to perform an experiment to analyze if the new design reduces the cognitive load and improves the performance of end users.

3.2 BCI Input Modality

3.2.1 SSVEPs - potentials and limitations

Steady-State-Visual-Evoked-Potentials (SSVEPs) are the changes of EEG activity triggered by visual stimuli and time-locked to the stimulus. The visual stimulus is typically a flickering box with high frequency rate (>6Hz). More specifically, if a series of identical stimuli are presented at high frequency (e.g., 8 Hz), the EEG signals of the recipient will enter into a steady state, in which the visual system resonates at the stimulus frequency. In other words, when the human eye is excited by a visual stimulus, the primary visual cortex in occipital brain regions generates electrical activity at the same (or multiples of) frequency of the visual stimulus.

SSVEPs constitute a basic building block of Brain Computer Interfaces (BCIs) by enabling the user to select among several "flickering boxes", each one corresponding to different commands, letters, options, etc (e.g. selecting letters in order to type a word). Each command is associated with a repetitive visual stimulus that has distinctive properties (e.g., frequency). The stimuli are simultaneously presented to the user who selects a command by focusing their attention on the corresponding stimulus. When the user focuses their attention on the stimulus, a visible SSVEP oscillating at the target frequency (and its harmonics) is produced. In general, SSVEP-based BCI systems have the advantage of achieving high accuracy. For example, our experimental study in D3.1 has shown that several subjects can achieve $\sim 100\%$ accuracy. In addition, short/no training time and few EEG channels are required. On the other hand, view-

ing flickering boxes can be tiring and disturbing to the user. Finally, SSVEP-based BCIs are usually designed with only a few options (e.g. 5 or 6), since it is difficult to distinguish between frequencies that do not differ significantly. Thus, implementing a full keyboard ($\sim$ 30 options) with SSVEPs is not straightforward and the typical choice is to implement it in layers (i.e. in the first layer select the box containing the desired letter among many and in the next layer select the letter among the ones contained in the box). This, however, increases significantly the complexity of the system.

3.2.2 SMR - potentials and limitations

Sensorimotor Rhythms (SMR) are brain waves that appear on EEG recordings from areas of the brain that are associated with planning, control, and execution of voluntary movements. When a user is in a resting state (i.e. no movement is occurring) the neurons in these motor areas generate synchronized electrical activity resulting in high amplitudes of EEG recordings. Movement or even preparation and imagination of a movement triggers an event that desynchronizes the electrical activity of the neurons in the motor areas resulting in reduced amplitude or power of a specific frequency band in the EEG. When the state of the user is reversed to the idle state, the electrical activity of the neurons are again synchronized. The detection of the desynchronization period can be associated with a command for a BCI application. For example, the system could associate the movement of the right or left hand (or the imagination of this movement) with a command that selects right and left. According to the 10-20 system brain activity produced by left and right hand movement is most prominent over locations C4 (right motor region) and C3 (left motor region), respectively, while foot movement invokes activity on the central electrode (Cz). Other type of movements such as the movement of individual fingers are not possible to be detected because the involved brain areas are not large enough compared to the hand and foot movement scenarios.

The main advantage of SMR-based BCIs is that they do not depend on external stimuli for invoking brain patterns (e.g. as in the case of SSVEPs). This provides more freedom to the user since the system is continuously available and they can decide freely when they wish to generate a control signal. In addition, this method does not require any special considerations for the interface design. On the other hand, it is much more complicated to implement than a stimulus-based BCI. These systems must continuously be able to detect whether the user is on an idle state or wishes to execute a command. This results in many false positives, meaning that a lot of control commands are passed on the BCI system unintentionally. Furthermore, for now, the detection of each command typically requires $\sim$ 3 seconds after the action onset, which makes their usage for regularly used commands impractical. When the detection of SMR can achieve high performance in lower times, it could be used as a virtual click for each command that the user wants to activate. This can be particularly useful in a MMI, where gaze is used to define the location and EEG through SMR is used to activate a click. A key limitation is the amount of training required by the user to be finally able to use the system. While, stimulus-based BCIs require little to no amount of training since most of the users can learn the simple task of focusing on a target letter or symbol within a few minutes, SMR-based BCIs highly depend on how well users can produce signals that are easily detected. Thus, such systems typically include training sessions for the user, so that they can produce appropriate signals.

3.2.3 ErrPs - potentials and limitations

Error related potentials (ErrPs) constitute characteristic EEG signals observed after subjects committing errors during various tasks. Different types of errors can evoke ErrP-type signals, e.g. depending on who is responsible for the error (the user or the interface) or on how the error is perceived by the user (e.g. as erroneous feedback to an action or as an observation of an error made by an independent operator). Through ErrPs, the EEG sensor offers a natural

way to correct the errors occurring in a Natural User Interface (NUI) by the responsibility of either the user or the interface itself. In this way, functions like "undo" or "backspace" can be automated leading to enhanced user experience and faster interaction.

The main advantage of ErrPs is that they can be detected in very short time (typically $\sim 200 - 800ms$ from the occurrence of an error). This allows for a very efficient auto correction system. However, the error detection suffers from relatively low accuracy (as shown in D3.1, the accuracy was at best around 70% for a moving cursor experiment). While we plan to investigate on the optimal experimental setup and interface design that can achieve higher detection rates, it is worth noting that accuracy is not a fitting metric for the error detection problem. More specifically, the major concern in an automatic correction system is the false positive rate (i.e. how many correct moves were detected as erroneous and were incorrectly reversed thereafter). This can be partially solved by adjusting the thresholds of the algorithm that decide when an error occurs. However, this comes at the expense of lower recall, since a stricter algorithm will yield less true positives as well.

3.3 Stress Level Modality

Galvanic skin response (GSR), also known as electrodermal activity (EDA), is the property of the human body that causes continuous variation in the electrical characteristics of the skin. GSR is also a good measure to indicate stress. In general, sympathetic activation increases when someone experiences excitement, or something important is happening or about to happen. It also increases with stressors - whether physical, emotional, or cognitive. However, apart from such stimuli and emotion states that have an impact on skin conductance values, there are also external factors which influence GSR values, such as ambient temperature.

Skin conductance is characterized by great variability among dierent individuals, as it depends on dierent factors including age, gender, ethnicity, and hormonal cycles. Thus, comparing absolute skin conductance levels really makes sense only when the measurements originate from the same individual. In order to overcome the individual variability problem, our developed algorithm establishes a personal baseline by observing the skin conductance levels for each individual for a specic amount of time. This baseline stores the typical skin conductance values for an individual,and how often they occur. Afterwards, it is used to create stress level estimates, which help to classify the level of arousal to high or low, while it is tuned for each individual. More details on the algorithm for stress-level detection have been presented in deliverable D3.1.

With respect to MAMEM's MMI the acquisition of stress level will be exploited in two ways: a) as a direct feedback to the persuasion design mechanisms that has the responsibility to tailor the interaction experience with the particularities of the user, b) as a parallel evaluator of the system that could, for instance, provide an additional source for detecting and verifying an error, next to the ErrPs.

4 Multimodal Interfaces

In the previous chapter we described the functionalists and limitations of the current version of MAMEM platform. However, optimizing gaze with other modalities is an important pathway for multimodal environment. Therefore, in this chapter we present our approach, design and preliminary experimental procedure for multi modal optimization.

4.1 Control Paradigm With Multimodal Input

4.1.1 Optimizing Gaze Control with Sensorimotor Rhythm Input

As discussed before, the "Midas Touch" comes across as a major drawback in gaze based interaction, which is particularly true for sophisticated tasks like web browsing and navigation. For example, in GazeTheWeb framework, users are able to read and navigate the web pages with their eye movements, however several pages on Web pose extensively rich hyperlink structure (Figure 3), and these links can be unintentionally activated, even though the user intends to read through the page. To resolve such problems we currently offer a switch button (e.g., to activate the hyperlink navigation mode when the user intends to click on links). Such an external switch button (which needs additional gaze input) to control the Web navigation environment interrupts the users natural interaction with the Web and works as an overhead. Therefore, we aim to explore the usability of brain waves in supporting the eye-based interaction with Web.

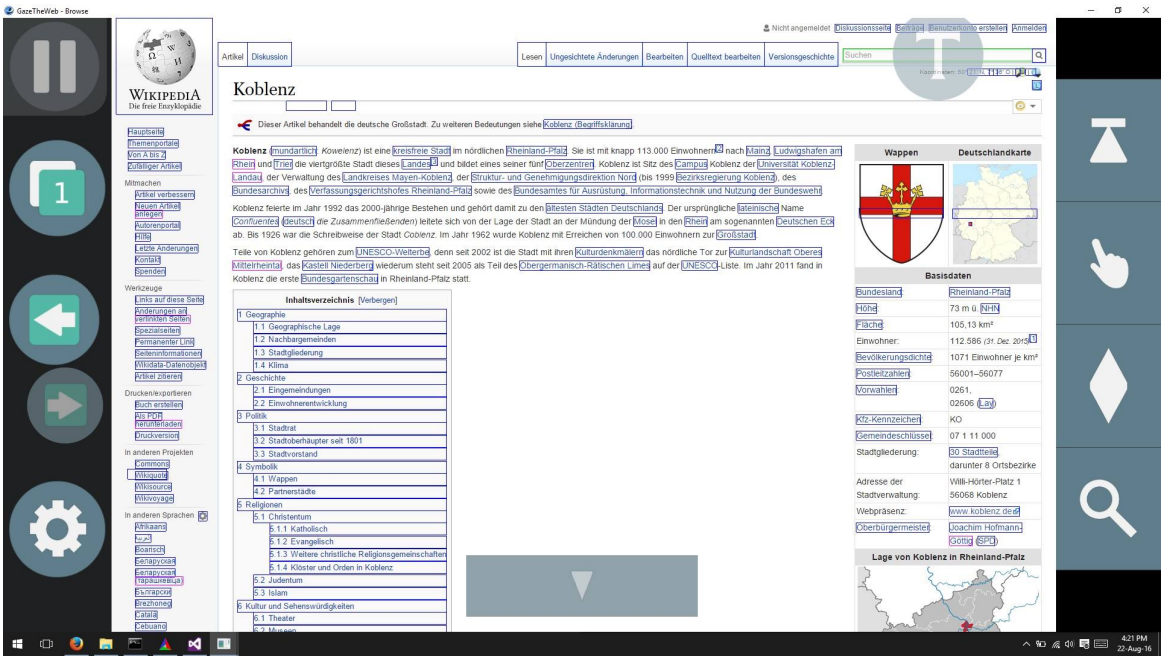


Figure 3: GazeTheWeb Browser: Highlighted Hyperlinks on a web page

SMR are waves of the brain accessed from the sensoricortex that is involved for planning, control and execution of voluntary movements (see MAMEM deliverable D3.1: Section 3.2 for more detail). We aim to exploit these control signals to trigger binary events. Binary events can be a simple virtual switch that changes the reading mode to selection mode or vice versa. Accessibility and training for SMRs could speed up the process of switching modes there by reducing the problem of selecting any link by looking at them and in the process reduce the problem of Midas Touch. If SMRs can provide us a solution with its clear EEG signals for execution of voluntary movements, this can provide us multiple ways in which this switching technique can be applied and used in multimodal systems. Moreover, other bio signals can be used in conjunction with SMR to utilize the complete dimension of multimodal interaction of

the system. Simulation of clicking to activate modes can be done by detecting movements in a particular axis and system can be trained to use that signal as a switching signal like SMR.

4.1.2 Optimizing SMRs detection

In this section we describe the actions that we have performed in order to achieve a suitable configuration of a MI BCI system for the MAMEM MultiModal Interface (MMI) with respect to findings and results reported in the D3.1. At first, a short description of the typical experimental protocol for MI BCI is provided. Then a software platform related to BCI, the OpenVibe platform, is shortly described. This choice would provide us with a quick and reliable way to perform motor imaginary experiments. After that, we provide information related to the data collection procedure for our dataset. To enhance furthermore this study we also include one well known dataset in the BCI community, the Gratz dataset B [20]. Results on the two datasets are provided using various algorithms described in D3.1. The objective is to provide insights related to algorithmic aspects, such as the system training procedure, the EEG signal acquisition process, the required channels, the user training procedure and the optimal mental strategy. The above insights will help us to optimize the integration of sensorimotor rhythms into the overall framework of the MAMEM MMI, and hence, they will enhance our ability to extend the functionality of GazeTheWeb in order to handle binary events (i.e. mode selection) using brain patterns.

4.1.2.1 Typical experimental protocol for Motor Imaginary (MI) BCI

The MI BCI experimental protocol consists of two basic steps. At the first step, the EEG data are acquired without showing any visual feedback to the user with respect to how the system classified their thinking (screening step), while, at the second step feedback is incorporated into the overall procedure (feedback step). In both steps many EEG trials are acquired for further processing. The trials acquired during the first step are gathered in ordered to train a classifier, while the ones from the second step are related to the training of the user. It is worth noting that without the feedback the user has no information about the decision process and how to adapt to the system. On the other hand, by providing feedback to the user we show him a way to adapt to the system by regulating and optimizing their mental strategy.

At the screening step, the subject tries to imagine the execution of a movement from a body part. Each trial starts with a fixation cross. Some seconds later a visual cue (an arrow pointing either to the left or right, according to the requested class) is presented. Afterwards the subjects have to imagine the corresponding hand movement over a period of time, usually 4 to 6 seconds (imaginary period). Each trial is followed by a short break. Also, a randomized short period is added to the break in order to avoid subjects adaptation. At the feedback step, feedback is provided to the subject during the imaginary period with respect to the classification of their EEG signals. Various types of feedback could be used at this stage (e.g. a bar, a smiley, etc.). Depending on the cue, the subject tries to move accordingly the feedback (usually towards the left or right side) by imagining left or right hand movements, respectively.

4.1.2.2 Data Collection Procedure

OpenVibe is a free and open-source software platform for the design, test and use of Brain-Computer Interfaces. The platform consists of a set of software modules that can be easily and efficiently integrated to design BCI applications. The platform contains some important features such as modularity, reusability, portability and support for different types of users [29].

Designing and operating a BCI experiment with OpenVibe platform requires three distinct steps. In the first step, a training dataset must be recorded for a given subject, while they perform specific mental tasks. The second step consists of an offline analysis of these records with the goal of finding the best calibration parameters (e.g. optimal features, relevant channels, etc.) for this subject. The last step consists of using the BCI online in a closed loop process.

Optionally, iterations can be done on data acquisition and offline training in order to refine the parameters. The online loop (third step) is common to any BCI and it is composed of six phases: brain activity measurements, preprocessing, feature extraction, classification, translation into a command and feedback. Existing and pre-configured ready-to-use scenarios are proposed to assist the user of OpenVibe platform. Currently, five complete scenarios are available related to BCI research; hand motor imagery based BCI, self-paced BCI based on foot movements, neurofeedback, real-time visualization of brain activity in 2D/3D, P300-speller. In this study, we used the hand motor imagery based BCI scenario, which allows to use OpenVibe as an interaction peripheral using imagined movements of the left and right hand. This scenario is inspired from the well-known Graz-BCI experimental protocol of the Graz University and in our study we use this scenario to collect the EEG data.

All the experiments were designed using the OpenVibe platform as the stimulation source and feedback initiation. Each session started with a ten second window during which the subject was not engaged to any activity. Then the recording of the trials started. During each trial, a left or right arrow appeared on the screen in a random way as a cue motivating for a left or right movement respectively. In the sessions without the feedback the arrow remained in the screen for the whole length of the trial (6 seconds), whereas in the ones with the feedback the arrow appeared only for 1 second and during the next five (seconds) the feedback bar showing graphically the output of the classifier would appear. Between the trials, 3 seconds of resting time was provided. The experiments included trials for imagery movements, with and without feedback. More specifically, each subject took part into an experiment consisting of 2 sessions. Each session was held in a different day and included 40 trials, 20 of left and 20 of right movement. At the first session, the subjects were asked to imagine a left or right hand movement without moving their hands at all, while the second session also included a feedback.

4.1.2.3 Description of Datasets

Gratz dataset B This dataset consists of EEG signals from 9 subjects. The subjects were right-handed and had normal or corrected-to-normal vision. The subjects were sitting in an armchair, watching at the screen monitor placed approximately 1m away from the screen at eye level. For each subject 5 sessions are provided, where the first two sessions contain training data without feedback (screening step), and the last three sessions are with feedback (feedback step). Three bipolar recordings (C3, Cz, and C4) with a sampling frequency of 250 Hz are provided. They are bandpass-filtered between 0.5 Hz and 100 Hz, and a notch filter at 50 Hz is enabled. The electrode Fz served as EEG ground. Further information on this dataset can be found in [20].

Our dataset This data set consists of EEG signals from 3 subjects acquired with the EbNeuro cap (64 channels based on the 10-10 international EEG system with a sampling frequency of 128 Hz). The subjects were right-handed and had normal or corrected-to-normal vision. The subjects were sitting in an armchair, watching at the screen monitor placed approximately 0.6m away at eye level. For each subject two sessions were recorded, where the first session contains training data without feedback (screening step), and the second session was recorded with feedback.

4.1.2.4 Data analysis protocol

The time segment of 0.5 - 2.5s after the onset of the visual cue was used to train the algorithms. It is the same procedure as the one used in [2]. To evaluate the algorithms a sliding window of 2 secs, from the beginning of the corresponding trial until the end of it, is used. A continuous classification output for each sample in the form of class labels (1, 2) is provided by each algorithm. A confusion matrix was built from all trials for each time point. From these confusion matrices, the time course of the accuracy as well as the kappa coefficient can be obtained. For each algorithm we provide the largest (or peak) kappa value and accuracy.

Results on Gratz dataset B The Gratz dataset B is composed by 5 sessions, 2 sessions have been created without the feedback procedure and the last 3 sessions with the feedback. To evaluate the various algorithms the following division between the sessions has been adopted: the first 3 sessions have been used for the training (2 sessions without feedback and the 1st session of using feedback), while the rest 2 sessions (with feedback) are used to evaluate the various algorithms. For the training and evaluation procedure, the EEG data from channels C3 and C4 are used. First, a band-pass filter from 8 to 40 Hz has been applied to the signals. Then, feature extraction and classification is applied using various combinations of features and classifiers (details on the algorithms can be found in D3.1). In Tables 1 and 2, we show the results by using our algorithms as well as results reported in [2]. In all cases an identical procedure for data analysis is used (i.e. filtering, trial segmentation etc.), the difference is on the feature extraction and machine learning algorithms. By observing the results on Tables 1 and 2 we can see that there is not an algorithm that performs best on all subjects. Overall, the algorithm reported in [2] provides the best mean kappa value (0.59), while the PWelch approach in conjunction with linear SVMs gives us the second best value (0.58). By applying a paired t-test (using ttest matlab function) we observe no significant statistical difference between the two methods (p=0.5313). Similar observations can be extracted by looking at the classification results with respect to accuracy (see Table 2).

Table 1: Classification results with respect to kappa value on unseen evaluation data on dataset Gratz B

	CSP [2]	FBCSP MIRSR [2]	PWelch lin. SVM	PWelch RBF SVM	PWelch MLR	DWT lin. SVM	DWT RBF SVM	DWT MLR	AR lin. SVM	AR RBF SVM	AR MLR
1	0.3190	0.4000	0.4625	0.3375	0.4312	0.3375	0.3000	0.3625	0.4063	0.3125	0.4000
2	0.2290	0.2070	0.2357	0.2571	0.2500	0.2643	0.2214	0.2143	0.2714	0.2500	0.1929
3	0.1250	0.2190	0.1687	0.2188	0.2437	0.2000	0.1938	0.1875	0.1563	0.1500	0.1563
4	0.9250	0.9500	0.8750	0.9500	0.8750	0.9312	0.8938	0.9000	0.9375	0.9312	0.9187
5	0.5250	0.8560	0.7500	0.7063	0.8062	0.5875	0.5500	0.5688	0.7688	0.6938	0.7312
6	0.5000	0.6130	0.6563	0.5625	0.6125	0.6687	0.6187	0.7000	0.6563	0.5938	0.5938
7	0.5440	0.5500	0.5375	0.5375	0.4875	0.4750	0.4937	0.4688	0.4625	0.5000	0.4563
8	0.8560	0.8500	0.8438	0.4500	0.8062	0.8500	0.6313	0.7937	0.7813	0.4937	0.7688
9	0.6560	0.7440	0.7500	0.7250	0.7000	0.7125	0.7125	0.6938	0.7250	0.7312	0.7125
Avg	0.5198	0.5987	0.5866	0.5271	0.5791	0.5585	0.5128	0.5432	0.5739	0.5173	0.5478

Results on our dataset Our dataset has been produced by adopting similar procedure with that of Gratz dataset B. Furthermore, we follow the same data analysis procedure as before. In the EEG data a notch filter at 50 Hz and a bandpass-filter between 0.5 Hz and 32 Hz were applied in order to remove electromagnetic noise and keep the spectral characteristics of the signals related to MI. The results with respect to accuracy are shown in Table 3 and the corresponding kappa values in Table 4. In our analysis, besides the pair of channels C3 and C4, we have also included the pair of channels FC3 and FC4, and the pair of channels CP3 and CP4. All above channels are placed on the brain region of interest. Finally, to analyze our dataset we used the PWelch combined with a linear SVM and the PWelch with MLR since these two approaches seemed to provide the best results at the Gratz dataset B. By observing the results related to the accuracy we can see that, for all channel combinations, the PWelch/MLR approach provides us with the highest accuracy. Furthermore, the configuration of CP3-CP4 provides the best results for both approaches. Similar findings also occur from the kappa value results (Table 4.

Table 2: Classification results with respect to accuracy (%) on unseen evaluation data on dataset Gratz B

	CSP [2]	FBCSP MIRSR [2]	PWelch lin. SVM	PWelch RBF SVM	PWelch MLR	DWT lin. SVM	DWT RBF SVM	DWT MLR	AR lin. SVM	AR RBF SVM	AR MLR
1	65.95	70.00	73.12	66.87	71.56	66.87	65.00	68.12	70.31	65.62	70.00
2	61.45	60.35	61.78	62.85	62.50	63.21	61.07	60.71	63.57	62.50	59.64
3	56.25	60.95	58.43	60.94	62.18	60.00	59.69	59.37	57.81	57.50	57.81
4	96.25	97.50	93.75	97.50	93.75	96.56	94.69	95.00	96.87	96.56	95.93
5	76.25	92.80	87.50	85.31	90.31	79.37	77.50	78.44	88.44	84.69	86.56
6	75.00	80.65	82.81	78.12	80.62	83.43	80.93	85.00	82.81	79.69	79.69
7	77.20	77.50	76.87	76.87	74.37	73.75	74.68	73.44	73.12	75.00	72.81
8	92.80	92.50	92.19	72.50	90.31	92.50	81.56	89.68	89.06	74.68	88.44
9	82.80	87.20	87.50	86.25	85.00	85.62	85.62	84.69	86.25	86.56	85.62
Avg	75.99	79.93	79.33	76.35	78.95	77.92	75.64	77.16	78.69	75.86	77.39

Table 3: Classication results with respect to accuracy (%) on our dataset for different channels configuration

	C3-C4		FC3-FC4		CP3-CP4	
	PWelch lin. SVM	PWelch MLR	PWelch lin. SVM	PWelch MLR	PWelch lin. SVM	PWelch MLR
1	56.25	58.75	60.00	63.75	58.75	65.00
2	51.25	55.00	62.50	58.75	61.25	71.25
3	70.00	67.50	70.00	76.25	76.25	68.75
Avg	59.16	60.41	64.16	66.25	65.41	68.33

Next steps for achieving SoA performance It is worth noting that with our dataset we achieved a mean accuracy of 68.33%, while for the Gratz Dataset B we managed to achieve 10% more in terms of accuracy using the same algorithms. While others in the MI BCI literature also report similar performance with our experiments (e.g. 66% mean accuracy is reported in [14]), the work from [20] shows that there is potential to increase the performance significantly. The difference in the accuracy between the two datasets could be explained by taking into account various factors of a BCI experiment related to overall procedure. More specifically, in the Gratz dataset B preliminary tests have been used to choose the "optimal" mental strategy for each subject (i.e. the best imagined movement) while in our dataset no such tests have been performed. Furthermore, the feedback in Gratz dataset B was a cartoon which depicts a smiley face taking various face expressions related to the classification output while in our case the feedback was a bar showing the amplitude of classification output. Finally, the training of

Table 4: Classication results with respect to kappa value (%) on our dataset for different channels configuration

	C3-C4		FC3-FC4		CP3-CP4	
	PWelch lin. SVM	PWelch MLR	PWelch lin. SVM	PWelch MLR	PWelch lin. SVM	PWelch MLR
1	0.125	0.175	0.2	0.275	0.175	0.3
2	0.025	0.1	0.25	0.175	0.225	0.425
3	0.4	0.35	0.4	0.525	0.525	0.375
Avg	0.1833	0.2083	0.2833	0.325	0.3083	0.3666

the users to provide signals of high quality is a known prerequisite. For the Graz dataset a session with feedback was also used in the training of the classifier. This session served also as a training session for the users. On the other hand, our results are on prior untrained users. Thus in order to improve the performance the aforementioned factors should be considered.

First, we must see the BCI control as a learning skill, it has to be learned, refined and mastered by the BCI user. To achieve this goal the experimental protocol needs to be changed. The experimental protocol that we have used can be divided into two steps: (1) training of the system, which is the without-feedback step and (2) training of the user, which is the with-feedback step. To enhance further the above protocol our goals must be concentrated around two general directions. The first direction is related with the development of better classifiers, which has been studied in this section. The second direction is related with aspects concerning the training of user. Research on user training in MI-BCI focuses around two aspects: (1) the influence of the users profile on their MI-BCI control performance and (2) the enhancement of the communication between the user and the system by improving the training protocols and feedback.

More specifically we have identified the following three directions to focus on improving: 1. the instructions provided to the user at the beginning of the experiments, 2. the training tasks proposed to the participant to control the MI-BCI, and, 3. the feedback provided concerning the systems decision (i.e. about which task was recognized). The instructions to the user must expose the real goal of BCI training, i.e., to produce clear, specific and stable EEG patterns. It is needed to explain to the user what the feedback means, especially for non-intuitive feedback such as the classifier output. Also, the demonstration of successful BCI use and the demonstration of the BCI feedback during correct task performance is crucial to understand the user the overall procedure. The feedback must be clear and meaningful. In cases where a user has a poor initial performance, perhaps the use of a subject independent classifier is more appropriate. By showing to the user as feedback some relevant EEG features or a measure of quality of the mental imaginary tasks we show them more information about what was right or wrong to the produced EEG patterns. Finally, it is desirable to include a variety of training tasks to be performed, and not be restricted on the imagination of right/left hand movement since different users might have different needs. This can be achieved by using some preliminary tests to check the best imagined movement (or mental task) for each subject. Also, the gamification of the training environment could provides a more pleasant environment to the user motivating them for engaging more in the training process. The aforementioned aspects will be refined and incorporated in the training cycles of the users that will be released in D5.2.

4.2 Automatic Error Correction of Gaze Input with ErrPs

4.2.1 The problem and existing solutions

While eye tracking offers a natural and easy way to interact with a computer, it cannot be considered error-free. In this direction, the authors of [22][2] present an extensive study evaluating different aspects of eye tracking systems and comparing gaze to mouse as an input method. More specifically, the authors compared four different input methods: a) gaze-based pointing and dwell-based selection with long dwell time (ETL), b) gaze-based pointing and dwell-based selection with short dwell time (ETS), c) gaze-based pointing and clicking the space button for selection (ETK) and d) regular mouse input (Mouse). Their findings indicate that point the need for a more accurate and faster eye tracking system. Regarding the performance of each input method measured by throughput (a popular metric) in bits/sec, the input methods were ranked from highest to lowest performance as Mouse, ETK, ETS, ETL. Furthermore, underlining the comparison of the four input methods with respect to the error rate, we can see that while the Mouse input method yields a minor 2% error rate, the eye-tracking based

[2]http://www.yorku.ca/mack/cogain.html

methods are much more error prone with error rates ranging from 14.7% for the ETS method to $\sim$ 19% for ETK and ETL. These comparisons show the need of the eye-tracking systems for a correction mechanism that will also increase the performance of the gaze-based interfaces.

Towards this goal researchers have tried to identify the sources responsible for the errors. In this report, we group these sources in two main categories; first, the popular Midas Touch problem, discussed previously, which is caused by the "dual requirement" from the eyes to both observe the interface and activate commands on it. This problem is mainly approached by increasing the dwell time or using additional sources to detect the activation intention of the user (such as focal visual fixations) [10]. In a similar direction, as we have shown in the previous deliverable (D3.1), adjusting the interface design to fit the eye-tracking needs increases the comfort and interaction accuracy significantly. Second, the accuracy of the gaze estimation itself may decrease significantly due to head movements. The gaze estimation problem is tackled by either proposing new calibration methods that also take into account the position of the user [7] or by modeling the error and using the model to predict the error in gaze estimation and "inform" the interface for them [3].

Compared to what has been considered so far in the literature, in this report we propose to design an error-aware multi-modal interface. More specifically, first, the main interaction source will be the eyes, which will be responsible for both *pointing* by gaze estimation and *selecting* by fixating at the same point or area for some time (dwell method). For the Automatic Error Correction (AEC) mechanism we propose the utilization of the EEG sensor, which through the ErrPs offers a natural way to detect errors. To the best of our knowledge, ErrPs have been only used so far for AEC in BCIs (e.g. to correct misspellings in P300 [31] or cVEP-based spellers [32]) and not in combination with a gaze-based interface. At first, we will also focus on correcting misspellings by implementing the *backspace* function in the keyboard, which will be automatically activated when an ErrP signal is detected. For the baseline design we will opt for what has been considered in the brain-based spellers [31, 32]; i.e. a letter is selected based on the gaze fixation, then a preview of the selection is shown to the user and finally if an ErrP signal is detected the letter is rejected otherwise the letter is entered. In the following, we first investigate on the most appropriate signal processing algorithms as well as interface designs in order to maximize the error detection true positive rate and minimize the false positive rate (Section 4.2.2). Finally, we present a user study showing the benefits of AEC in the proposed error-aware MMI.

4.2.2 Optimizing the ErrP detection

The objective in this section is to maximize the ErrP detection rate by targeting two directions; first, to optimize the algorithmic framework by investigating different signal analysis algorithms (e.g. pre-processing, feature extraction and classification) and second, to optimize the interface design by exploring different ways to present the letter selection preview, the time required, etc. Towards the first direction, several choices have to be made. To start with, the interface and the experimental protocol should be defined. While our intention, as mentioned in the previous section, is to use ErrPs to correct the typing errors, we noticed that the users are not always recognizing the errors when made but sometimes they would type a few additional letters, then recognize the errors and end up backspacing multiple letters. This poses as an additional challenge for the interface of the keyboard so that it can facilitate time-locked ErrPs. For this reason and in order to remove the additional factor of knowing the exact time of the triggering event, we went for a more straightforward experimental protocol using SSVEPs, where the users would see the outcome of their choice in an instant manner. The goal of these experiments is to identify the necessary parameters and configuration that fits best to the designated devices for MAMEM's clinical trials (EBN cap and EPOC+). Towards the second direction, we investigate whether a preview of the selection is required to get more discriminant ErrPs signals.

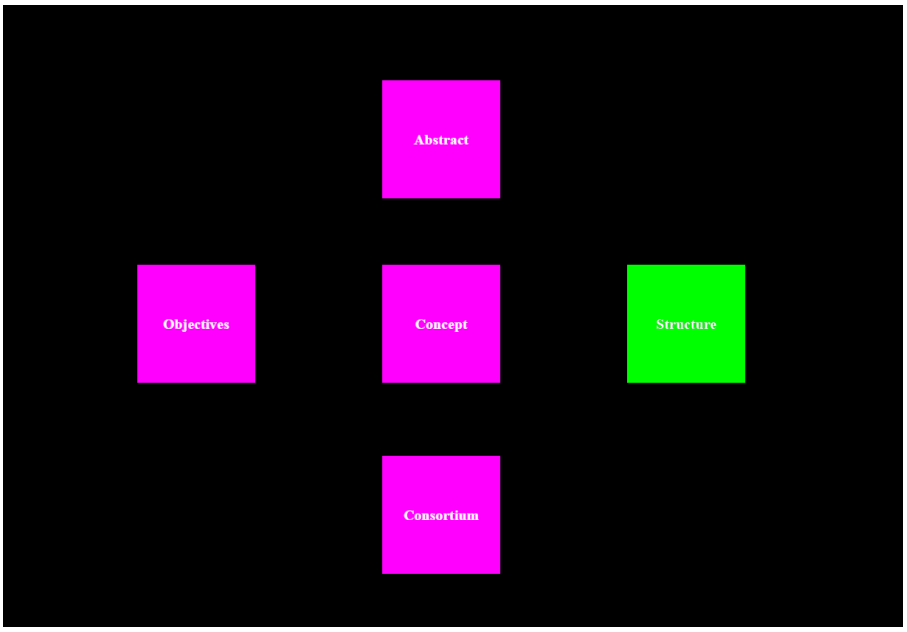


Figure 4: Selection preview for the SSVEP-based interface

4.2.2.1 Experimental protocol & dataset acquisition

Experimental protocol The experimental protocol relies on the SSVEP-based selection of 5 boxes in the context of the MAMEM site. The participants were asked to select one of the 5 magenta boxes flickering at different frequencies. The boxes would flicker for 5 seconds, then they would stop and a preview of the box that was selected by the system was shown for 2 seconds to the participant by turning the magenta color of the selected box to green 4. In the case that the previewed box was not the same as the one the participant was asked to observe, we would expect to create an ErrP in the recorded EEG signal a few ms (200-800) after the preview. Considering that some of the subjects were able to achieve close-to-100% accuracy on selecting boxes with SSVEPs and in order to maintain a similar ratio of correct and error trials for each participant, we opted for a random classifier to select the boxes. More specifically, a random generator would produce 70% correct classifications and 30% erroneous. The percentages were chosen based on the work from [13], who indicated that a lower correct trial probability can induce a stronger expectation of losing that helps to overcome the otherwise relatively strong optimism bias of the participants. Considering that unexpected errors lead to a larger ERN relative to expected errors, this would eventually increase the discrimination ability of the system between correct and error trials. The participants were not aware of the random classifier so as to get a natural response to the unexpected erroneous trials.

Dataset The same experimental protocol was deployed twice, capturing the signals with the EBN cap the first time (EBN dataset, 64 electrodes, 128Hz sampling rate) and with EPOC+ the second (EPOC dataset, 14 electrodes, 128Hz sampling range). 2 subjects participated in the study for the EBN dataset and 5 for the EPOC dataset, all male, right handed and between 26-37 years old. The following setting was identical for both datasets. Each participant did a total of 100 trials (20 times for each box, 5 boxes), out of which 70 were correct and 30 erroneous. After the start of the selection preview, which lasted 2 seconds, the EEG signal was recorded for these 2 seconds in order to acquire the potentials and then the system would redirect to the selected option so that the participant could move to the next trial. MAMEM's architecture was used to synchronize the signals with the events.

4.2.2.2 Visualizing the signals

At first, our objective is to visualize the signals per subject and see whether ErrPs can be discriminated from the correct-originating EEG signals. For this, the signals were band-pass

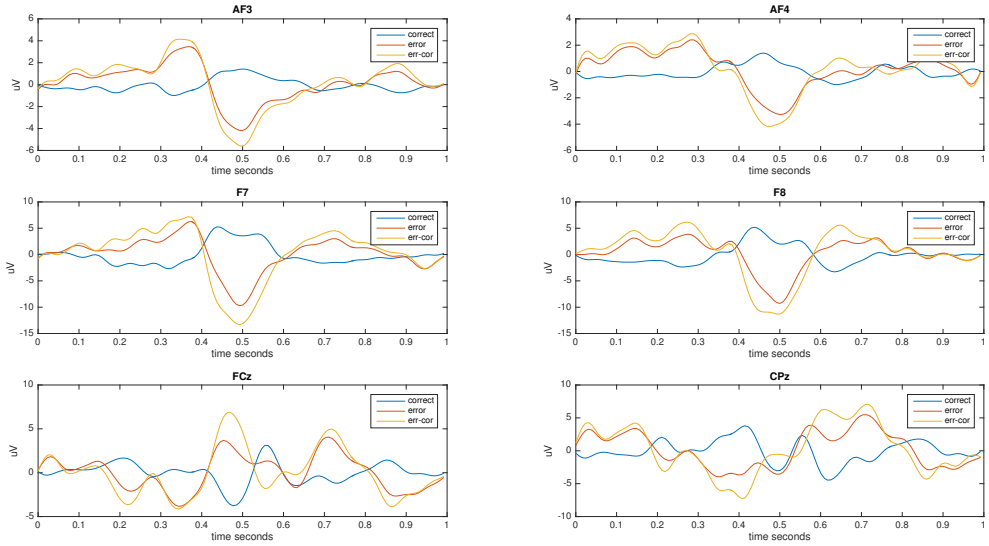


Figure 5: Subject 1 - EBN: ErrPs signal visualization

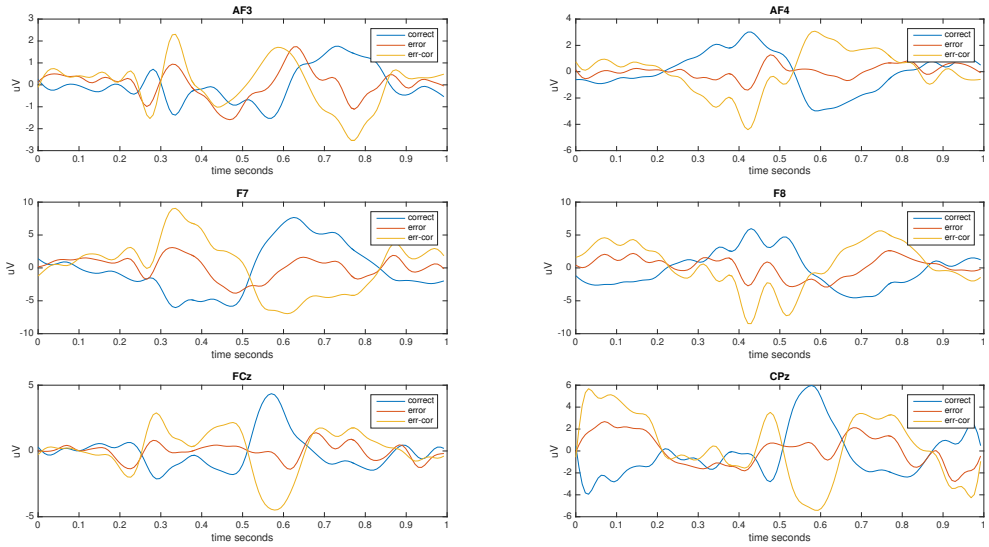


Figure 6: Subject 2 - EBN: ErrPs signal visualization

filtered (1-16Hz) using an IIR Butter filter and zero phase filtering to avoid distortions and latency. Finally, the average signal from the error trials, the correct trials and their difference were plotted for each subject. For the EBN dataset, consisting of 64 electrodes, we provide plots only for the 5 electrodes that showed visually the highest discrimination ability between error and correct trials. For the EPOC dataset plots for all 14 electrodes are provided.

The plots can be seen in Figures 5 and 6 for the EBN dataset and Figures 7 to 11 for the EPOC dataset. We can see that the electrodes AF3, AF4, F7 and F8 for both datasets and for all subjects provide the highest difference between error and correct trials, as it was also expected from the theory. For the EBN dataset the FCz electrode also shows significant potential but this electrode is not included in the EPOC headset. Moreover, subject 1 from the EBN dataset and subjects 1, 3 and 5 from the EPOC dataset show a bigger difference between error and correct trials compared to the rest. Also, the time the ErrPs occur differ from subject to subject, calling for a subject-specific classifier in order to detect the signals. Finally, we can see that in the case of the EPOC device the signals are quite noisy (the SNR is below one, showing that the noise is more powerful than the signal), so we need to consider artifact removal and additional preprocessing steps to clean the data prior to classification.

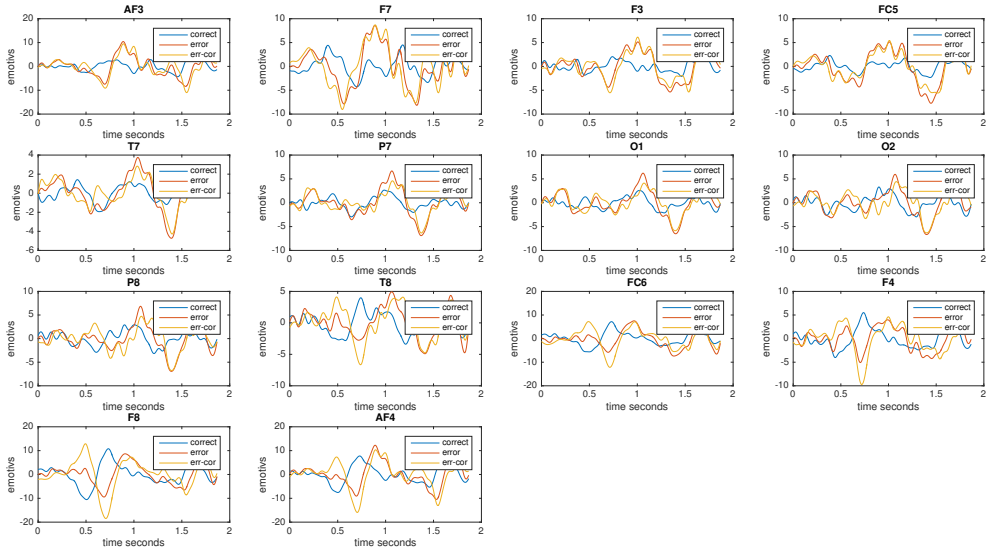


Figure 7: Subject 1 ErrPs signal visualization

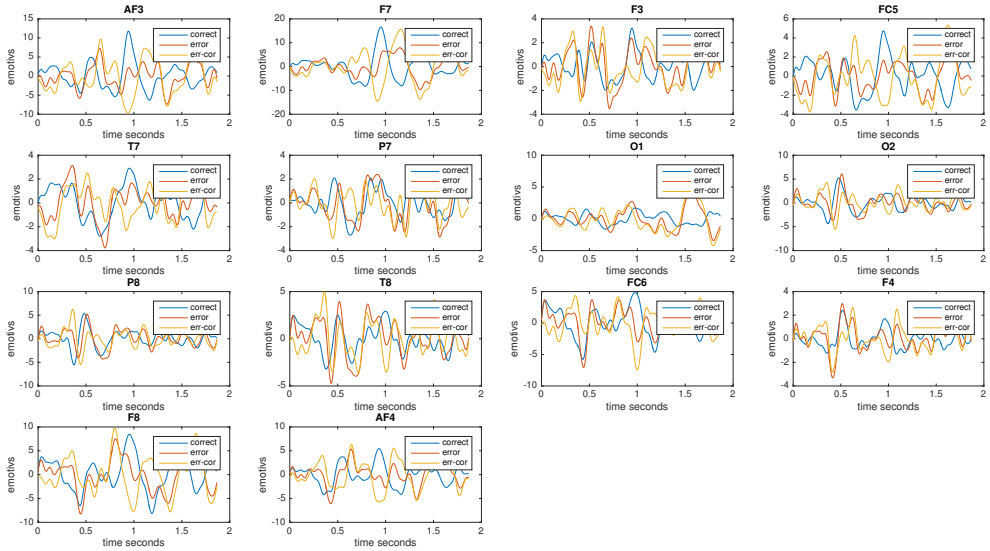


Figure 8: Subject 2 ErrPs signal visualization

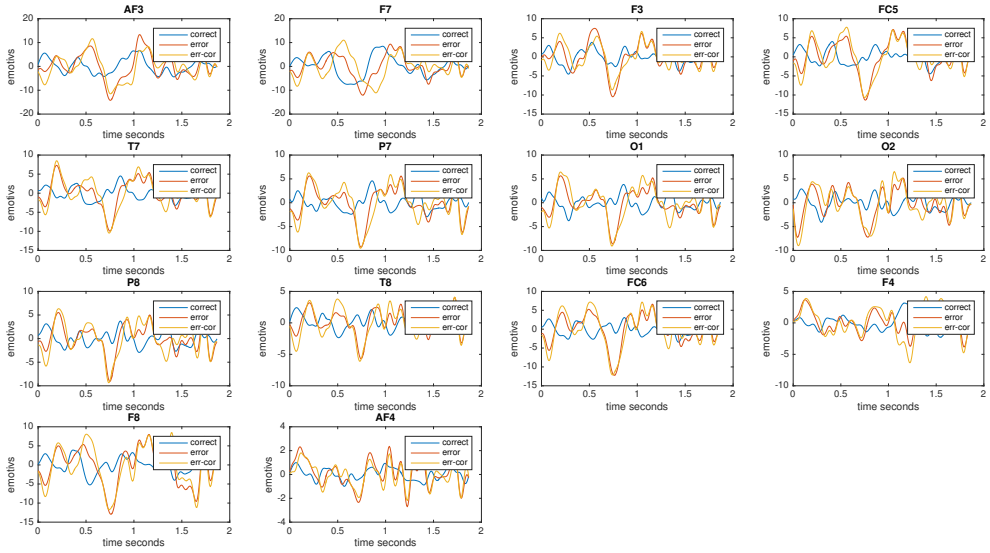


Figure 9: Subject 3 ErrPs signal visualization

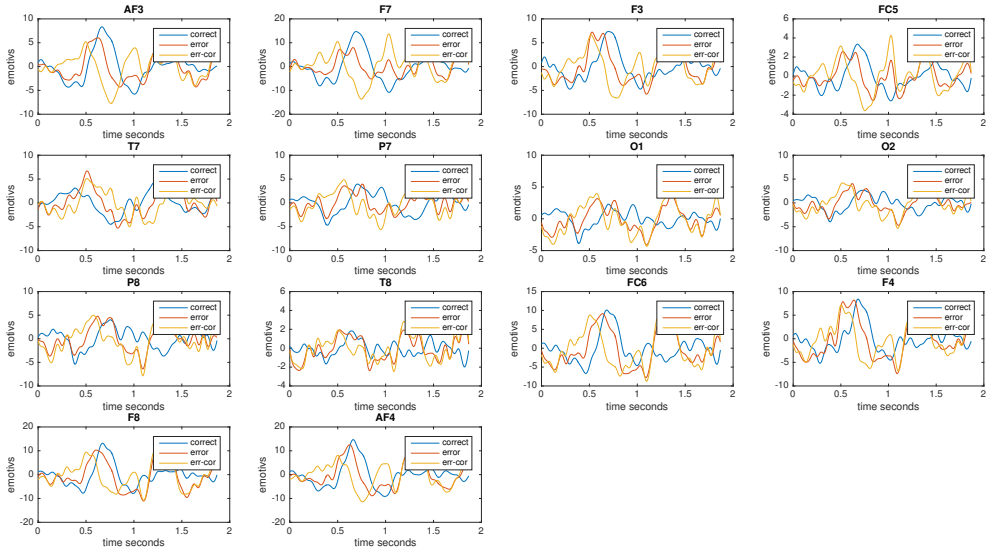


Figure 10: Subject 4 ErrPs signal visualization

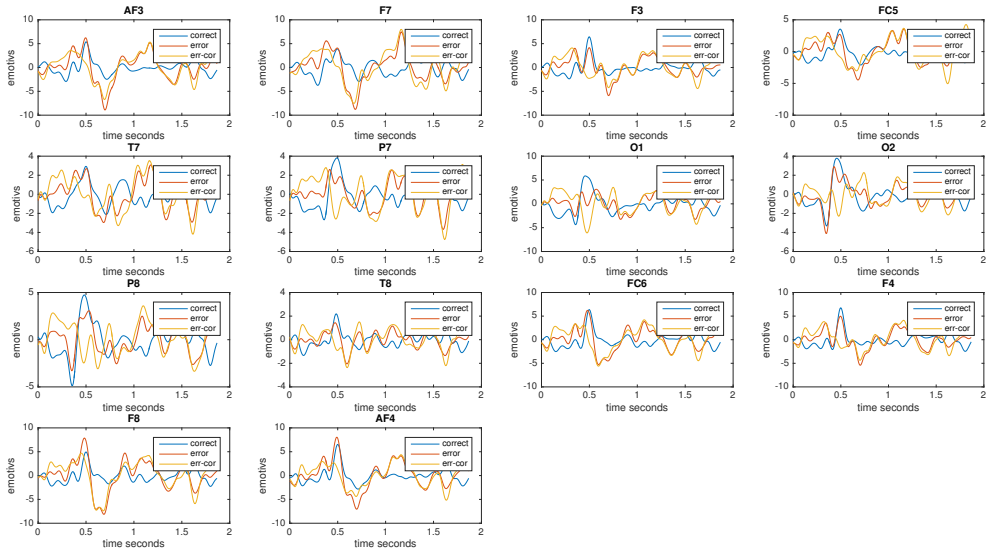


Figure 11: Subject 5 ErrPs signal visualization

Table 5: ErrP detection results - EBN dataset

	Accuracy			Precision/Recall		
	S01	S02	Avg	S01	S02	Avg
No Preproc	0.98	0.91	0.95	1/0.93	0.91/0.83	0.96/0.88
Windsor	0.98	0.86	0.92	1/0.93	0.83/0.77	0.91/0.85
AMUSE	0.88	0.84	0.86	0.86/0.77	0.73/0.7	0.79/0.73

4.2.2.3 ErrP detection

EBN dataset In order to verify that we can use the captured signals to detect ErrPs we have applied a set of algorithms to the data of each subject using 10-fold cross validation. While we have tested various algorithms for classification (e.g. naive threshold based classifier, SVMs with various kernels, etc.), we only report here the one that consistently provided better results (linear SVMs). In addition, we test two different algorithms for pre-processing (AMUSE and Windsor [9]). For feature extraction we have used the typical temporal features, i.e. sub-sampling the signal, keeping 1 sample every 4 (this ratio was enough since we filtered the signals at [1,16]Hz). Features from the 5 identified electrodes were extracted and then concatenated in a single vector, representing each trial. Moreover we report both accuracy and precision/recall metrics. The results are shown in Table 5. We can see that the ErrP detection rate in terms of all metrics is higher without any pre-processing, which was expected since the EBN capturing device provided clean signals as shown in the previous section as well. Finally, it is worth noting that the precision of ErrP detection is 100% for Subject 1 and 91% for subject 2, meaning that the system would return very few false positives, which shows the potential of using ErrPs for error correction in the MAMEM MMI.

EPOC dataset A similar procedure was used to analyze the signals of the EPOC dataset. The only difference is in the used electrodes, where in the case of EPOC there was no FCz electrode so the rest 4 were only used (i.e. AF3, AF4, F7 and F8). In addition to the two pre-processing algorithms used for the EBN dataset, here we also employed a different pre-processing step since the signals were shown to be noisy in the previous section. More specifically, we opted for an outlier detection process, i.e. detecting and removing observations that seem to differ a lot from the rest of the samples (e.g. observations that contain too much noise and the signal is contaminated). For this, we devised a heuristic method for detecting the outliers. First, the pairwise distance among all observations is computed. Then each observation

Table 6: ErrP detection results - EPOC dataset

	Accuracy						Precision/Recall					
	S01	S02	S03	S04	S05	Avg	S01	S02	S03	S04	S05	Avg
No pre-poc	0.66	0.57	0.65	0.65	0.64	0.63	0.55/ 0.57	0.24/ 0.3	0.29/ 0.43	0.45/ 0.57	0.43/ 0.52	0.36/ 0.52
Windsor	0.63	0.62	0.59	0.62	0.63	0.62	0.33/ 0.4	0.33/ 0.37	0.38/ 0.37	0.33/ 0.33	0.39/ 0.42	0.35/ 0.38
AMUSE	0.73	0.62	0.54	0.58	0.57	0.61	0.61/ 0.5	0.42/ 0.37	0.33/ 0.5	0.35/ 0.37	0.36/ 0.32	0.41/ 0.41
Outlier rem.	0.65	0.77	0.64	0.61	0.58	0.65	0.53/ 0.52	0.60/ 0.62	0.39/ 0.45	0.4/ 0.43	0.35/ 0.44	0.45/ 0.49

is assigned to a score, which is the sum of distance between this observation and its k nearest neighbors. Afterwards, the observations are sorted based on their assigned score. Finally, the samples with assigned scores is larger than a threshold are removed. The threshold value can be chosen arbitrarily or by exploiting the increase in the ranking values. In our case, the threshold was chosen arbitrarily by visually inspecting the signals and their distances. The results are shown in Table 6. We can see that outlier detection and removal, on average, increases slightly the error detection rate both in terms of accuracy and precision. However, the absolute numbers are quite low for an online auto correction system with this device. This can be mostly attributed to the noisy nature of the signals, as also shown visually in the previous section. Artifact removal algorithms like Windsor and AMUSE where unable to help since they mostly tend to remove artifacts in the context of eye blinks. In order to increase the performance we plan to follow two directions; first, we will explore a signal denoising/restoration path, investigating into pre-processing algorithms shown to clean signals from the EPOC+ device, such as the correction algorithm proposed in [8]. Finally, temporal features are probably not well suited for the EPOC+ signals, since they are prone to noise, which affect significantly the machine learning cost function. Towards this direction, we plan to explore additional and more sophisticated feature extraction algorithms that are not as much affected by the noise in the signals. Prominent alternatives include spectral features, brain connectivity patterns, activation asymmetry index (in order to explore differences in they way the 2 hemispheres are activated) and cross frequency coupling (functional interactions among distinct brain rhythms).

4.2.2.4 Interface optimization

The objective in this section is to explore the interface options so that it can evoke the highest ERN peaks and in this way allow for the best separation of the error trials from the correct ones. Towards this direction, we devised two experimental protocols with respect to the preview of the selection, capturing the signals with the EPOC+ device. The first was described in Section 4.2.2.1, while the second was identical with the only difference that there was no green box as a preview but the system would redirect to the selected page and this would trigger or not the ErrPs. In order to compare the two interfaces, we plot the average signals obtained by both protocols for each subject and for the 4 prominent channels identified in the previous section (i.e. AF3, AF4, F7 and F8). The visualizations are shown in Figures 12 to 16. As it is obvious, we can see that the green box preview (right figures) helped significantly in evoking the error-related peaks so that they can be more easily discriminated from the correct-related signals.

4.2.2.5 Next steps towards an AEC system

EBN cap experiments While we have reported preliminary experiments with the EBN capturing device, we plan to extend the experimental process by repeating the experiments with more subjects so that we can identify any possible shortcomings.

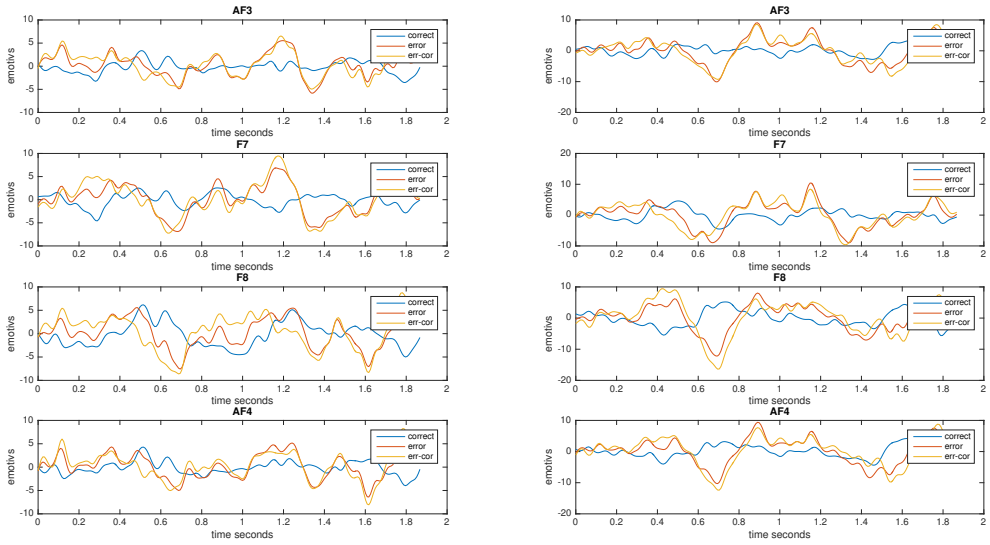


Figure 12: Subject 1 - EPOC: Simple redirect (left) vs Green box preview (right)

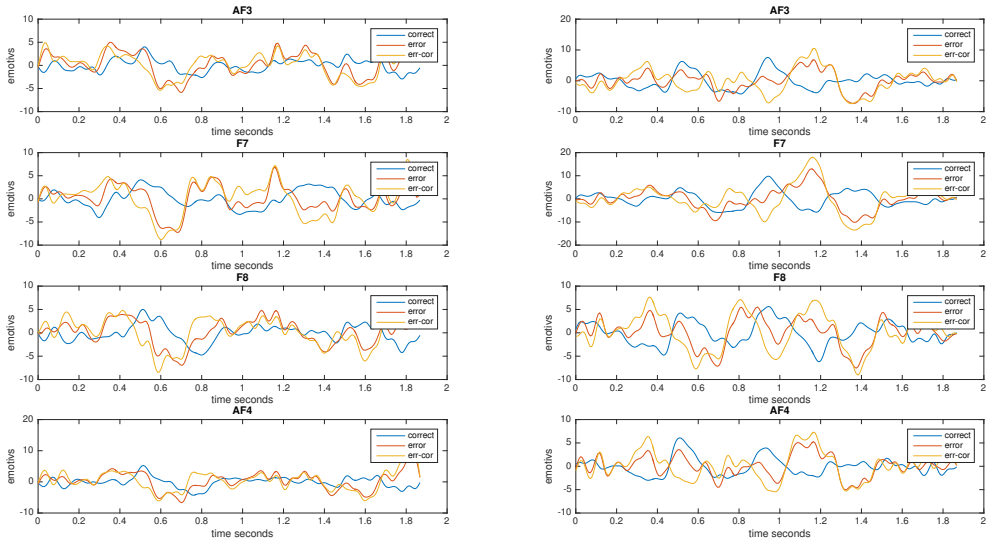


Figure 13: Subject 2 - EPOC: Simple redirect (left) vs Green box preview (right)

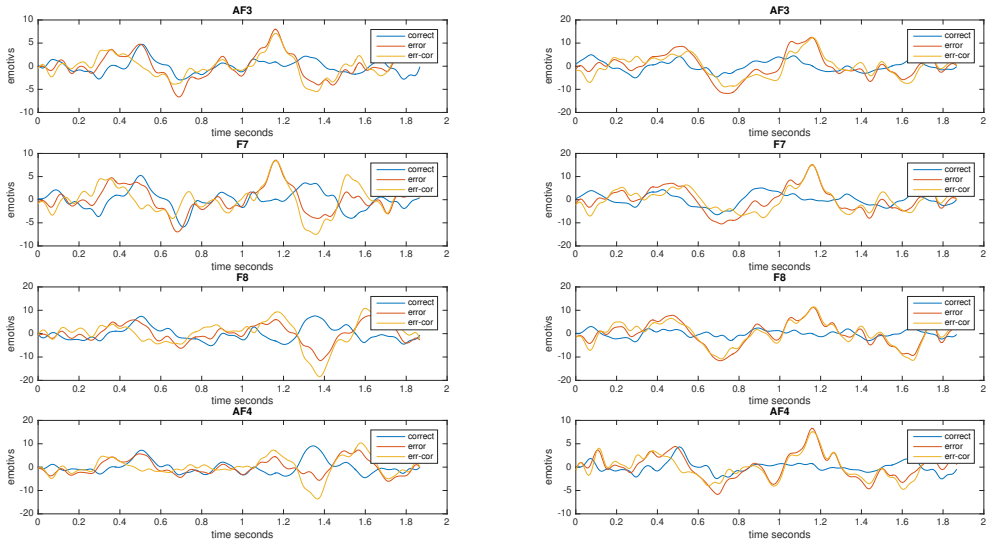


Figure 14: Subject 3 - EPOC: Simple redirect (left) vs Green box preview (right)

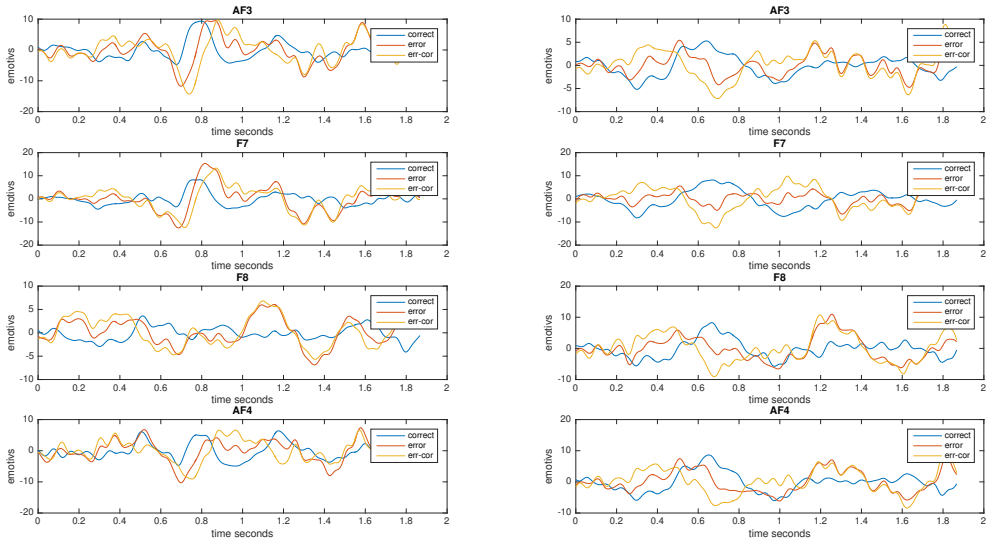


Figure 15: Subject 4 - EPOC: Simple redirect (left) vs Green box preview (right)

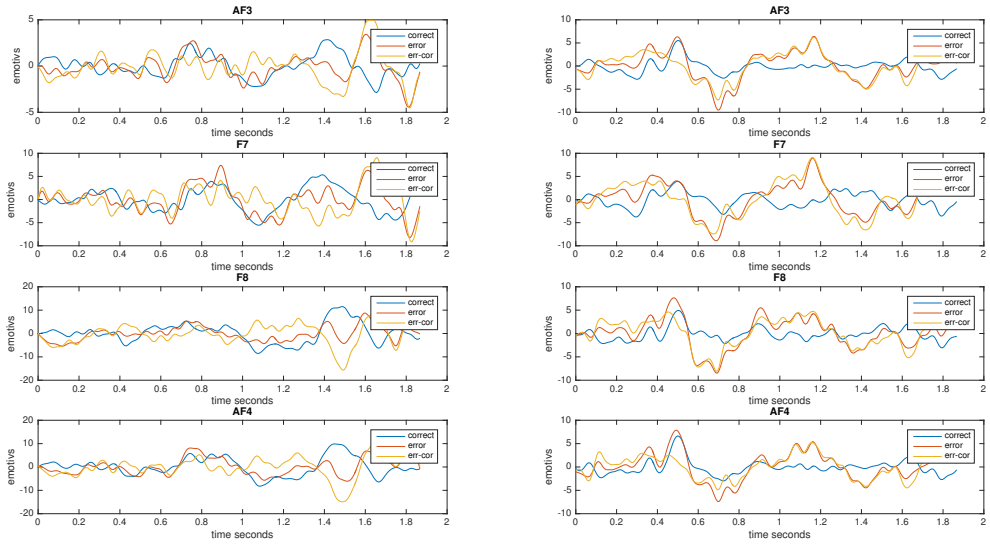


Figure 16: Subject 5 - EPOC: Simple redirect (left) vs Green box preview (right)

Feasibility testing on the error-aware MMI So far all tests have been performed using the SSVEP-based interface. In the following time, we plan to run user-based experiments on how to introduce ErrPs in a gaze-keyboard-typing scenario (i.e. time-locking the ErrPs, optimizing the keyboard interface to include previews of the selected letters, etc.). Moreover, after the definition of the AEC system and in order to define the benefits of AEC, we also plan the following feasibility tests. Users will be asked to type a set of words/sentences without and with AEC. In the first case, GazeTheWeb will be used as the interaction platform, while in the second case GazeTheWeb will be modified to include the ErrP-based automatic correction. The performance measure will include both objective (e.g. time to perform an action, number of errors, etc.) and subjective indicators (based on standardized questionnaires).

GSR-based correlation of ErrPs with stress levels During the experiments with the SSVEP based ErrPs we have also recorded the GSR signals, which will be used in two direction. Initially, our plan is to investigate whether there is a correlation between the stress levels detected by the GSR data and the ErrPs, assuming that errors tend to stress the users. Second, if there is indeed a correlation, GSR signals could be used as an additional input in the AEC mechanisms by increasing the error detection rate and/or time-locking the error events when this is a problem as in the case of the gaze-typing errors.

The "undo" function In this deliverable, we have presented the integration of the EEG sensor through ErrPs in a MMI by implementing an automatic correction mechanism for spelling. In the future, we plan to implement an automatic "undo" function that will be used in the cases where the user erroneously performs an action in the GazeTheWeb interface (e.g. unintentional button clicks).

5 Conclusions

Deliverable 3.1 concluded with the direction that the individual modalities will be investigated and implemented in unison to bring forth a Multimodal Framework for MAMEM's GazeTheWeb Browser. Going in that direction, we have analyzed these modalities in details to try to bring them together in GazeTheWeb browser. The browser had initial mode of interaction to be via eye tracking. With the addition of brain signals and investigation of those signals to extract information via ErrPs, SMRs or SSVEPs, we have managed to build the framework that responds to brain signals when tabs with varying frequencies flicker. So navigation has been taken over by the eye tracking system and selection process has been implemented by BCI signals. Improvements in this section of brain signals is being researched and investigated so as to make the interfaces seamless in multimodal interaction. Improvements in GazeTheWeb browser has been done where Gaze control optimization could be done via a switch that activates hyperlink navigation mode. Automatic Error correction via EEG signal and in totality error-aware MMI could also be seen as a part of this deliverable. In the upcoming deliverable, we seek to improve the experience from technology and design perspective to make GazeTheWeb a complete MMI experience for multimedia interaction.

A Documentation for eeg-processing-toolbox

A.1 Description

In this section we provide the necessary documentation for the eeg-processing-toolbox supporting experimentation with EEG signals, which is available on github[3]. This software follows a modular architecture that allows the fast execution of experiments of different configurations with minimal adjustments of the code. The experimental pipeline consists of the Experimenter class which acts as a wrapper of five more underlying parts:

- The **Session** object: Used for loading the dataset and segmenting the signal according to the periods that the SSVEP stimuli were presented during the experiment. The signal parts are also annotated with a label according to the stimulus frequency.
- The **Pre-processing** object: Includes methods for modifying the raw EEG signal.
- The **FeatureExtraction** object: Performs feature extraction algorithms for extracting numerical features from the EEG signals.
- The **FeatureSelection** object: Selects the most important features that were extracted in the previous step.
- The **Classification** object: Trains a classification model for predicting the label of unknown samples.

A.2 Documentation of the classes

In the following we provide the documentation of the data structures (I/O system), along with the documentation for the abstract classes of each module and the wrapper class Experimenter that combines all the modules. More detailed documentation for all classes can be found in the github project.

A.2.1 Documentation of the I/O system

A.2.1.1 InstanceSet

Listing 1: InstanceSet Class description - Properties

```
 A class for describing a set of instances and labels

 Properties:
-instances : a m x n matrix where m = # instances and n =
 # features
-labels: a m x 1 matrix containing the labels for each
 instance
-K: a m x m matrix containing the kernel of the instances
 required only for the Fast vesrion of LibSVM
```

[3] https://github.com/MAMEM/eeg-processing-toolbox

Listing 2: InstanceSet Class description - Functions

```
Construct an instanceSet
   IS = eegtoolkit.util.InstanceSet(instances, labels)
Compute the kernel
   K = IS.computeKernel(kernel, gamma, maxlag, scaleopt)
 kernel can be one of 'linear' (default),'rbf','chi','xcorr',
 'spearman','correlation','cosine','euclidean','seuclidean',
 'mahalanobis'
Get the training kernel with indexes in the trainidx vector
   Ktrain = IS.getTrainKernel(trainidx)
Get the test kernel with test indexes testidx and train indexes
   Ktest = IS.getTestKernel(trainidx, testidx)
Get instances of specific indices in vector idx
   instance = IS.getInstancesWithIndices(idx)
Get the instances of a specific label
   instances = IS.getInstancesForLabel(label)
Get the indices corresponding to a specific label
   indices = IS.getInstanceIndicesForLabel(label)
Get the instances including the labels as the last row
   dataset = getDataset(IS)
Get the instances with specific indices. The last column of the
matrix will contain the label.
   dataset = IS.getDatasetWithIndices(idx)
Remove instances with specific indices. A new InstanceSet object
is returned by this function without the specified instances
   IS = IS.removeInstancesWithIndices(idx)
Write the dataset to a csv file
   IS.writeCSV(csvname)
Write the dataset to a weka-readable file (arff). The whole
dataset is written if the indices are not given (function
called with 1 argument)
   IS.writeArff(fname, indices)
```

A.2.1.2 Trial

Listing 3: Trial Class description

```
 A class that defines a Trial

 Properties:
 signal: the signal of the trial, matrix m x n, where m is the
channels and n the samples
 label: a label for the trial (typically assigned with the
frequency that is calculated using DIN data)
 duration: the duration of the trial in seconds
 samplingRate: the samling rate used to capture the signal
 subjectid: the subject from whom the trial was recorded
 sessionid: the session id

 type: the type of the trial. Supported types of trials:
 SSVEP = 1;
 ERRP = 2;
 MI = 3;
```

A.2.1.3 Session

Listing 4: Session Class description

```
 SESSION class
 Session I/O, splitting sessions into trials and applying filters

 Properties:
 trials: A cell array with the Trials of the loaded sessions.
 filt: Filter to be applied when data is loaded
 sessions: Cell array with the filenames of the dataset
 subjectids: Vector with the subject ids corresponding
to the loaded trials
 sessionids: Vector with the session ids corresponding
to the loaded trials

 Functions:
init a session
   session = eegtoolkit.util.Session();
init a session with a filter (created with 'filterbuilder')
   session = eegtoolkit.util.Session(filt);
load trials for a subject
   session.loadSubject(subjectid);
load a specific session
   session.loadSubjectSession(subjectid,sessionid);
load everything
   session.loadAll();
clear loaded data
   session.clearData;
apply a filter that was created with 'filterbuilder'
   session.applyFilter(filt);


DATASETS
1. SSVEP Dataset I (SINGLE)
2. SSVEP Dataset II (MULTI)
3. SSVEP Dataset III (EPOC-MULTI)
4. SSVEP Dataset SCCN
5. ERRP Dataset
6. MI Dataset
```

A.2.2 Documentation of the module classes

A.2.2.1 preprocessing

Listing 5: preprocessing Abstract Class description

```
 Abstract class for preprocessing. Implement the functions for
 any preprocessing step (e.g. filtering, artifact removal,
 sub-sampling, rereferencing, etc.

 Properties:
 originalTrials: cell array with the trials to be processed
 processedTrials: the processed trials to be returned

 Functions:
Implement this function to process the in trials and return the
processed trials (out)
   out = obj.process(in);
Info & run time so that the experiments are easily documented.
configInfo is a string with the configuration information and
time is of type double.
   configInfo = obj.getConfigInfo();
   time = obj.getTime();
```

A.2.2.2 featextraction

Listing 6: featextraction Abstract Class description

```
 Abstract class for feature extraction. Implement the functions
 for any feature extractor (e.g. PWelch, FFT, DWT, PYAR, etc.)

 Properties:
 instanceSet: Output - the features in an InstanceSet structure
 trials: Input - the signals in a cell of Trial structures

 Functions:
Implement this function to extract the features from the input
Trials and return the features in an instanceSet
   out = obj.extract(in);
Info & run time so that the experiments are easily documented.
configInfo is a string with the configuration information and
time is of type double.
   configInfo = obj.getConfigInfo();
   time = obj.getTime();
```

A.2.2.3 featselection

Listing 7: featselection Abstract Class description

```
 Abstract class for feature selection. Implement the functions
 for any feature selection method (e.g. PCA, SVD, FEAST, etc.)

 Properties:
 originalInstanceSet: Input - the original instanceSet with the
 features
 filteredInstanceSet: Output - the filtered instanceSet

 Functions:
Implement this function to process the originalInstanceSet trials
and return the filteredInstanceSet
   obj.compute();
Info & run time so that the experiments are easily documented.
configInfo is a string with the configuration information and
time is of type double.
   configInfo = obj.getConfigInfo();
   time = obj.getTime();
```

A.2.2.4 classification

Listing 8: classification Abstract Class description

```
 Abstract class for classification. Implement the functions for
 any classifier (e.g. LibSVM, LDA, MLR, RFs, Adaboost, etc.)

 Properties:
 instanceSet: the training/test set in an InstanceSet structure

 Functions:
Implement this function to train the classification models. Each
implementation should define the type of models that are to be
trained by the build function.
   obj = obj.build();
Implement this function to apply the classification models to the
provided instance, where instance is a matrix #instances x
#features including the instances. The function should return
the following three outputs; 1) output, the classified labels of
the instances (#instances x 1), 2) probabilities, the
probabilities for the output label (instances x 1) and 3) ranking,
a matrix #instances x #classes with the probabilities of each
instance for each class (used for multiclass classification).
   [output, probabilities, ranking] = obj.classifyInstance(instance);
Implement this function to reset the classifier (e.g. delete the
stored classification models
   obj.reset();
Info & run time so that the experiments are easily documented.
configInfo is a string with the configuration information and
time is of type double.
   configInfo = obj.getConfigInfo();
   time = obj.getTime();
```

A.2.2.5 aggregation

Listing 9: aggregation Abstract Class description

```
 Abstract class for aggregation. Implement the functions for any
 fusion methodology (e.g. late fusion, VLAD, channel
 concatenation, etc.)

 Properties:
 featextractors: cell array with the the feature extractors to be
 used on the trials. Each cell contains a featextraction object,
 and the trials are given to each of the objects within the
 Experimenter class
 instanceSet: output - the aggregated features. Can be a simple
 InstanceSet or the extended FusionInstanceSet

 Functions:
Implement this function to process the trials of each
featextraction object in the cell array featextractors. It should
store in the instanceSet property the features.
   obj.aggregate();
Info & run time so that the experiments are easily documented.
configInfo is a string with the configuration information and
time is of type double.
   configInfo = obj.getConfigInfo();
   time = obj.getTime();
```

A.2.3 Documentation of the Experimenter class

Listing 10: Experimenter Class description

```
 EXPERIMENTER class
 Wraps all the necessary objects for performing an experiment.
Before running an experiment all the required properties must be
set.
 Required:
 - session
 - featextraction
 - evalMethod
 - classification
 Optional:
 - featselection
 - aggregator
 - preprocessing

 to run the experiment execute the "run()" method.
 Example:
 experiment = eegtoolkit.experiment.Experimenter;
 experiment.session = eegtoolkit.util.Session;
 experiment.session.loadSubject(1);
 experiment.transformer = eegtoolkit.featextraction.PWelch;
 experiment.extractor = eegtoolkit.featselection.FEAST;
 experiment.classification = eegtoolkit.classification.LIBSVM;
 experiment.evalMethod = experiment.evalMethod.EVAL_METHOD_LOSO;
 experiment.run;
 results = experiment.results;
 confmatrix = results{1}.getConfusionMatrix;
```

A.3 Examples

Some examples are available that are based on the datasets that can be found below.

- exampleCSP, extract common spatial patterns in dataset III of BCI competition II
- exampleCombiCCA, SSVEP recognition using the CombinedCCA method from [2]. Based on this dataset
- exampleDefault, performs a simple experiment on Dataset I & II
- exampleEPOCCCASVM, SSVEP recognition using SVM on the CCA coefficients, based on Dataset III
- exampleERRP, recognition of error related potentials, based on the dataset provided by [3]
- exampleEarlyFusion, demonstrates how to merge features extracted by
- different electrode channels, based on Dataset II.
- exampleEpoc, performs an experiment for the dataset that was recorded with an EPOC device (Dataset III)

Title	ID	Description
SSVEP-SINGLE	1	EEG signals with 256 channels captured from 11 subjects executing a SSVEP-based experimental protocol. Five different frequencies (6.66, 7.50, 8.57, 10.00 and 12.00 Hz) presented in isolation have been used for the visual stimulation. The EGI 300 Geodesic EEG System (GES 300), using a 256-channel HydroCel Geodesic Sensor Net (HCGSN) and a sampling rate of 250 Hz has been used for capturing the signals.
SSVEP-SINGLE	2	EEG signals with 256 channels captured from 11 subjects executing a SSVEP-based experimental protocol. Five different frequencies (6.66, 7.50, 8.57, 10.00 and 12.00 Hz) presented simultaneously have been used for the visual stimulation. The EGI 300 Geodesic EEG System (GES 300), using a 256-channel HydroCel Geodesic Sensor Net (HCGSN) and a sampling rate of 250 Hz has been used for capturing the signals.
SSVEP-EPOC-MULTI	3	EEG signals with 14 channels captured from 11 subjects executing a SSVEP-based experimental protocol. Five different frequencies (6.66, 7.50, 8.57, 10.00 and 12.00 Hz) presented simultaneously have been used for the visual stimulation, and the Emotiv EPOC, using 14 wireless channels has been used for capturing the signals.
SSVEP-SCCN	4	This is an external dataset provided by [24]
ERRP	5	This is an external dataset provided by [11]
MI	6	This is an external dataset provided by the BCI Competitions (Dataset III from BCI competition II) [1]

Table 7: Available datasets for loading with eeg-processing-toolbox

- exampleITCCA, SSVEP recognition using the ITCCA method from [2]. Based on this dataset
- exampleL1MCCA, SSVEP recognition using the L1MCCA method from [2]. Based on this dataset
- exampleLSL, Online recognition of SSVEP signals using the LSL library.
- exampleLateFusion, merging the output of different classifiers by majority voting, based on Dataset II.
- exampleMotorPWelch, classification of right/left hand motor imagery based on the dataset III of BCI competition II
- exampleOptimal, performs an experiment with the optimal settings for Dataset I & II
- exampleSMFA, SSVEP recognition with using SMFA [4]

A.4 Datasets

There are 6 datasets that can be directly loaded in the toolbox by using the ID in the second column of Table 7 as an argument in the *session.load* functions.

B Documentation for eyeGUI library

In this section we provide the documentation for eyeGUI library to design and develop graphical user interfaces suitable for eye-based input control (available on MAMEM GitHub). eyeGUI (used in the development of GazeTheWeb-Browse) specifically focuses on the concept of eye-controlled user interfaces to ease the development of interactive GUI elements, compared to other available open source approaches like OpenGazer[4], which addresses the generic problem of measuring eye gaze.

The eyeGUI library enables the manipulation and rendering of user interfaces for eye tracking input. A variety of elements, like buttons, images and texts, to build a proper interface can be used from the framework. Most of them can be customized in their size, appearance or behavior, e.g., buttons can be given an arbitrary icon that scales automatically when the overall size of the interface changes, though the ratio of height and width of the image stays the same. eyeGUI layouts are supposed to be overlay for specific application interacting with the gaze input from the eye tracking device. Moreover stand-alone application can also be developed using eyeGUI layouts.

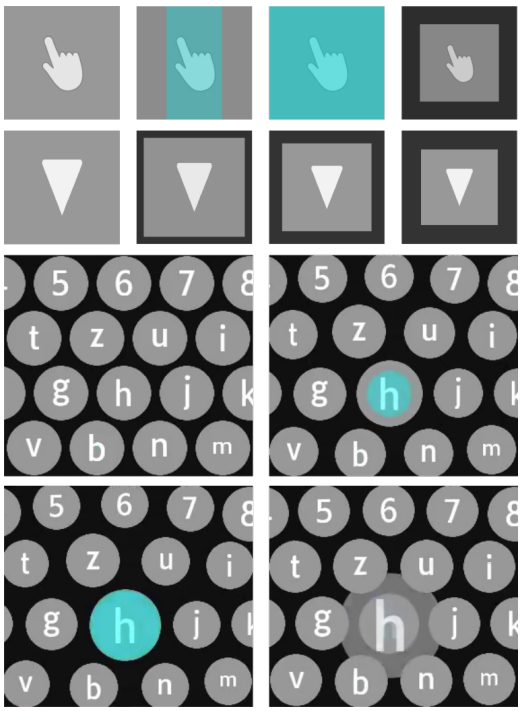


Figure 17: eyeGUI: Interactive elements like buttons (upper row), sensors (middle row) and character typing (bottom)

All elements in eyeGUI are designed especially for eye tracking in their size, appearance, and user interaction, e.g., buttons get activated when the gaze hits them, and they shrink after triggering to provide the user with interaction feedback. A colored overlay increasing in size works as a visual representation of the remaining time until the triggering. It also offers other eye tracking specific features like optionally showing the gaze path during usage. Figure 17, showcases some examples of these elements; the first row signifies a button triggering through eye gaze interaction, where the animated highlighting over the icon shows the focus duration and the button is pressed after the duration is exceeded. The second row indicates a sensor like button, relevant for progressive elements like scrolling. The four images at bottom show different stages of eye-typing with magnifying effect of character selection.

[4] https://github.com/opengazer/OpenGazer

B.1 eyeGUI Architecture

Figure 18, outlines the architecture of eyeGUI framework. It is developed in C++ 11 and based on OpenGL. It offers to build user interfaces for eye tracking by adding XML-files as layouts and manipulating elements (e.g. buttons) within these layouts via listeners. The listeners can be accessed in the application environment to give every button an own functionality and also to interact with external APIs. Eye tracking device (e.g., SMI REDn Scientific, Tobii EyeX) would send raw gaze data to application, which implements a listener for that stream and receives data. Filtered gaze signal through application is piped to eyeGUI layouts, which handle interaction and call events in the interface (e.g. button pressed), and the application would react to those user interactions by call backs.

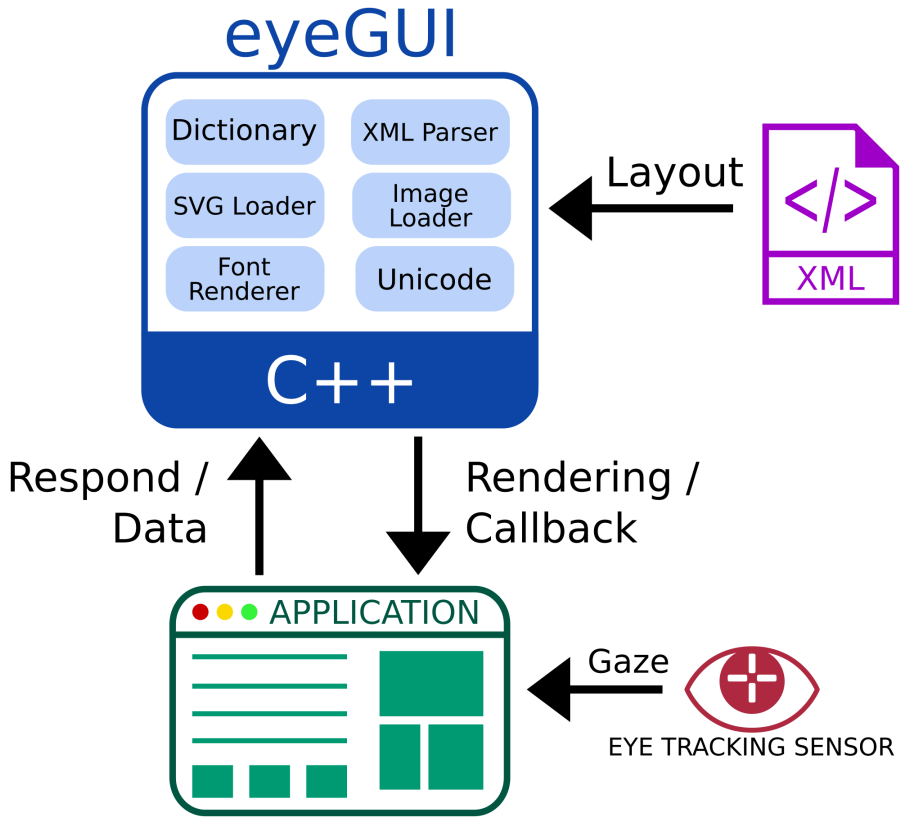


Figure 18: eyeGUI architecture

B.2 eyeGUI Integration

The integration of eyeGUI is similar to other OpenGL user interface libraries[5]. The developer is free to choose how to create a window and on which way to initialize the OpenGL context. Before the render loop is entered, the GUI object for eyeGUI must be instantiated and an arbitrary number of layouts from XML files might be added. During the render loop, for every frame the most recent gaze input is used to update eyeGUI, which provides feedback whether the input has been used by any layout. Based on that feedback, the developer can decide how to update the application's content. For interactions with the application, listeners might be employed to enable eyeGUI to call back into the application if a button is triggered or other events occurred. A detailed sequence of a minimal eyeGUI application is listed in Figure 19. All functions are accessible through a single header files with C++ functions and the memory allocation for displayed images and other is handled internally.

For a more in-depth understanding on eyeGUI usage and functions, Figure 20 provides the overview and self explanatory structure of eyeGUI classes. We provide further details of

[5]ImGui: `https://github.com/ocornut/imgui`

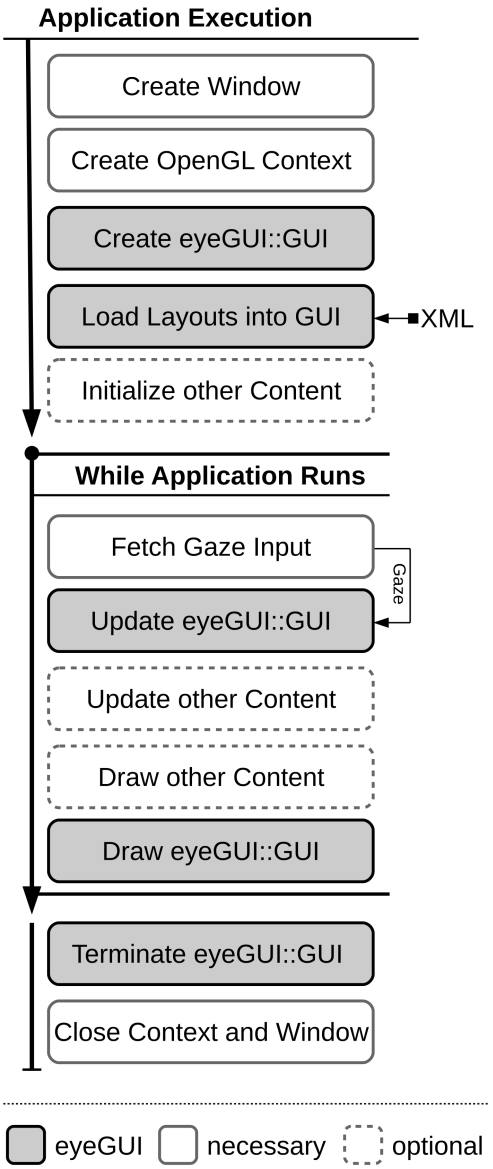


Figure 19: Sequence diagram of exemplary eyeGUI usage in custom application

some relevant classes with their attributes and properties in Figures 16-21. Figure 21 shows the possible attributes of elements. Additional attributes for each element are listed in the subsequent figures, i.e., Figure 22 indicates the attributes and an example to display images in the layout. To use a block of solid color, attributes and example is included in Figure 23. The shown example would use background color from chosen style (please take a look at Styling for more details). With default styling, a block is invisible. Figure 24 indicates the structure to place multiple elements inside a frame. Cells are defined by rows and columns sized using relative scales in percent of available space. Each row can have free count of columns. Size attributes per row and columns are mandatory and must sum to 100% for each row's columns and for the rows itself.

Frames Frames are more of an internal structure used in the layout class. There is one main frame holding the root element of a layout. Interesting for the user is the possibility to add floating frames on top of this main frame. One can load a brick consisting of elements to a floating frame using the programming interface while providing relative position and size in respect to the GUI size.

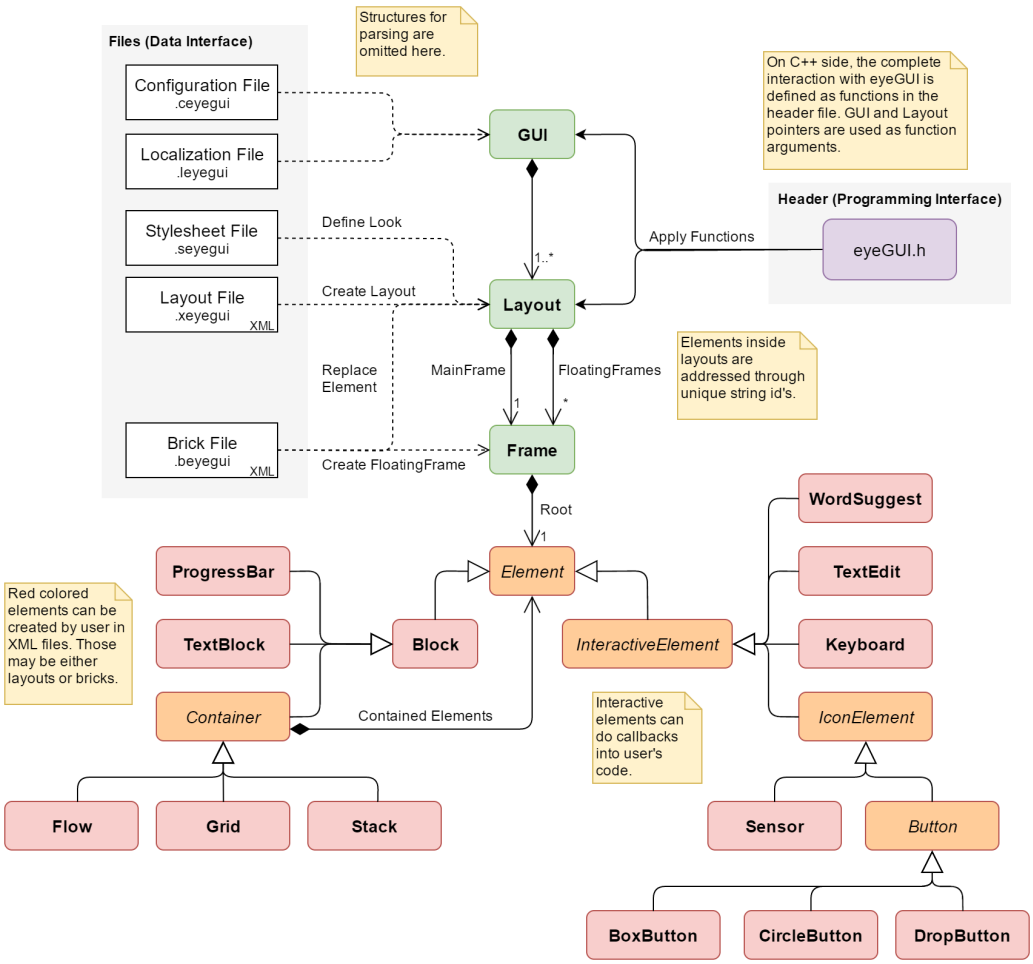


Figure 20: Simplified eyeGUI class structure

Styling Layout XML files do not contain any style information but each element can have a style attribute linking to a style defined in a stylesheet. Those stylesheets are loaded per layout and defined in ".seyegui" files. Colors must be given in RGBA hexadecimal code with a "0x" as prefix. Each style has to have an unique name and can inherit properties from another style by postfixing it with ":" and the name of the parent style. The parent must be defined before the usage as parent. If a style property is not defined at all, some fallback is used. Following properties are possible to define: A element uses the style of its parent element. If no style is chosen, the style called "default" is used. This style is always created and uses fallback values, if not overridden in the stylesheet.

Configuration Dwell time of button activation, animation play time and further interaction parameters can be set up globally in a configuration file. It enables one to adapt the behavior to the needs and abilities of an user. Adaptability is a key factor to improve usability of the application's interface since different users are at custom learning stages and especially the user group targeted by MAMEM may benefit through their various interaction drawbacks.

Manipulation Manipulation of layouts or elements is a powerful tool to create functionality inside the user interface depending on actions of the user or other influences. They can even be done inside the methods of a listener implementation of SensorListener or ButtonListener. There are multiple functions in the interface available to change the order of layouts in the GUI, change the visibility or block input. In most cases, a pointer to the owning GUI and the layout itself is necessary. Manipulation of elements inside a layout with a function call is done by determining the element to manipulate by a pointer to the layout and the id of the element specified in the XML.

Attribute	Type	Description
id	string	Unique id to fetch element from C++ side
style	string	Name of style used for this element and its children
relativescale	percent	Relative scaling of element
border	percent	Space between this element and border of provided space
dimming	bool	*true* if element should dim or *false* if not
adaptivescaling	bool	*true* if element should use adaptive scaling or *false* if not

Figure 21: eyeGUI elements: attributes

Attribute	Type	Description
src	string	Relative path to displayed image (png, tga, bmp, jpg or svg) like "Images/Lena.png"
alignment	string	*original*, *stretched* or *zoomed*

Example:

```
<picture id="portrait" alignment="stretched" src="Images/Lena.png"></picture>
```

Figure 22: eyeGUI elements: including a picture

Keyboard Keyboard to type in symbols with users eyes. The character set depends one the character set chosen in the GUIBuilder before GUI construction. One keyboard has multiple keymaps which can be set active using the interface functions and indices. In addition, one can tell the keyboard via interface to display lower or upper case letters. Another feature is called fast-typing, which is explained in the corresponding paragraph. Fast-Typing can be activated or deactivated using the 'setFastTypingOfKeyboard' function in the header. Without fast-typing, a letter in the keyboard is pressed when a certain threshold is full and an animation is displayed. With this feature, each letter that is looked at shortly and is visually marked with a circle is remembered. The listeners get that letters not before a single letter is focused until the threshold is full and the animation plays.

Dependencies Following are the dependencies of eyeGUI library:

- NanoSVG: https://github.com/memononen/nanosvg
- TinyXML2: https://github.com/leethomason/tinyxml2
- GLM: http://glm.g-truc.net/0.9.7/index.html (MIT license chosen)
- stb_image: https://github.com/nothings/stb
- FreeType 2.6.1: http://www.freetype.org/ (FreeType license chosen)
- UTF8-CPP: https://github.com/nemtrif/utfcpp

Attribute	Type	Description
consumeinput	bool	*true* if block should consume input or *false* if not
backgroundsrc	string	Relative path to displayed image (png, tga, bmp, jpg or svg) like "Images/Lena.png". If empty, no background image is used
backgroundalignment	string	*original*, *stretched* or *zoomed*

Example:

```
<block style="blocky"></block>
```

Figure 23: eyeGUI elements: including a block of solid color

C Documentation for the GSR signal analysis algorithm

In this section we provide the documentation for the GSR signal analysis algorithm also included in the eeg-processing-toolbox.

Listing 11: StressDetection Class description - Properties

```
        Detects the stress level of a subject using the GSR
    measurements from a Shimmer 3+ device. The input Skin
    Conductance (SC) Values  in (kU) are processed and different
    thresholds for stress detection are computed. A 5-minute
    period  is the minimum time-period required to compute the
    thresholds. For real-time detection, acquisitions larger
    than two minutes are required.

Properties:
    stressLevelThresholds: A 5x1 vector with the five thresholds
    (from 1 to 5) calculated based on calibration data.
```

Attribute	Type	Description
consumeinput	bool	*true* if grid background should consume input or *false* if not
innerborder	percent	Space between grid border and inner elements
showbackground	bool	*true* if background should be rendered or *false* if not
backgroundsrc	string	Relative path to displayed image (png, tga, bmp, jpg or svg) like "Images/Lena.png". If empty, no background image is used
backgroundalignment	string	*original*, *stretched* or *zoomed*

Example:

```
<grid>
  <row size="40%">
    <column size="100%">
      <block style="red"></block>
    </column>
  </row>
  <row size="60%">
    <column size="100%">
      <block style="blue"></block>
    </column>
  </row>
</grid>
```

Figure 24: eyeGUI elements: including a grid

Listing 12: StressDetection Class description - Functions

```
 Functions:
This function calculates the thresholds based on gsrData, a 1xn
vector containing the calibration data, where n corresponds to
the number of GSR samples in kOhms. Pass the sampling rate as a
parameter (256Hz is hardcoded at the moment)
    SD.trainThresholds(gsrData)

This function searches for an appropriate GSR stream in an XDF
file and calculate stress thesholds. (More details on XDF files
on https://github.com/sccn/xdf)
    SD.trainThresholdsFromXDF(xdfFilename)

This function calculates the stress levels over a period based on
gsr, a 1xn vector containing the GSR samples. The trainThresholds
method must be executed prior to this method.
    SD.detectStress(gsr)

This function calculates the stress levels in real-time from a
GSR LSL stream and output the results in a new LSL stream called
"Stress_Levels". Again the trainThresholds must be executed first
before using this method.
    SD.runStressDetectionOutlet()
```

Listing 13: StressDetection Class description - Example

```
Example: To run the stress detection method
   sd = eegtoolkit.services.StressDetection;
   sd.trainThresholds(calibrationGSRData);
   sd.detectStress(gsrData);

   calibration data is 1xn vector containing SC values, used to
   identify stress level thesholds. As explained above, at least
   a 5 minute observation period is required,

   gsrData is 1xn vector containing SC values that are used to
   classify stress levels (from 1 to 5). At least a two-minute
   acquisition period
   is required.
```

References

[1] Bci competitions. http://www.bbci.de/competition/ [Online; accessed 2-December-2015].

[2] K. K. Ang, Z.Y Chin, C. Wang, C. Guan, and H. Zhang. Filter bank common spatial pattern algorithm on bci competition iv datasets 2a and 2b. *Frontiers in Neuroscience*, 6:39, 2012.

[3] Michael Barz, Florian Daiber, and Andreas Bulling. Prediction of gaze estimation error for error-aware gaze-based interfaces. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, ETRA '16, pages 275–278, New York, NY, USA, 2016. ACM.

[4] Meera M Blattner and Ephraim P Glinert. Multimodal integration. *IEEE multimedia*, 3(4):14–24, 1996.

[5] Pavel Bobrov, Alexander Frolov, Charles Cantor, Irina Fedulova, Mikhail Bakhnyan, and Alexander Zhavoronkov. Brain-computer interface based on generation of visual images. *PloS one*, 6(6):e20674, 2011.

[6] Andrew Campbell, Tanzeem Choudhury, Shaohan Hu, Hong Lu, Matthew K Mukerjee, Mashfiqui Rabbi, and Rajeev DS Raizada. Neurophone: brain-mobile phone interface using a wireless eeg headset. In *Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds*, pages 3–8. ACM, 2010.

[7] Juan J. Cerrolaza, Arantxa Villanueva, Maria Villanueva, and Rafael Cabeza. Error characterization and compensation in eye tracking systems. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, pages 205–208, New York, NY, USA, 2012. ACM.

[8] W D Hairston. Accounting for timing drift and variability in contemporary electroencephlography (eeg) systems. *Technical report*, 2012.

[9] Ulrich Hoffmann, Jean-Marc Vesin, Touradj Ebrahimi, and Karin Diserens. An efficient p300-based braincomputer interface for disabled subjects. *Journal of Neuroscience Methods*, 167(1):115 – 125, 2008. Brain-Computer Interfaces (BCIs).

[10] Patrick Horain, Amine Chellali, Malik Mallem, Boris B. Velichkovsky, Mikhail A. Rumyantsev, and Mikhail A. Morozov. New solution to the midas touch problem: Identification of visual commands via extraction of focal fixations. *The 6th international conference on Intelligent Human Computer Interaction, IHCI 2014*, 39:75 – 82, 2014.

[11] Iñaki Iturrate, Jonathan Grizou, Jason Omedes, Pierre-Yves Oudeyer, Manuel Lopes, and Luis Montesano. Exploiting task constraints for self-calibrated brain-machine interface control using error-related potentials. *PloS one*, 10(7):e0131491, 2015.

[12] Alejandro Jaimes and Nicu Sebe. Multimodal human–computer interaction: A survey. *Computer vision and image understanding*, 108(1):116–134, 2007.

[13] Ryan K. Jessup, Jerome R. Busemeyer, and Joshua W. Brown. Error effects in anterior cingulate cortex reverse when error likelihood is high. *Journal of Neuroscience*, 30(9):3467–3472, 2010.

[14] C. Jeunet, E. Jahanpour, and F. Lotte. Why standard brain-computer interface (bci) training protocols should be changed: an experimental study. *Journal of Neural Engineering*, 13(3):036024, 2016.

[15] Kapil D Katyal, Matthew S Johannes, Timothy G McGee, Andrew J Harris, Robert S Armiger, Alex H Firpi, David McMullen, Guy Hotson, Matthew S Fifer, Nathan E Crone, et al. Harmonie: A multimodal control framework for human assistive robotics. In *Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on*, pages 1274–1278. IEEE, 2013.

[16] Minho Kim, Byung Hyung Kim, and Sungho Jo. Quantitative evaluation of a low-cost noninvasive hybrid interface based on eeg and eye movement. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 23(2):159–168, 2015.

[17] Ravi Kuber, Wai Yu, Philip Strain, Emma Murphy, and Graham McAllister. Assistive multimodal interfaces for improving web accessibility.

[18] Li Khim Kwah, Lisa A Harvey, Joanna HL Diong, and Robert D Herbert. Half of the adults who present to hospital with stroke develop at least one contracture within six months: an observational study. *Journal of Physiotherapy*, 58(1):41–47, 2012.

[19] Denis Lalanne, Laurence Nigay, Peter Robinson, Jean Vanderdonckt, Jean-François Ladry, et al. Fusion engines for multimodal input: a survey. In *Proceedings of the 2009 international conference on Multimodal interfaces*, pages 153–160. ACM, 2009.

[20] R. Leeb, C. Brunnera, G. R. Muller-Putz, A. Schloogl, and G. Pfurtscheller. Bci competition 2008 - graz data set b. `https://lampx.tugraz.at/~bci/database/002-2014/description.pdf`, 2008.

[21] Yue Liu, Xiao Jiang, Teng Cao, Feng Wan, Peng Un Mak, Pui-In Mak, and Mang I Vai. Implementation of ssvep based bci with emotiv epoc. In *2012 IEEE International Conference on Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS) Proceedings*, pages 34–37. IEEE, 2012.

[22] I Scott MacKenzie. Evaluating eye tracking systems for computer input. *Gaze Interaction and Applications of Eye Tracking: Advances in Assistive Technologies: Advances in Assistive Technologies*, page 205, 2011.

[23] David McNeill. *Hand and mind: What gestures reveal about thought*. University of Chicago press, 1992.

[24] M. Nakanishi, Y. Wang, Y.T. Wang, and T.P. Jung. A comparison study of canonical correlation analysis based methods for detecting steady-state visual evoked potentials. *PLoS ONE*, 10(10), 2015.

[25] Jakob Nielsen. A virtual protocol model for computer-human interaction. *International Journal of Man-Machine Studies*, 24(3):301–312, 1986.

[26] Sharon Oviatt. Multimodal interactive maps: Designing for human performance. *Human-computer interaction*, 12(1):93–129, 1997.

[27] Sharon Oviatt and Philip Cohen. Perceptual user interfaces: multimodal interfaces that process what comes naturally. *Communications of the ACM*, 43(3):45–53, 2000.

[28] Pernilla Qvarfordt and Shumin Zhai. Conversing with the user based on eye-gaze patterns. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 221–230. ACM, 2005.

[29] Y. Renard, F. Lotte, G. Gibert, M. Congedo, E. Maby, V. Delannoy, O. Bertrand, and A. Lcuyer. Openvibe: An open-source software platform to design, test, and use brain-computer interfaces in real and virtual environments. *Presence: Teleoperators and Virtual Environments*, 19(1):3553, 2010.

[30] Catherine Sherrington, Julie C Whitney, Stephen R Lord, Robert D Herbert, Robert G Cumming, and Jacqueline CT Close. Effective exercise for the prevention of falls: a systematic review and meta-analysis. *Journal of the American Geriatrics Society*, 56(12):2234–2243, 2008.

[31] Martin Spler, Michael Bensch, Sonja Kleih, Wolfgang Rosenstiel, Martin Bogdan, and Andrea Kbler. Online use of error-related potentials in healthy users and people with severe motor impairment increases performance of a p300-bci. *Clinical Neurophysiology*, 123(7):1328 – 1337, 2012.

[32] Martin Spler, Wolfgang Rosenstiel, and Martin Bogdan. Online adaptation of a c-vep brain-computer interface(bci) based on error-related potentials and unsupervised learning. *PLoS ONE*, 7(12):1–11, 12 2012.