POLITECNICO DI MILANO
Polo Regionale di Como
M.Sc. in Computer Science and Engineering
Department of Electronics, Information and Bioengineering

DEVELOPMENT OF A PORTABLE AUDITORY P300-BASED BRAIN-COMPUTER INTERFACE FOR YES-NO COMMUNICATION TO AMYOTROPHIC LATERAL SCLEROSIS

AI & R Lab
Artificial Intelligence and Robotics
Laboratory of Politecnico di Milano

Advisor: Prof. Matteo Matteucci
Tutor: Ing. Fabio Veronese

Dissertation of:
Emanuele Opna De Bernardi, ID 800493

Academic Year 2014-2015

To my parents who have supported me every day to find the way I've always searching for...

Abstract

A Brain-computer interface (BCI) is a direct communication pathway between the brain and an external device. This interface uses a special technique called Electroencephalography (EEG) that allows to record the neural brain activity focusing on the temporal content of the EEG. This pattern are events-correlated and are based on the P300 wave signal that is an event related potential (ERP) component eliticied in the process of decision making. Using the oddball paradigm is possible to elicit this wave, in which low-probability target items are mixed with high-probability non-target (or "standard") items.

In particular, BCIs provide a non-muscular communication for individuals diagnosed with a late-stage motoneuron disease (e.g., amyotrophic lateral sclerosis (ALS)). In the final state of the disease, a BCI cannot rely on the visual modality so this work studies a method to achieve high accuracies using auditory stimuli only. The presented BCI offers communication with binary choices (yes/no) independent from vision and it requires only little time per selection to patients who lost all motor functions and have a short visual attention span.

The main purpose of this thesis is thus to develop a portable BCI application for delivering acoustic stimuli via headphones to participants based on an auditory three-stimulus oddball paradigm. For this purpose the application was trained on three subject in order to test the audio protocol and the offline classification obtaining an average accuracy of 98.33% and an average communication speed of 1.14 bits/min.

Riassunto

Un' interfaccia cervello-computer (BCI, dall' inglese *Brain-Computer Interface*) è un dispositivo che permette la diretta comunicazione tra il cervello ed un dispositivo esterno. Questo tipo di interfaccia utilizza l' elettroencefalografia (EEG, dall' inglese *Electroencephalography*) che permette di registrare l' attività celebrale focalizzando il contenuto temporale dell' EEG. Questi modelli sono evento-correlati e si basano sul segnale d' onda P300, definito come un potenziale ad evento correlato (ERP) suscitato nel processo di scelta di un evento raro. Questo avviene utilizzando il paradigma oddball, in cui vi è una bassa probabilità di ottenere un bersaglio raro ed un' alta probabilità di ottenere un bersaglio voluto.

In particolare, la BCI permette a dei soggetti in cui è stata diagnosticata una malattia degenerativa come ad esempio soggetti con sclerosi laterale amiotrofica - ALS, dall' inglese *amyotrophic-lateral sclerosis*, una comunicazione non muscolare. All' ultimo stadio della malattia, una BCI non può essere utilizzata tramite lo stimolo visivo per la poca concentrazione, quindi è stato esaminato un metodo che raggiunge elevate accuratezze utilizzando degli stimoli acustici. La BCI presentata in questo lavoro di tesi permette di comunicare con una scelta binaria tra parole (si/no) indipendentemente dal segnale visivo, richiedendo bassi tempi di selezione per pazienti che hanno perso le loro funzioni motorie ed hanno una ridotta capacità di concentrazione.

Lo scopo principale della tesi presentata è quello di sviluppare su tablet un' applicativo BCI portatile, che permetta di inviare in cuffia al paziente degli stimoli acustici basati sul paradigma odd-ball a tre stimoli. A questo scopo l'applicativo sviluppato è stato addestrato e testato su tre soggetti sani in modo da verificarne il corretto funzionamento, testando inoltre il protocollo audio tramite classificazione delle scelte binarie, ottenendo un' accuratezza media del 98.33% ed una velocità di comunicazione di 1.14 bits/min.

Acknowledgements

This thesis allowed me to work for a long period during my master of science course in Politecnico di Milano. During this period I learned a lot of interesting things and I grew up in many different aspects, also sharing many thoughts and ideas with many people.

First of all I want to thank **Matteo Matteucci** that supervised my work and he gave me the opportunity to get closer to this wonderful field, building something that may helps people in their life combining it to the Android development world that always fascinated me. Also he gave to me the opportunity to follow three students of the Politecnico di Milano for their bachelor thesis, making a work experience with me during my thesis period. So I would to thank **Matteo**, **Nicolas** and **Davide** for helping me to build the hardware needed for the thesis realization, for lending themselves as subjects to test my work and for writing the user manual presented at the end of the thesis.

Also I would like to thank **Fabio Veronese**, my PhD tutor, that especially in the last part of my work helps me to put together some things that have allowed me to achieve good results, the researcher **Simone** and the PhD student **Ashkan** who shared with me the launch breaks and many moments of companionship during my thesis work. Another important thank goes to professor **Fabio Salice** for the provided tools that I have used in the AirLab and **Daniel** that made me company in the last period of my permanence here. **Enzo**, **Matteo Thor**, **Jody**, **Massimo**, **Marco Dac** and **Alessandro** to have shared with me many good times together during this years, **Alessandro** who introduced me the Como Campus, and in particular **Cri**, **Consa**, **Romesh**, **Mikk**, **Bomba**, **Ale**, **Elsa**, **Orkun**, **Matteo**, **Giovanni**, **Fabio**, and **Giulia** who accompanied me from the beginning of this long difficult journey at Politecnico. Also to many others colleagues which I spent part of my work.

I used many software tools and components, and most of them have been available for free, so I would to be grateful to **GNU/Linux** distributions in particular to **Ubuntu Studio** (the OS that I have used for a long time), **Google** for giving me all materials to develop my work as the **Android API**, the **SDK**, the **Android Studio** platform, **Android developer** site for the support, **Chrome**, and the updated Factory images installed on my tablet used to test the application. **Gimp** for editing images, **OpenShot** for creating demos, **LaTeX** that hapls me to document my thesis, **Java**, **Octave** for running the genetic algorithm code, **NoMachine** to control the remote server, **EdfBrowser** to view files generated from the application, **Wikipedia** for some consultations, and many thanks to **Stackoverflow** which helps me to solve some problems during the design phase of the application.

Many people accompanied me indirectly during my study career. I want to thank first of all family. Thanks **dad** and **mom** for the supporting me throughout my journey since I was a child and always helping me to believe in what I want. Thanks **Titi** for helping me with your important advices during my study career.

Cattu and **Robb** (my best friends) and **Corra** thanks for giving me my best music moments. **Brich**, **Dany**, **Ely**, **Ele**, **Riki**, **Ale** and **Jeeck**, and all the other friends that always believed in me during these long years of my life.

Thanks **Tina**, **Antonio** and **Giada** for your big moral support and for always made me feel like I was at home.

The final special thank goes to **Sara**. Thanks for having always been at my side during the university and in my life, thank your for always believed in me and giving me day by day the right encouragement to continue my work in many hard times. Thank you to giving me the force to reach my goal until the end.

At the beginning of each chapter I quoted some sentences that are a great source of inspiration for me. **Jimi Hendrix** has contributed in some way to this thesis and in my life in many occasions. His music always helped me to overcome hard times and with it he has contributed to make a better world.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> *"Music doesn't lie. If there is something to be changed in this world, then it can only happen through music."*
>
> Jimi Hendrix

Nowadays, Brain-Computer Interfaces (BCI) are still the most innovative technology in the area of control digital machines directly by means of signals sent by brain. With this type of technology, people can express their wills or control equipment through the brain without languages or actions. These interfaces are also used to provide a communication between completely disabled patients due to spinal cord injury or motor disease locked-in state (LIS) patients, in particular the ones that are affected by amyotrophic lateral sclerosis (ALS). This disease is a progressive neurodegenerative pathology involving motor neurons in the cerebral cortex, corticospinal tract, brainstem, and spinal cord.

1.1 Scientific research in the BCI field

Research in the field of brain-computer interfaces has its roots in the investigations and studies about the brain anatomy and physiology. With a BCI a person ideally does not have to make use of the common output pathways of peripheral nerves and muscles, which is the main argument in favour of a BCI system. However, it is only recently that advances in cognitive neuroscience and brain imaging technologies have started to provide us with the ability to interface directly with the human brain. Farwell and Donchin (1988) showed that it is possible for humans to communicate using a BCI, by means of their event-related potentials (ERPs; e.g., P300), without the involvement of their voluntary muscle activity. This ability is made possible through the use of sensors that can monitor some of the physical processes that occur within the brain that correspond with certain forms of thought.

A BCI is a system that allows the communication between the brain and an external device such as a computer. This communication system enables the generation of a control signal from brain responses bypassing physical movements and conveys messages directly from the brain to a computer. Therefore, it constitutes a novel communication option for people with severe motor disabilities, such as ALS-LIS patients that have only residual control over few muscles (e.g., vertical eye movement), patients with brain injuries, or patient suffering neurological diseases in which remain concious, but cannot move any of their muscles. Depending on the degree of impairment caused by the injury or the progression of the disease, communication can become very difficult or even impossible. Patients may become unable to express their opinions and decisions on important questions regarding their clinical treatment or their living and biologic wills.

This loss of communicative abilities can be overcome with interfaces that bypass the need for muscular control and detect the user's intentions directly from signals recorded from the brain. These brain-computer interfaces (BCIs) are currently not only used for communication but also for restoration of motor control. So a common principle of all BCI paradigms is that they must be controllable without the use of muscles to suit the needs of the aforementioned patient groups.

BCI systems can use a variety of different electro-physiological signals and currently most BCI paradigms are based on control signals extracted from the electroencephalogram (EEG). EEG BCIs with ALS patients have been implemented using different components of the EEG such as slow cortical potentials (SCPs), sensor monitor rhythms (SMR) and event-related potentials (ERPs). Only one particular EEG based BCI has been successful with this patients. The P300 event-related potential (ERP) [47] elicited by sensory stimuli as the measured brain response, is the direct result of a decision of making. The P300 is considered to be an endogenous potential, as its occurrence links not to the physical attributes of a stimulus, but to a person's reaction to it. More specifically, the P300 is thought to reflect processes involved in stimulus evaluation or categorization. The P300 is elicited by unexpected stimuli with variations in latency between 250 and 700 ms on central to parietal locations [21], and it is elicited using the oddball paradigm, in which low-probability target items are mixed with high-probability non-target (or "standard") items.

Initially it is proposed the visual P300 BCI paradigm with a matrix composed by letters and numbers. A matrix rows and columns flash randomly and the user is required to focus attention on the cell with the desired item. Any cell flashes only twice in a sequence and thus it becomes an oddball which elicits a P300. In addition to spelling, these visual BCI systems can also be used to control various applications, such as an internet browser [97], a painting application [81] and even a wheelchair controller [112]. Generally, successful use of a BCI is defined by the accuracy achieved (as percent of targets selected correctly), the content of information of each selection (i.e., the number of different targets) and the time needed for one selection. All three factors determine the speed of the communication possible with the BCI that usually is defined as information transfer rate (ITR) in bits/min.

The visual P300 BCI permits fast communication without user training for healthy participants but the performance of patients with motor disease is usually lower than that of healthy controls. In the latest stage of amyotrophic lateral sclerosis (ALS) patients lose all voluntary muscular control [5], including voluntary fixed visual control, which makes the use of a visual BCI difficult if not impossible. Since these are the users that would benefit most from a BCI, P300 BCI systems based on auditory stimulation have been proposed.

The thesis is focused on a study of BCI for ALS patients that consider an optimized paradigm employing auditory distinguishable stimuli with minimum effort (S. Halder et al., 2010 [117]).

1.2 Main thesis purpose

Recent studies have led to evaluate an acoustic BCI design based on a three-stimulus oddball paradigm [76, 69]. This paradigm would permit the ALS patients a binary selection with high accuracies while still offering a competitive communication speed using the auditory stimulation.

One of the first studies used the words “Yes”, “No”, “Stop”, “Pass” as possible targets with three healthy subjects and three ALS patients [120]. On average the subjects achieved accuracies of 65% which results in a communication speed of 0.43-1.80 bits/min. A recent study transferred the P300 spelling matrix to the auditory modality where number were assigned to identify rows and columns. For selection, the numbers of the rows and columns were presented to the user [82]. The participants achieved 65% accuracy. Due to the high number of bits per selection the communication speed was higher (up to 2.85 bits/min) than in the study by Sellers and Donchin (2006) [120]. Most recently a similar study was designed with auditory stimulation using sounds instead of words [53] and subjects achieved on-line accuracies of 60% and off-line accuracies of 70%.

For ALS patients that are incapable of using multi-class auditory P300 BCI, applied instead of words tones as stimuli for selection. Another study was realized presenting to participants two streams of rare standard tones, each consunting an oddball paradigm from two different directions [61]. Participants were required to focus their spatial attention on only one of the auditory oddball streams to make a binary selection. Healthy subjects reached promising accuracies between 63% and 97%.

This thesis project has been realized from the input of Mauro Marchetti, researcher in Cognitive Science (Ph.D. in Cognitive Sciences at University of Padova, Department of General Psychology, Padova (Italy)), where he focused mainly his study with Brain-computer interfaces (BCI) and event-related potential (ERP) techniques for ALS-LIS patients [89, 129, 90, 124]. This thesis is a collaboration with Politecnico di Milano to study the communication improvements for patients with this types of pathologies based on auditory system stimulation. On the specifications given, the thesis is based on the auditory BCI performed by S. Halder et al. [117]. In this study a design of an oddball paradigm is used to deliver sounds to the subject presenting two targets and a series of frequent standard tones randomized in sequences. Target one and target two differed in one physical property (loudness, pitch or location). The experiment aimed at determining which type of stimulus characteristics chosen for discriminating between target 1 and 2 would lead to the best discrimination and thus, to the highest information transfer rate (ITR). Selections were made by focusing on either one of the two targets.

The main purpose of the thesis has been to develop a portable acoustic P300 BCI for Android tablets (extending the “Progetto ON” developed by Politecnico di Milano in collaboration with Info Solution S.p.A. [127] based on visual P300 BCI communication for Amyotrophic Lateral Sclerosis [46]), to allow YES-NO communication to Amyotrophic lateral sclerosis (ALS) patients, similar to the one proposed by S. Halder et al. [117]. The application interface has been designed according to the researcher specifications and developed for delivering acoustic stimuli (e.g., single tone, complex tones, single words, etc.) via headphones to participants while their EEG is recorded.

The results have been tested on participants to reproduce the study (S. Halder et al., 2010) and verify the correct functioning of the application designed for the auditory oddball brain-computer interface. At this purpose we have collected and validated several data of healthy subjects in the AI & R Lab (Artificial Intelligence and Robotics Laboratory of Politecnico di Milano - Como Campus, Como (Italy)) and the classification protocol is tested obtaining results.

1.3 Thesis structure

This thesis consists of 5 chapters structured to drive the reader in understanding the choices made and the methodologies used for the study. At the end there is an appendix with the user manual and a complete example of use with the application.

This first chapter provides a brief description of the research area of brain-computer interfaces for ALS-LIS patients, the reasons for a new interface in order to improve the communication speed in the final stage of the disease, and the aim and motivation of this thesis.

The second chapter explains better the psychological background of the thesis. In the first section there are some basic knowledge of biology regarding the brain and its nature. In particular basic sound characteristics and sound propagation, the human auditory system, and some basic notion of psychoacoustics are presented. Other sections show modern techniques of neural activity recording and the use of the electroencephalogram (EEG). The research of the oddball paradigm to make a binary selection on acoustic stimulation and why it is better performed than the visual one to ALS patients at the end of their disease is explained. The last two sections are concerned with the definition of a brain-computer system and they explain the EEG signal processing, the possible classification algorithms used in the BCI field, and the state of the art of current BCI technology applied to ALS patients for communication.

The third chapter provides an introduction of the study made in this thesis and it explains in detail how the system is designed, the hardware composition used, and the connection scheme for this type of acoustic brain-computer interface applied. The fourth chapter contains the implemented logic project of the application and the algorithm developed, while the fifth chapter explains our genetic algorithm used for the classification in order to obtain results.

In the sixth chapter experimental data acquisition and results obtained to validate the study and the classification protocol are presented, while the last chapter (seventh) is dedicated to results discussion and the possible future developments that may will improve the project that we propose in this thesis.

Appendix A contains the user manual which describes the application installation, the main elements with the cables wiring diagram, how to prepare the participant for the training acoustic session and how to send the configuration session files to a classificator to get the results offline.

Chapter 2

Auditory P300-based BCI

"With the power of soul, anything is possible"

Jimi Hendrix

As mentioned in the first chapter, working with brain-computer interfaces in Computer Science filed requires a fundamental basic knowledge about the communication path that goes through the signals produced by neural activity with the stimulation of human auditory system. Section 2.1 contains elementary notions of brain nature and its fundamental structure, introducing the aspects of senses and human perceptions through the hearing stimulation. Section 2.2 defines some important concepts of sound propagation and how the human auditory system can perceive sounds with basics notions of psychoacoustics. Section 2.3 explains the electroencephalography (EEG) technique and its application to BCI world while Section 2.4 explains in detail the Event-Related Potential P300 based on acoustic stimuli with the reference to the oddball paradigm. In Section 2.5 we explain how a brain-computer interface system is built and how the EEG signals are processed and classified in order to make an output action. At the end of the chapter (Section 2.6) we present the state of art of brain-computer interfaces applied to ALS patients. We show the studies made over the years in order improve their communication speed until today.

2.1 Basics of Clinical Neurophysiology

The brain is the body's control center managing just about everything we do. Whether we are thinking, dreaming, playing sports, or even sleeping, the brain is involved in some way [52]. A wonder of evolutionary engineering, the brain is organized into different parts that are wired together in a specific way. Each part has a specific job (or jobs) to do, making the brain the ultimate multitasker. Working in tandem with the rest of the nervous system, the brain sends and receives messages, allowing for ongoing communication.

2.1.1 The brain

The *cerebrum*, the largest part of the human brain, is associated with higher order functioning, including the control of voluntary behaviour. Thinking, perceiving, planning, and understanding language all lie within the cerebrum's control. The cerebrum is divided into two hemispheres: the right hemisphere and the left hemisphere. Bridging the two hemispheres is a bundle of fibers called the *corpus callosum*. The two hemispheres communicate with each another across the corpus callosum. Covering the outermost layer of the cerebrum is a sheet of tissue called the *cerebral cortex*. Because of its gray color, the cerebral cortex is often referred to as *gray matter*. The wrinkled appearance of the human brain also can be attributed to characteristics of the cerebral cortex. More than two-thirds of this layer is folded into grooves. The grooves increase the brain's surface area, allowing for inclusion of many more neurons. The function of the cerebral cortex can be understood by dividing it somewhat arbitrarily into zones.

In Figure 2.1 the four main section of the cerebral cortex are presented. The *frontal lobe* is responsible for initiating and coordinating motor movements, for higher cognitive skills (such as problem solving, thinking, planning, and organizing), and for many aspects of personality and emotional make-up. The *parietal lobe* is involved with sensory processes, attention, and language. Damage to the right side of the parietal lobe can result in the difficulty of navigating spaces, even familiar ones. If the left side is injured, the ability to understand spoken and/or written language may be impaired. The *occipital lobe* helps process visual information, including recognition of shapes and colors. The *temporal lobe* helping processing auditory information and integrate information from other senses. Neuroscientists also believe that the temporal lobe has a role in *short-term memory* through its hippocampal formation, and in learned emotional responses through its *amygdala*.

All these four brain sections constitute the forebrain. Other parts of the forebrain include the *basal ganglia*, which are cerebral nuclei deep in the cerebral cortex (the the *thalamus* and the *hypothalamus*). The cerebral nuclei help coordinate muscle movements and reward useful behaviors; the thalamus passes most sensory information on to the cerebral cortex af-

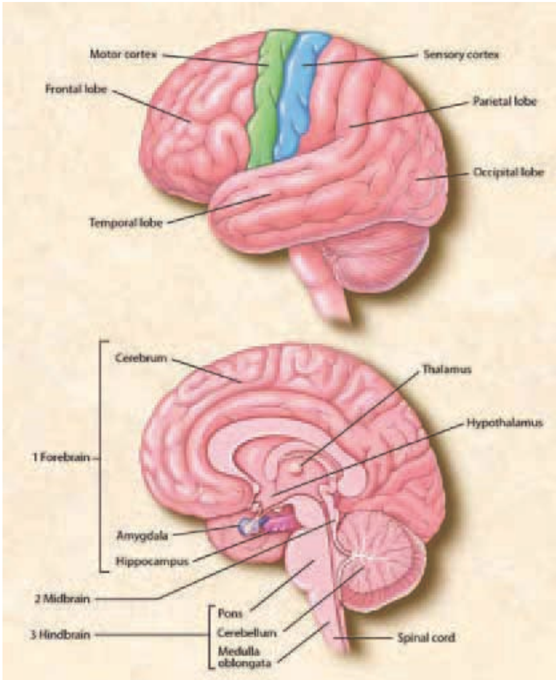


Figure 2.1: The top image shows the four main sections of the cerebral cortex: the frontal lobe, the parietal lobe, the occipital lobe, and the temporal lobe. Functions such as movement are controlled by the motor cortex, and the sensory cortex receives information on vision, hearing, speech, and other senses. The bottom image shows the location of the brain's major internal structures.

ter helping to prioritize it; and the hypothalamus is the control center for appetites, defensive and reproductive behaviors, and sleep-wakefulness.

The *midbrain* consists of two pairs of small hills called colliculi. These collections of neurons play a critical role in visual and auditory reflexes and in relaying this type of information to the thalamus. The midbrain also has clusters of neurons that regulate activity in widespread parts of the central nervous system and are thought to be important for reward mechanisms and mood.

The *hindbrain* includes the pons and the medulla oblongata, which control respiration, heart rhythms, and blood glucose levels. Another part of the hindbrain is the *cerebellum* which, like the cerebrum, also has two hemispheres. The cerebellum's two hemispheres help control movement and cognitive processes that require precise timing, and also play an important role in Pavlovian learning [103].

The spinal cord is the extension of the brain through the *vertebral col-*

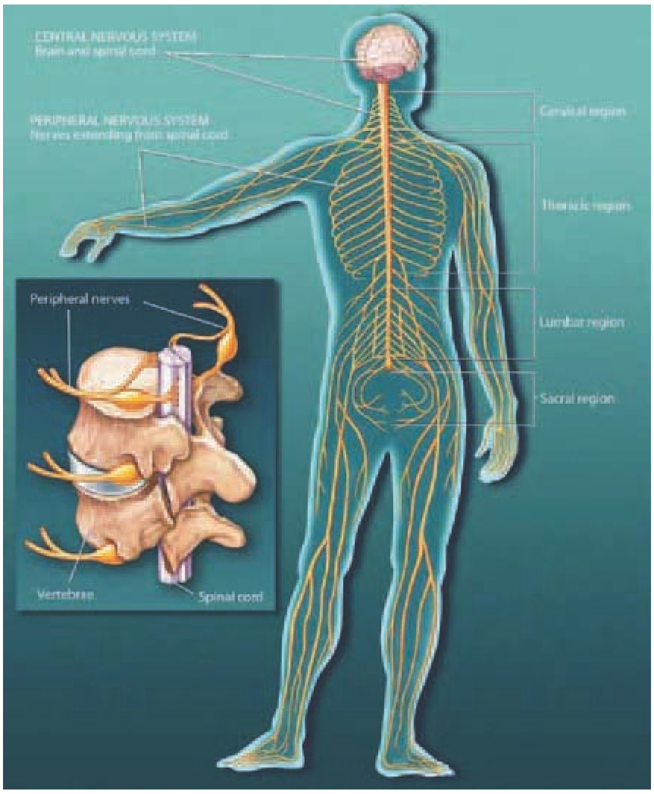


Figure 2.2: *Nervous system.*

umn. It receives sensory information from all parts of the body below the head. It uses this information for reflex responses to pain, for example, and it also relays the sensory information to the brain and its cerebral cortex. In addition, the spinal cord generates nerve impulses in nerves that control the muscles and the viscera, both through reflex activities and through voluntary commands from the cerebrum.

2.1.2 Parts of the nervous system

The nervous system has two great divisions: the central nervous system (CNS), which consists in the brain and the spinal cord, and the peripheral nervous system (PNS), which consists of nerves and small concentrations of gray matter called ganglia. The brain sends messages via the spinal cord to the body's peripheral nerves, which control the muscles and internal organs. In Figure 2.2 the nervous system is represented.

Messages are carried throughout the nervous system by the individual units of its circuitry: *neurons*. The next section describes the structure of

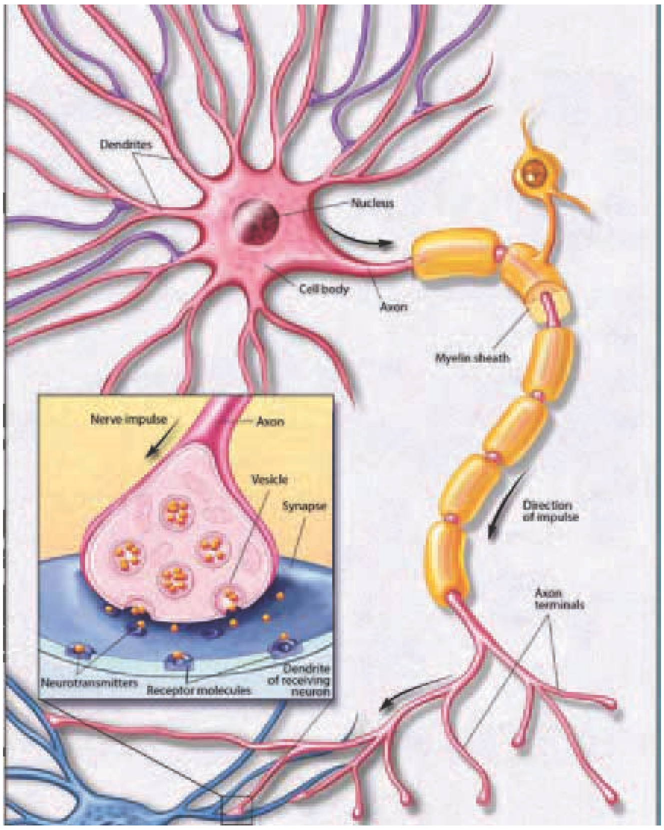


Figure 2.3: Neuron structure.

neurons, how they send and receive messages, and recent discoveries about these unique cells.

2.1.3 The neuron

Cells within the nervous system, called neurons, communicate with each other in unique ways. The neuron is the basic working unit of the brain, a specialized cell within the nervous system that transmit information to other nerve cells, muscle, or gland cells. The human brain contains between 100 million to 100 billion neurons and neuron consists of a *cell body*, an *axon* and the *dendrites*. The cell body contains the nucleus and cytoplasm. The axon extends from the cell body and often gives rise to many smaller branches before ending at nerve terminals. Dendrites extend from the neuron cell body and receive messages from other neurons. In Figure 2.3 the neuron structure is represented.

Neurons communicate with electrical and chemical signals at special contact points called *synapses* [42]. Synapses are the contact points where one neuron communicates with another. The dendrites are covered with synapses formed by the ends of axons from other neurons. When neurons receive or send messages, they transmit electrical impulses along their axons, which can range in length from a tiny fraction of an inch (or centimeter) to three feet (about one meter) or more. Many axons are covered with a layered *myelin sheath*, which accelerates the transmission of electrical signals along the axon. This sheath is made by specialized cells called *glia* and the brain contains at least ten times more glia than neurons.

Nerve impulses involve the opening and closing of *ion channels*. These are selectively permeable, water-filled molecular tunnels that pass the cell membrane and allow *ions* (electrically charged atoms) or small molecules to enter or leave the cell. The flow of ions creates an electrical current that produces tiny voltage changes across the neuron's cell membrane. The ability of a neuron to generate an electrical impulse depends on a difference in charge between the inside and outside of the cell.

When a nerve impulse begins, a dramatic reversal in the electrical potential occurs on the cell's membrane, as the neuron switches from an internal negative charge to a positive charge state. This change is called *action potential* and passes along the axon's membrane. When this voltage changes reach the end of an axon, they trigger a release of a *neurotransmitters*.

Neurotransmitters are the brain's chemical message. They are released at nerve terminals, diffuse across the synapse, and bind to receptors on the surface of the target cell (often another neuron, but also possibly a muscle or gland cell). These receptors act as on-and-off switches for the next cell. Each receptor has a distinctly shaped region that selectively recognizes a particular chemical messenger. A neurotransmitter fits into this region in much the same way that a key fits into a lock. When the transmitter is in place, this interaction alerts the target cell's membrane potential and triggers a response from the target cell, such as the generation of an action potential, the contraction of a muscle, the stimulation of enzyme activity, or the *inhibition* of neurotransmitter release. Sorting out the various chemical circuits is vital to understand the broad spectrum of the brain's functions, including how the brain stores memories, why sex is such a powerful motivation, and what makes up the biological basis of mental illness.

2.2 Hearing perception

Hearing allows us to communicate with each other by receiving sounds and interpreting speech. Like the visual system, our hearing system picks up several qualities in the signals it detects (for example, a sound's location, its loudness, and its pitch). Our hearing system does not blend the frequencies of different sounds, as the visual system does when different wavelengths of light are mixed to produce color. Instead, it separates complex sounds into their component tones or frequencies so that we can follow different voices or instruments as we listen to conversations or to music [43, 63].

The hearing mechanism is the final recipient of sounds produced by auditory system. Designers of audio systems must know the range of frequencies and the sound pressures to which this mechanism responds and the manner in which speech sounds and music must be presented to the listener if he is to gain a satisfactory amount of information and pleasure from the audio signal [147].

In Subsection 2.2.1 defines some basic characteristics of sound and how it is measured, while in Subsection 2.2.3 is explains more in detail how our auditory system perception works with some basic knowledge of psychoacoustics.

2.2.1 Physical characteristics of sound

This subsection contains basic knowledge of sound propagation and the main sound physical characteristics that stimulate our hearing perception are defined.

Sound propagation

A *sound wave* is measured as pressure variation along the direction of propagation. This distribution is defined as *sound source* and it is characterized by its *acoustic power* measured in Watt (W). The shortest distance between to corresponding points of the sound wave is called *wavelength* and it is indicated with λ. Wavelength depends by *period* T and the *speed of sound* c (344 m/s measured at 20° Celsius). The relation between these factors is:

$$\lambda = c \cdot T. \tag{2.1}$$

A periodic sound with period T is described in terms of *frequency* f measured in Hertz (Hz). The binding between the period T and frequency f is described in the Equation 2.2

$$f = \frac{1}{T}. \tag{2.2}$$

Considering the definition in the Equation 2.2 it is possible to define the relation:

$$f = \frac{c}{\lambda}, \tag{2.3}$$

which highlights how the frequency is inversely proportional to the wavelength. The speed of sound in air depends on the temperature and increases of about 0.6 m/sec per degree centigrade, while it is independent to the atmospheric pressure and sound frequency. It depends also from the medium in which it propagates.

A sound source in air is propagated in all directions and its pressure variation is expanded resulting a *spheric wave*. Otherwise if we consider a sound source propagated in a medium (e.g., a pipe), neglecting the effects at its borders, the pressure variation is propagated only along the direction of the movement giving a *plane wave*.

In non ideal situations, a wave propagation through a inhomogeneous medium gives phenomena called *diffraction*, *reflection* and *attenuation* [115]. Diffraction is given by a real source (e.g., loudspeaker) and the radiation efficiency depends by its wavelength. If the source dimension (considered as the source radius) of the loudspeaker is smaller than the wavelength, the sound is irradiated in all directions with the same efficiency and generates a spheric wave. Otherwise if the wavelength is comparable to source dimension, the sound is irradiated with different efficiency according to its direction [20].

Reflection happens whenever characteristic of the transmission medium changes and the main cause is the presence of a discontinuity in it. When a sound wave which propagates in the air meets an obstacle, this causes a partial absorption and a reflection of the incident wave on the medium where the angle of the reflection is the same as the angle of the incidence. The percentage of the wave absorbed depends on the type of material [20].

Attenuation is a measure of the energy loss of sound propagation. When sound wave is propagated in a non ideal media there is a thermal consumption of energy. Acoustic attenuation in a lossy medium plays an important role in many scientific researches and engineering fields, such as medical ultrasonography, vibration and noise reduction.

Sound intensity

The physical equivalent of sound is the change of air pressure measured in pascal (Pa). Pressure variation is linked to the perception of sound volume called *loudness*: the greater is the sound pressure, the greater is the perceived sound volume. Usually *effective pressure* ($p_{eff} = \frac{P_0}{\sqrt{2}}$) is considered as a reference, which is the root mean square of pressure variations. The minimum effective pressure that may be perceived is 0.00002 Pa, while the *threshold of pain* varies around 20 Pa, in relation to the sound frequency [20].

As explained before a sound propagating in an inhomogeneous medium looses its main characteristics form the source, so the effective sound pressure measured at the end of the medium varies in relation to its length. A sound source then is defined by its *acoustic power* measured in Watt (W). For example the acoustic power of normal speech is about $10^{-5}W$. The *acoustic intensity* (I) is defined as the average power transmitted per unit area in the direction of sound propagation:

$$I = \frac{p_{eff}^2}{\rho \cdot c} \tag{2.4}$$

where ρ is the transmissive medium density (in air at $20°$ Celsius with standard atmospheric pressure, $\rho = 1.21 Kg/m^3$), p_{eff} is the effective pressure and c is the speed of sound in the medium. Considering this values, the acoustic intensity goes from $10^{-12}W/m^2$ (hearing threshold) to $1W/m^2$ (pain threshold).

Sound Pressure Level and Intensity Level

The values of pressure, power and intensity of the acoustic sounds are distributed in a very wide range of values. For this reason, these variables are commonly expressed in logarithmic scale comparable with our auditory system perception [20].

The *sound pressure level* (SPL) is the logarithm of the ratio between the measured pressure p and the reference pressure p_{ref}

$$SPL = 20 \cdot log_{10} \cdot (\frac{p}{p_{ref}}) \tag{2.5}$$

where p_{ref} implicitly refers to the effective pressure. SPL value is dimensionless and it is expressed in *decibel* (dB) on a logarithmic scale. In order to define a reference pressure it is convenient to use the minimum audible one (0.00002 Pa). For ordinary speech p is about 0.1 Pa and using the Equation 2.5 we obtain SPL = 74 dB [9, p.52].

Similarly, also the power and the acoustic intensity are expressed in decibel using a reference value. The acoustic *intensity level* (IL) is defined as:

$$IL = 10 \cdot log_{10} \cdot (\frac{I}{I_{ref}}) \tag{2.6}$$

Being the expression of the acoustic intensity (in the Equation 2.4) containing a square, a multiplication by a factor 10 in the Equation 2.6 is considered (the squaring in logarithms corresponds to a multiplication by 2).

As for the SPL, we consider as reference acoustic intensity the minimum audible ($10^{-12} W/m^2$). and the decibel scale assumes values from 0 dB (threshold of hearing) to 120 dB (pain threshold). Although the pain threshold is around 120 dB, a prolonged exposure to high intensity sound sources can cause permanent damage to the ear (mainly young people) [95].

2.2.2 The organ of hearing

Human auditory system has a complex structure and performs significantly advanced functions. Not only it is able to process a large set of stimuli, but it can precisely identify the height or timbre of a sound, or the direction from which it came. Many functions of the auditory system are performed by what we call ear, but great emphasis is placed on the processing that takes place in the central nervous system.

As explained in Subsection 2.2.1, our ear is stimulated with sound waves [115]. This sound waves are collected by the external ear (the pinna and the external auditory canal) and funneled to the tympanic membrane (eardrum) to make it vibrate [152]. Attached to the tympanic membrane, the malleus (hammer) transmits the vibration to the incus (anvil), which passes the vibration on to the stapes (stirrup). The stapes pushes on the oval window, which separates the air-filled middle ear from the fluid-filled inner ear to produce pressure waves in the inner ear's snail-shaped *cochlea*. The separation of frequencies occurs in the cochlea, which is tuned along its length to different frequencies, so that a high note causes one region of the cochlea's basilar membrane to vibrate, while a lower note has the same effect on a different region of the basilar membrane.

Hair cells topped with microscopic bundles of hairlike stereocilia (which are deflected by the overlying tectorial membrane), riding on the vibrating basilar membrane. Hair cells convert mechanical vibration to electrical signals, which in turn excite the 30,000 fibers of the *auditory nerve* that carries the signal to the brainstem. Because each hair cell rides on a different part of the basilar membrane, each responds to a different frequency. As a result, each nerve fiber carries information about a different frequency to the brain. Auditory information is analyzed by multiple brain centers as it flows to the superior temporal gyrus, or auditory cortex, the part of the brain involved in perceiving sound.

In the auditory cortex, adjacent neurons tend to respond to tones of similar frequency. However, they specialize in different combinations of tones. Some respond to pure tones, such as those produced by a flute, and some to complex sounds like those made by a violin. Some respond to long sounds

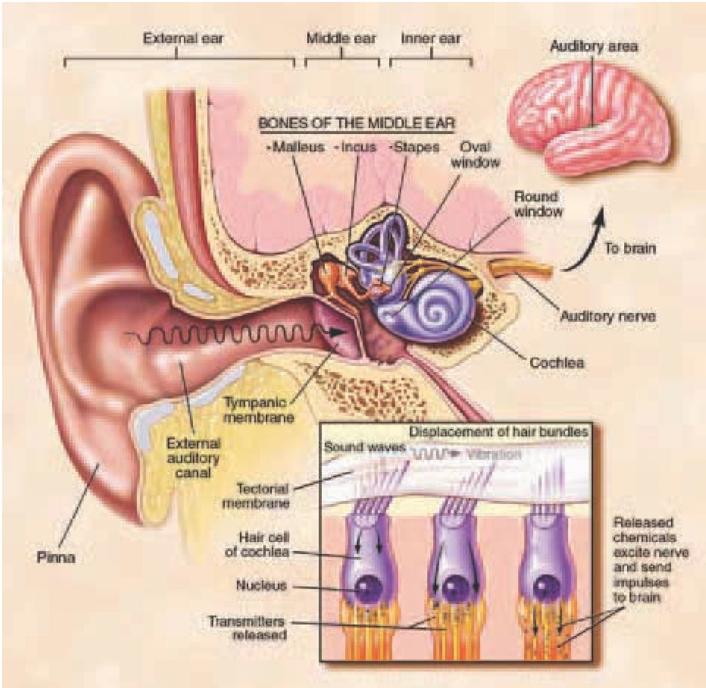


Figure 2.4: Auditory system.

and some to short, and some to sounds that rise or fall in frequency. Other neurons might combine information from these specialist neurons to recognize a word or an instrument. In other words, the basilar membrane is a wide-band mechanical filter which partially separates a complex sound into its components. As a result, a particular group of nerves is excited more vigorously by a particular frequency than by other frequencies [2].

Sound is processed in different regions of the auditory cortex on both sides of the brain. However, for most people, the left side is specialized for perceiving and producing speech. Damages to the left auditory cortex, such as from a stroke, can leave someone able to hear but unable to understand language.

We shall conveniently select a mathematical model to explain the ability of the ear to detect one tone in the presence of others tones at other frequencies. This model is the same one as we would use in describing the properties of an electrical filter for separating one frequency component from an assemblage of such components. The bandwidths of the hearing process, as measured by a person's ability to detect a pure tone in the presence of a white or pink noise, are commonly called *critical bandwidths* [51, 73, 44].

To understand how sound is perceived by our auditory system by its natural property, Subsection 2.2.3 describes more in detail elements of psychoacoustics and sound perception.

2.2.3 Elements of Psychoacoustics

Sound perception is linked to vibrations of the eardrum in the ear and are caused by small variations in air pressure. The variation of air pressure then is the physical equivalent of the sound and it can be observed by placing a sheet of paper over the cone of a loudspeaker: when a sound is emitted, the paper begins to vibrate. In fact the movement toward the outside of the speaker diaphragm determines an increase in pressure, and then pushes out the sheet of paper. Conversely the inward movement of the membrane causes a decrease in pressure and attracts the sheet toward the speaker. The eardrum has a similar behaviour: an increase of pressure pushes the tympanic membrane towards the interior, while a decrease in pressure attracts it to the outside. The movements of the eardrum are then transmitted to the cochlea that turns them into electrical impulses that are sent to the brain through the nerve endings [115].

This perception consists of various sensations that allows us to distinguish a sound from another. The main three sensations that allow the sound perception are *pitch*, *loudness* and *timbre*. These feelings are the result of the ear and the brain processing and are not directly measurable, but we can measure only their corresponding physical quantities.

For periodic sounds the pitch is determined primarily by the *fundamental frequency* that is the number of repetitions according to the pattern of the vibration. Between periodic sounds there are *pure tones* that are formed by a singular sinusoidal component. In the frequency domain, a pure tone is represented by an impulse in correspondence of the frequency of the sinusoid. For a sound composed by many harmonics (sinusoids), the fundamental frequency is the greatest common divisor of the set of frequencies that constitute the spectrum.

The perceived intensity depends on the flow of energy that accompanies the vibration. It also depends on other factors such as pitch, duration and presence of other sounds. In addition to the primary sensations (pitch, loudness and timbre), *directionality* is linked to the perception of sound, and it is linked to the phase difference by which the sound reaches the ears.

As it was explained in Subsection 2.2.2, the cochlea helps us to perceive the pitch of a pure tone. It is important to say that in front of a pure tone of a given frequency, the maximum amplitude of the basilar membrane is localized in a region of the membrane well defined. The x position of this region depends on the frequency of the sound. For each frequency there is a maximal region of sensibility of membrane called *resonance region.* The extension of audible frequencies goes from 16 Hz from 20 KHz. The extension of frequencies that ranges approximately from 20 Hz up to 4000 Hz covers approximately two thirds of the extension of the basilar membrane (12 to 35 mm from the base). The remaining portion of the frequency scale (4000-16000 Hz) is compressed in the remaining third. The range of frequencies on the abscissa axis corresponds to the first seven octaves of music, recognized as the most important in music.

The doubling of the frequency (jump of octave) of the sinusoidal stimulus, the resonance region is affected by a displacement constant of 3.5-4 mm, independently of the frequency of departure. In other words, when the frequency f is multiplied by a given value, the position of the maximum of the resonance is shifted by a certain amount, following a logarithmic type law.

Pitch

Pitch is defined as the aspect of auditory sensation in terms of which sounds may be ordered on a scale extending from low to high, such a music scale. For a music scale is intended a consideration to the *octave* as the fundamental unit: a note considered an octave to another, it means that it has twice the frequency of the other one. Pitch is a subjective quantity and it is a function of the frequency of a sound, but it is also dependent upon the sound pressure level and the composition. In order to have a consistent scale for pitch with the curve of perception of the height, the *mel* scale has been introduced. By definition 1000 Hz corresponds to 1000 mel (with 60 dB of SPL above the hearing threshold at 1000 Hz) and at each octave mel are doubled [20]. In Figure 2.5 the audible pitch trend (on the ordinate axis) is represented with the increment of frequency (on the abscissa axis).

The ability to distinguish between two almost identical stimuli is often characterized, in psychophysics studies, as a measure of minimum significant difference called *just noticeable difference* defined as *jnd*. Two stimuli are considered equal if they differ by less than jnd. The studies on the perception of pitch showed that jnd frequency depends not only on the value of the initial frequency of the stimulus, but also on the sound intensity, its duration and the rate of change of the frequency.

It was shown that with the increase of the intensity level of the stimulus from 40 to 90 dB, for frequencies above 1000 Hz the perceived pitch increases, while for frequencies below 1000 Hz it decreases from its initial intensity. If

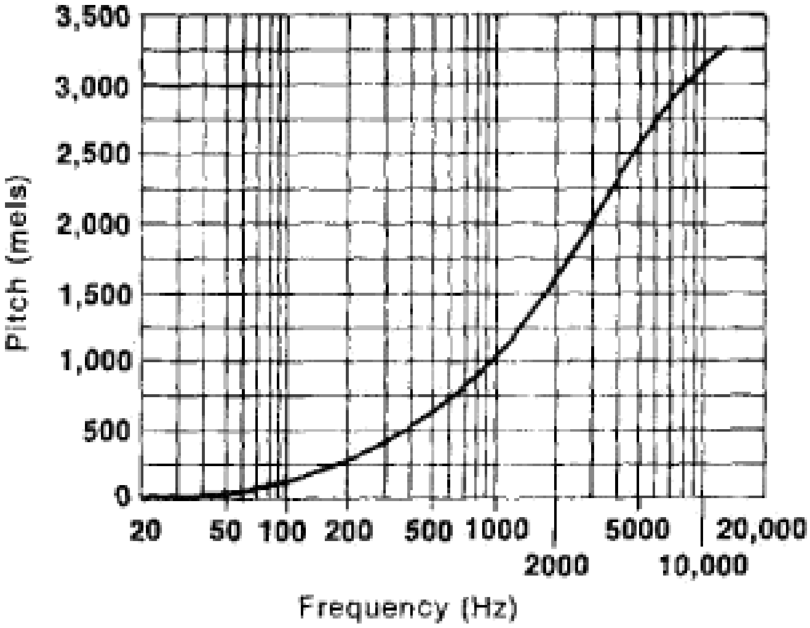


Figure 2.5: Pitch perception with frequency.

the stimulus is around 1000 Hz, the change is almost zero [138].

Loudness level

The ear is sensitive to an extremely wide dynamic range of power intensity (from 10^{-12} to $1\frac{W}{m^2}$). The choice of the logarithmic scale offers a considerable values compression and these values are referred to a reference value (minimum audible intensity). Subsection 2.2.1 defines a logarithmic scale with the acoustic intensity level as a function of sound pressure level (SPL). It is observed that continuous pure tones (characterized by the same SPL but at different frequencies) produce different sensations of intensity. So it is necessary to find experimentally the values of equal perceived intensity at different frequencies considering a reference SPL at 1000 Hz [50]. This result is shown in Figure 2.6 and represents *curves of equal loudness*.

A pure tone with 50 dB of SPL at 1000 Hz is considered *flat* while is just audible at 60 Hz. In order to produce the same sensation of intensity at lower frequency, more energy is necessary than that is required to produce the same sensation to the reference frequency at 1000 Hz.

The *loudness level* of sound at frequency f is given by its SPL at frequency 1000 Hz that determines the same perception of the intensity. This levels are measured in *Phon*.

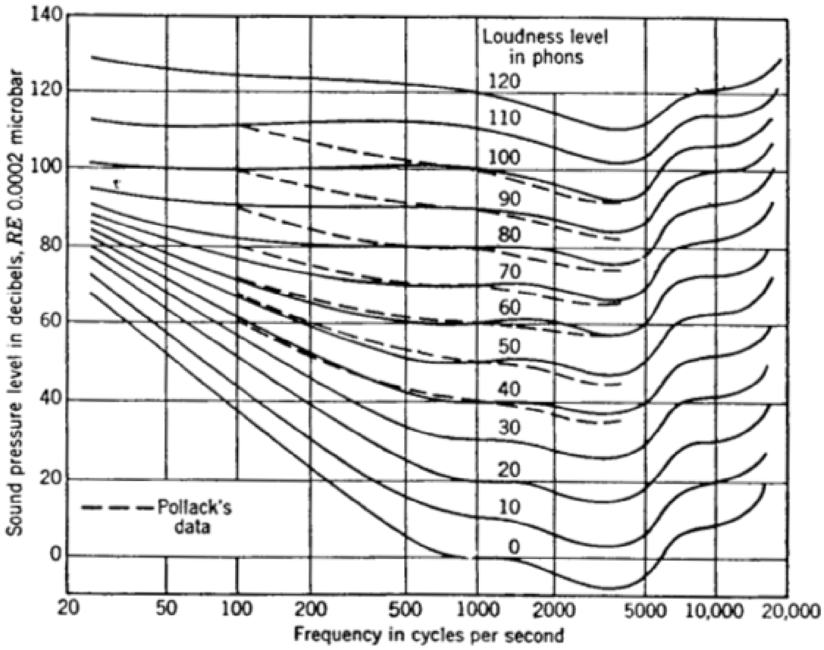


Figure 2.6: Curves of equal-loudness for pure tones. The dotted lines are equal-loudness contours for bands of noise.

Timbre perception

Timbre is used to denote the "quality" or the "colour" of sound. Timbre is the attribute that allows us to judge two different sounds that have the same intensity, the same pitch and the same duration. It is determined from the power spectrum of the stimulus, the waveform (phase), and the intensity and temporal characteristics (duration) of the sound.

2.2.4 Processing of the auditory stimulus in nervous system

To perceive a complex tone, with the excitation of the basilar membrane, our auditory system uses a complex reconstruction mechanism of the fundamental frequency with an analysis system of neural information. The first transfer function of information located on the basilar membrane to the central nervous system is performed by the hair cells. Every time the membrane is excited they are subject to a mechanical stress in the corresponding zone and then causes an electrical pulse in the nerve endings each time that this solicitation exceeds a certain threshold. The cells that form the nerve endings, which are the basic element processing in the nervous system, are the neurons (Section 2.1.3 on page 11).

When a pure tone causes motion in correspondence of a resonance area of the basilar membrane, hair cells are stimulated and they cause a train of electrical impulses in the nerve fibers connected to them. A nerve fiber of the auditory nerve is able to transmit the resonance position on the membrane (each nervous fiber is associated to one zone) and the temporal distribution of the impulses with its periodicity and its wave form configuration.

In order to determine time delays and then to process the location of the information of the sound source, the nervous system uses the *cross-correlation* between neural signals from the two ears.

2.3 Measuring brain activity

Brain activity produces electrical and magnetic activity. Therefore, to study this type of activity, sensors can detect different types of changes in electrical or magnetic activity, at different times over different areas of the brain. Brain imaging techniques allow doctors and researchers to inspect activity within the human brain, without invasive neurosurgery. There are a number of accepted, safe imaging techniques in use today in research facilities and hospitals throughout the world.

- *Functional magnetic resonance imaging* (fMRI) is a non invasive imaging method based on non electrical brain signals that can be used for BCIs. It works by detecting the changes in blood oxygenation and flow that occur in response to neural activity. When a brain area is more active, it consumes more oxygen and to meet this increased demand blood, flow increases to the active area. fMRI can be used to produce activation maps showing which parts of the brain are involved in a particular mental process.

- *Computed tomography* (CT) scanning builds up a picture of the brain based on the differential absorption of X-rays. During a CT scan the subject lies on a table that slides in and out of a hollow, cylindrical apparatus.

- *Positron Emission Tomography* (PET) uses trace amounts of short-lived radioactive material to map functional processes in the brain. When the material undergoes radioactive decay a positron is emitted, which can be picked up be the detector. Areas of high radioactivity are associated with brain activity.

- *Magnetoencephalography* (MEG) is an imaging technique used to measure the magnetic fields produced by electrical activity in the brain via extremely sensitive devices known as SQUIDs. These measurements are commonly used in both research and clinical settings. There are many uses for the MEG, including assisting surgeons in localizing a pathology, assisting researchers in determining the function of various parts of the brain, neurofeedback, and others.

- *Near infra-red spectroscopy* (NIRS) is an optical technique for measuring blood oxygenation in the brain. It works by shining light in the near infra-red part of the spectrum (700-900nm) through the skull and detecting how much the re-merging light is attenuated. How much the light is attenuated depends on blood oxygenation and thus NIRS can provide an indirect measure of brain activity.

- *Electroencephalography* (EEG) refers to recording of electrical activity from the scalp with electrodes. Being this technique the one used for the study of the thesis it is explained more accurately in Subsection 2.3.1 and 2.3.2.

2.3.1 Electroencephalography

The measurement of brain electrical activity using the electroencephalograph (EEG) provides a non invasive and inexpensive method to directly measure brain function and make inferences about regional brain activity. The purpose of this subsection is to provide an overview of the major concepts and methods associated with the use of EEG.

In 1929 Hans Berger has implemented the first recording of human brain electrical activity using the electroencephalography technique. With his first report he has demonstrated the scalp-recorded brain activity with an electroencephalographic measures [57]. In the ensuing years, a rapid developments in data collection, data reduction and data analysis have resulted in important progress in this area [17].

Berger used two large pad electrodes soaked in saline, one placed over the forehead and the other placed at the back of the head. He observed that there were regular rhythmic waves at about 10Hz in relaxed adults and noticed that these waves were best seen when subjects had their eyes closed in the absence of stimulation or other mental activity such as imaging or problem solving. These waves become known as *alpha* waves and are explained more in Subsection 2.3.2. Berger during its work confirmed the important fact that scalp and direct recordings were essentially identical in form except that the amplitude at the scalp was attenuated.

After a few years, however, physiologist Adrian & Matthews (1934) [1] also observed human EEG activity, which Jasper & Carmichael (1935) [70] and Gibbs, Davis, & Lennox (1935) [54] confirmed the details of Berger's observations. These findings led to the acceptance of the EEG as a real phenomenon.

Over the ensuing decades, the EEG proved to be very useful in both scientific and clinical applications. In its raw form, however, the EEG is a very coarse measure of brain activity, and it is very difficult to use it to assess the highly specific neural processes that are the focus of cognitive neuroscience. The drawback of the EEG is that it represents a mixed up conglomeration of hundreds of different neural sources of activity, making it difficult to isolate individual neuro-cognitive processes. However, the neural responses associated with specific sensory, cognitive, and motor events, are embedded within the EEG and it is possible to extract these responses from the overall EEG by means of a simple averaging technique (and more sophisticated techniques, as well). These specific responses are called *event-related potentials* (ERP in Section 2.4 on page 30) to denote the fact that they are electrical potentials associated with specific events.

Descriptive characteristics of the EEG

The EEG can be characterized with respect to many different parameters. However, the most common used to characterize it are frequency and amplitude.

The EEG recording electrodes and their proper function are critical for acquiring appropriately high quality data for interpretation. Many types of electrodes exist, often with different characteristics. Basically the following types of electrodes exist:

- disposable (gel-less, and pre-gelled types)
- reusable disc electrodes (gold, silver, stainless steel or tin)
- headbands and electrode caps
- saline-based electrodes
- needle electrodes

Commonly used scalp electrodes consist of Ag-AgCl disks, 1 to 3 mm in diameter, with long flexible leads that can be plugged into an amplifier [15].

AgCl electrodes can accurately record also very slow changes in potential [45].

In 1958, the International Federation in Electroencephalography and Clinical Neurophysiology adopted a standardisation for electrode placement called *10-20 electrode placement system* [71]. This system standardized physical placement and designations of electrodes on the scalp. The head is divided into proportional distances from prominent skull landmarks (nasion, preauricular points, inion) to provide adequate coverage of all regions of the brain. Label 10-20 designates proportional distance in percent between ears and nose where points for electrodes are chosen. Electrode placements are labelled according adjacent brain areas: F (frontal), C (central), T (temporal), P (posterior), and O (occipital). The letters are accompanied by odd numbers at the left side of the head and with even numbers on the right side. In Figure 2.7 the main standard schematic of 10-20 electrodes placement system is represented.

As it is known from Subsection 2.1.1 on page 8, different brain areas may be related to different functions of the brain and each scalp electrode is located near certain brain centres. Precisely, in frontal zone *Fp1* is located in the area of attention, while *Fp2* in judgement area. *F7* is located near centres for rational activities in the area of verbal expression, *F3* near motor planning purpose, *Fz* near the intentional and motivational centres and it is placed in the area of working memory, *F4* in the area of motor planning for left upper extremity to make the plan and *F8* close to source of emotional impulses such as anger, joy and happiness.

Cortex around *C3*, *C4* and *Cz* locations, deals with sensory and motor functions. In particular C3 is located in the area of right sensorimotor while C4 in the left area (upper arms, hands) and Cz for mid/left size.

In temporal zone near *T3* (verbal memory, visual memory) and *T4* (emotional memory) are located emotional processors, while at *T5* (verbal understanding), *T6* (emotional understanding) certain memory functions stand.

In parietal zone near *P3* (special temporal information for verbal reasoning), *P4* (special temporal information for math word problem and nonverbal reasoning) and *Pz* contribute to activity of perception and differentiation on cognitive processing. In the occipital zone can be found the primary visual areas below *O1* and *O2*.

High impedance can lead to distortions which can be difficult to separate from actual signal. It may allow inducing outside electric frequencies on the wires used or on the body. In order to prevent signal distortions impedances at each electrode contact with the scalp should all be bellow 5 K Ohms, and balanced within 1 K Ohm of each other.

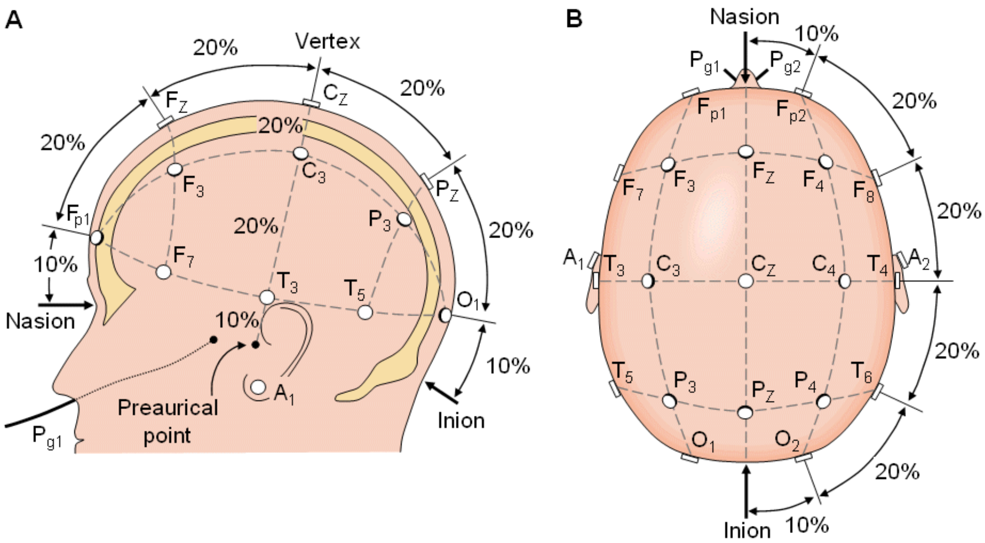


Figure 2.7: Placement of electrodes for non invasive signal acquisition using EEG. This standardized arrangement of electrodes over the scalp is known as the International 10/20 system and ensures ample coverage over all parts of the head. The exact positions for the electrodes are at the intersections of the lines calculated from measurements between standard skull landmarks. The letter at each electrode identifies the particular subcranial lobe (FP Prefrontal lobe, F Frontal lobe, T Temporal lobe, C Central lobe, P Parietal lobe, O Occipital lobe). The number or second letter identifies its hemispherical location (Z: denotes line zero and refers to an electrode placed along the cerebrum's midline; even numbers represent the right hemisphere; odd numbers represent the left hemisphere; the numbers are in ascending order with increasing distance from the midline).

Several different recording reference electrode placements are mentioned in the literature. Physical references can be chosen as vertex (Cz), linked-ears, linked-mastoids, ipsilateral-ear, contralateral-ear, C7 reference, bipolar references, and tip of the nose. Reference-free techniques are represented by common average reference, weighted average reference, and source derivation. Each technique has its own set of advantages and disadvantages. The choice of reference may produce topographic distortion if relatively electrically neutral area is not employed. Linking reference electrodes from two earlobes (electrodes *A1* and *A2*), or mastoids reduces the likelihood of artificially inflating activity in one hemisphere. Nevertheless, the use of this method may drift away "effective" reference from the midline plane if the electrical resistance at each electrode differs [75]. Cz reference is advantageous when it is located in the middle among active electrodes, however for

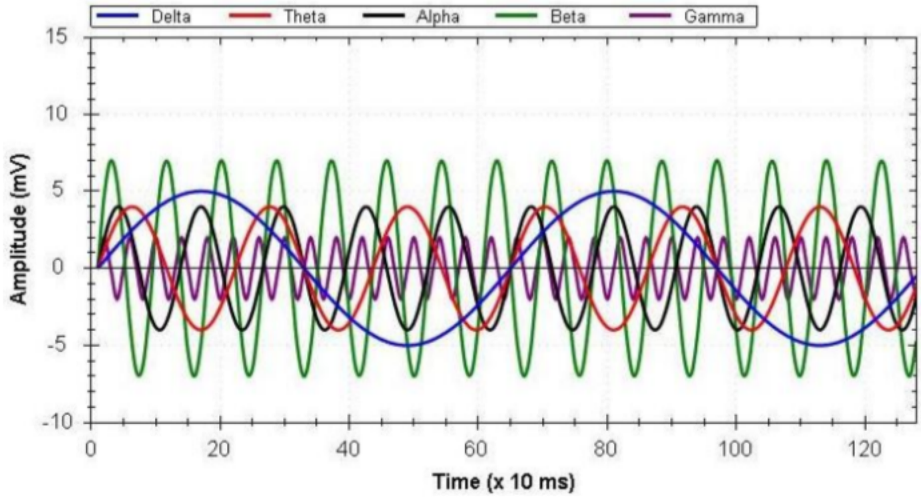


Figure 2.8: Five main frequency bands of human brain waves activity.

close points it makes poor resolution. Reference-free techniques do not suffer from problems associated with an actual physical reference. Referencing to linked ears and vertex (Cz) are predominant.

With modern instrumentation, the choice of a ground electrode plays no significant role in the measurement [101]. Forehead (Fpz) or ear location is preferred [22], but sometimes wrist or leg is also used. The acquisition of a channel is the combination of all active electrodes with reference and ground electrodes.

2.3.2 Analysis of EEG signals

As said in Subsection 2.3.1, Hans Berger found that different electrical frequencies could be linked to actions and different stages of consciousness. This was done by observing subjects performing different task, like solving mathematical problems, while recording their EEG. In Figure 2.8 the main graph with the five frequency bands of human brain waves activity is represented in order to show their relations. On the ordinate axis the amplitude, measured in mV, is represented while on the abscissa axis time is expressed in ms.

To obtain basic brain patterns, subjects are instructed to close their eyes and relax. Brain wave shapes are commonly sinusoidal, they are measured from peak to peak and normally their range goes from 0.5 to 100 μV in amplitude. The contribution of sine waves with different frequencies visualized

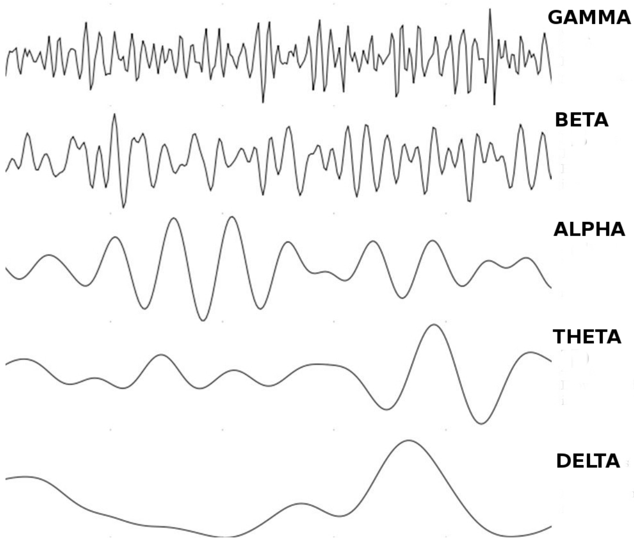


Figure 2.9: Brain waves graph. Gamma waves are for hyper brain activity which is great for learning. Beta waves are busily engaged in activities and conversation. Alpha waves becomes in very relaxed state and deepening into meditation. Theta waves are in drowsy and drifting down into sleep and dreams. Delta waves for deeply asleep and not dreaming.

in Figure 2.8, is obtained by means of Fourier Transform from the power spectrum of the raw EEG signal. Although the spectrum is continuous, ranging from 0 Hz up to one half of sampling frequency, the brain state of the individual may make certain frequencies more dominant.

Brain waves have been categorized into five basic groups represented in Figure 2.9:

- Gamma (from 30 to 120 Hz): reflect the mechanism of consciousness. Beta and gamma waves together have been associated with attention, perception, and cognition. They are associated with the formation of ideas, language and memory processing, and various types of learning [23, 16, 93].

- Beta (from 12 to 30 Hz): are often divided into β_1 and β_2 to get a more specific range. These waves are small and fast, associated with focused concentration and best defined in central and frontal areas. Many people lack sufficient beta activity, which can cause mental or emotional disorders such as depression and ADHD (Attention-Deficit/Hyperactivity Disorder) [14, 41] and insomnia. Stimulating beta activity can improve emotional stability, energy levels, attentiveness and concentration [123, 65, 55].

- Alpha (from 7.5 to 12 Hz): can be usually observed better in the posterior and occipital regions with typical amplitude of about 50 μV (peak-peak). Alpha activity is induced by closing the eyes and by relaxation, and inhibited by eye opening or thinking and calculating. Most of the people are remarkably sensitive to the phenomenon of "eye closing", i.e. when they close their eyes their wave pattern significantly changes from beta into alpha waves. Many studies monitoring the EEG activity of experienced meditators have revealed strong increases in alpha activity [19]. Alpha activity has also been connected to the ability to recall memories, lessened discomfort and pain, and reductions in stress and anxiety [153, 72, 100, 60].

- Theta (from 3.5 to 7.5 Hz): are linked to inefficiency, daydreaming, and the very lowest waves of theta represent the fine line between being awake or in a sleep state. Theta is also a very receptive mental state that has proven useful for hypnotherapy, as well as self-hypnosis using recorded affirmations and suggestions [151, 119].

- Delta (from 0.5 Hz to 3.5 Hz): are the slowest waves and occurs for deep, dreamless sleep. Delta is the slowest band of brainwaves. When your dominant brainwave is delta, your body is healing itself and "resetting" its internal clocks [11].

2.3.3 Practical application of EEG technology

The most used application of EEG is to observe and study records, to search for, or to understand, brain damages and disorders, like for instance epilepsy [139]. Empirical research and case studies throughout the decades have led to functional brain localization (see Figure 2.10), that combined with electrodes placed according to the 10-20 system (see Figure 2.7 on page 26) makes activity in these areas observable.

The study of brain waves and how they relate to different mental states, has led to number of alternative methods and beliefs on how to manipulate these waves. For instance, in order to become, e.g., more relaxed, focused and smarter, it is possible to listen a music that plays in specific frequencies bringing the brain waves working into a relaxed state [156]. Besides this somewhat regarded pseudo-science, there have been a lot of interesting studies of mental states and how they are effected, reported for instance in Braverman [13]. According to his article, an increase of alpha activity

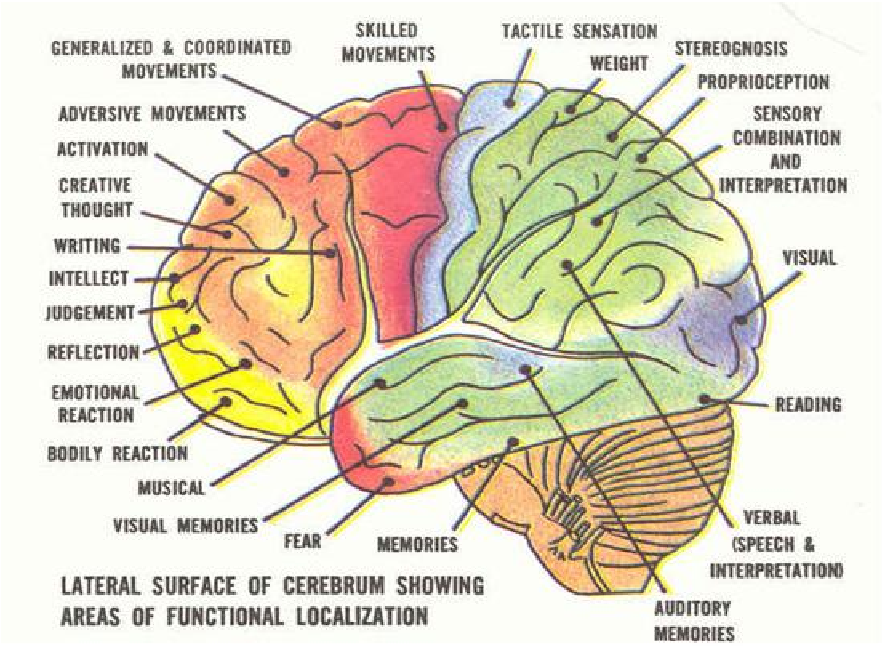


Figure 2.10: Basic functional brain localization map.

is found when taking antidepressants, and addictive drugs like morphine, heroin and marijuana. It has also been identified that drug users often lack a natural amount of occurring alpha waves, and thus this can explain why they become addicts. Alcoholics have been found to have an excess of occurring beta waves, and that this can inhibit their ability to relax. Alcohol research shows that its use increases the amplitude of the slow waves frequencies and decrease the fast waves.

Braverman talks about "how brain waves symbolizes the various parts of our consciousness, and that if we get the knowledge and treatment to change them, we can get closer to get our very balanced brain waves, or happiness". One way to get knowledge is to use ERP (Event-related Potential) discussed below in Section 2.4.

2.4 The Auditory P300 Event-Related Potential

An event-related potential (ERP) (or Evoked-Potential EP to a stimulus) is the measured brain response that is the direct result of a specific sensory, cognitive, or motor event [87]. The study of the brain through ERPs provides a non-invasive means of evaluating brain functioning as described in

Subsection 2.3.3.

ERPs are electrocortical potentials generated in the brain during the presentation of a stimulus. The stimulus could be generated by a sensor or a psychological event. It generates a time delay wave in EEG that can be detected after processing EEG signals. These methods can be simple averaging techniques, in which, EEGs are averaged over total time (time from presenting the stimulus to time when EEG settles down) or advanced approaches such as linear discriminant analysis or support vector machines (Section 2.5). There are different types of ERPs based on the source of stimulus presentation such as visual, auditory and tactile. This section discusses the P300 which is a formed from auditory evoked potential (AEP) and focuses on the P300 wave in general.

A year after Berger's publication of his findings in 1938 (Subsection 2.3.1 on page 23), Pauline Davis, in collaboration with her husband Hallowell, reported the first study on event-related potentials (ERPs) performed on awake humans in 1939 [26]. In the following decades, auditory ERP research added to the basic knowledge about sound processing in the healthy brain and it improved our understanding of the neurophysiological underpinnings of various clinical disorders.

The next major advance was the discovery of the *P3* or *P300* component by Sutton, Braren, Zubin, and John [140] (explained in detail in Subsection 2.4.5). They found that when subjects could not predict whether the next stimulus would be auditory or visual, the stimulus elicited a large positive P3 component that peaked around 300 ms post-stimulus. This component was much smaller when the modality of the stimulus was perfectly predictable.

2.4.1 The P50 component

The *P50* is a positive deflection that occurs approximately 50 ms after the beginning of sound. It is called "the earliest, the smallest in amplitude, the most variable and consequently the least studied of the auditory ERPs" [59]. Although its amplitude is relatively small, it robustly follows the beginning of all kinds of sounds, including short clicks or noise bursts. It has been argued that the P50 reflects pre-attentive arousal due to the appearance of a new event in the auditory scene. Since off-responses to sound do not comprise this "warning" element of sound, the P50 is weak, or often absent, after sound offset [59].

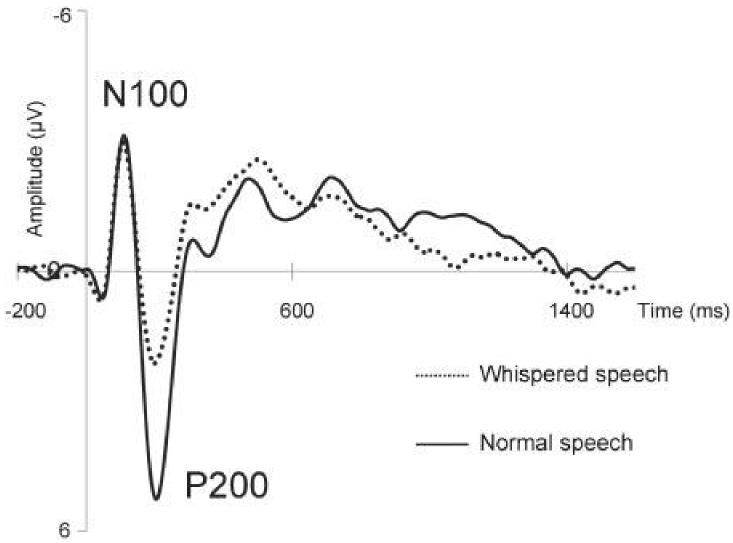


Figure 2.11: N100-P200 ERP components in response to normal and whispered speech. The P200 amplitude difference is caused by the intensity difference between the speech stimuli. Note that amplitude is depicted as "negative-up".

2.4.2 The N100-P200 component

The N100 is a relatively large, negative deflection that reliably occurs approximately 100 ms after an abrupt change in the auditory environment [67]. The N100 (see Figure 2.11) has been described as reflecting neural processes that are sensitive to stimulus features such the "quickness" of the sound's onset and offset, i.e., the sound intensity at the edges.

Hillyard in 1973 [116] investigates N100 component systematically with a shadowing task in a *dichotic listening* paradigm (well-known in the field of auditory cognition and selective attention), in which listeners are asked to verbally repeat information entering one ear while ignoring different information presented to other ear.

The P200 (see Figure 2.11) is a positive deflection occurring approximately 200 ms after the beginning of the sound. Similar to the N100, the P200 is a salient deflection that reliably occurs across individuals. The P200 amplitude varies with the physical characteristics of a sound, such as its intensity [62] and frequency [142]. It also varies with the acoustic properties of speech sounds, for example those signalling prosodic content [118]. Figure 2.11 shows the N100-P200 complex in response to normal and whispered speech [142].

2.4.3 Mismatch negativity MMN

The *mismatch negativity* (MMN) is a negative deflection in the EEG waveform that typically occurs in between 150-250 ms after the beginning of the stimulus [110]. In the oddball-paradigm (see Section 2.4.6), MMN occurs in response to an infrequent stimulus (the deviant) that occurs within a stream of frequent stimuli (the standard). MMN is considered to reflect early sensory memory processing involved in matching the incoming stimulus with that of previously processed stimuli. The occurrence of MMN does not require attentional engagement to the sounds.

Whereas the N100 amplitude, and that of other components, is larger when a participant pays attention to the stimuli [106], MMN can be obtained even when participants ignore the stimuli [88, 148] or when they are asleep [24]. In order to obtain MMN during sleep, however, the difference between the deviant and standard must be prominent.

With auditory stimuli, MMN can be elicited by an oddball stimulus that differs from the ongoing stream of stimuli with regard to, for example, intensity, frequency, and duration. Slightly more complex stimulus configurations also elicit MMN, for example when a sequence of tones that consistently increases in frequency is interrupted by a deviant with the same frequency as the former tone.

In speech, MMN research has been often used to obtain neural correlates of *voice-onset time* (VOT) discrimination in listeners (e.g., [102]). MMN is obtained through calculation of a difference wave between the waveforms induced by an equal number of frequent and oddball stimuli. The resulting waveform tends to peak around 200 ms after stimulus onset, but latency and amplitude vary with factors such as *inter-stimulus interval* (ISI), the ratio of occurrence of frequent and deviant stimuli, the difference in stimulus characteristics between frequent and deviant stimuli, and so forth.

2.4.4 The N200 component

The *N200* is a negative deflection typically evoked in between 180-325 ms following the presentation of a deviant stimulus. A subcomponent of the N200 is the auditory MNN and is called N2a [111]. The N2b component of the N200 is a component that follows the N2a and it is evoked when the participant pays attention to the stimuli and it often occurs in tandem with the P300 component (Subsection 2.4.5). The N2c is in the classification tasks [149].

2.4.5 The P300 component

The *P300* component of the event-related sensory potentials is consistently related to attention, decision making, and memory updating and therefore provides a valuable tool for investigation of these processes in the human brain [108].

The P300 was first reported over 40 years ago [140]. Its discovery comes from the confluence of increased technological capability for signal averaging applied to human neuroelectric measures and the impact of information theory on psychological research [141]. The original studies manipulated stimulus information to assess how electric brain patterns varied among conditions [6]. Subsequent results elucidated the roles of stimulus probability and task relevance, which provided the basis for its functional analysis often from data obtained with the "oddball" paradigm [38, 155].

The P300 wave also known as P3 is the most important and studied component of ERPs, which can be recorded/measured after the stimulus presentation in an EEG. The P300 is observed in an EEG as a significant positive peak 300 ms to 500 ms after an infrequent stimulus is presented to a subject. It is suggested to be related to the end of the cognitive processing, to memory updating after information evaluation or to information transfer to consciousness [39, 18].

Typical peak latency of this positive wave occurs around 300 ms for most users, therefore it is called as P300 wave.

- *single-stimulus*
- *oddball*
- *tree-stimulus*

In the typical P300-based experiments three different types of paradigms are being used.

The single-stimulus paradigm includes one type of stimuli called target. In a typical oddball paradigm, the subject is normally presented with target and standard (or irrelevant) stimuli. The three-stimulus paradigm consists of target, standard and deviant (see Subsection 2.4.6).

Figure 2.12 illustrates variants of the oddball task. The single-stimulus procedure infrequently presents the target with no other stimuli occurring (top). The traditional two-stimulus oddball presents an infrequent target in a background of frequent standard stimuli (middle). The three-stimulus oddball presents an infrequent target in a background of frequently occurring standard stimuli and infrequently occurring deviant stimuli (bottom). In

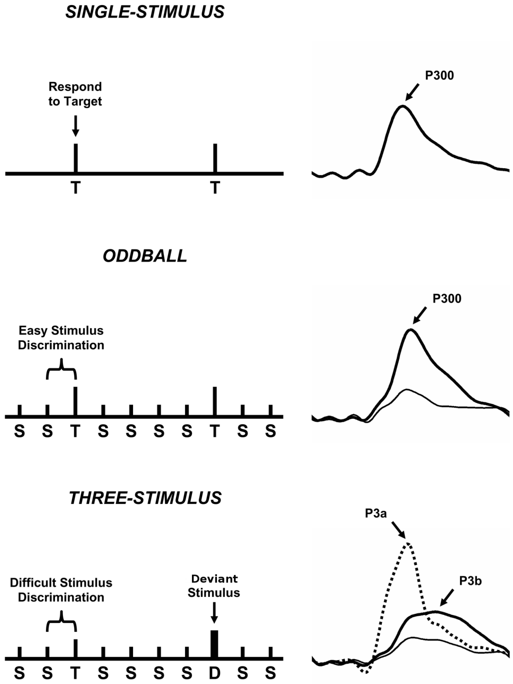


Figure 2.12: Schematic illustration of the single-stimulus (top), oddball (middle), and three-stimulus (bottom) paradigms, with the elicited ERPs from the stimuli of each task at the right [108]. The single-stimulus task presents an infrequent target (T) in the absence of any other stimuli. The oddball task presents two different stimuli in a random sequence, with one occurring less frequently than the other does (target=T, standard=S). The three-stimulus task is similar to the oddball with a compelling deviant (D) stimulus that occurs infrequently. In each task, the subject is instructed to respond only to the target and otherwise to refrain from responding. The deviant elicits a P3a, and target elicits a P3b (P300).

each case, the subject is instructed to respond mentally or physically to the target stimulus and not respond otherwise.

The P300 component is measured by assessing its amplitude and latency. Amplitude (μV) is defined as the difference between the mean pre-stimulus baseline voltage and the largest positive-going peak of the ERP waveform within a time window (e.g., 250-500 ms, although the range can vary depending on stimulus modality, task conditions, subject age, etc.). Latency (ms) is defined as the time from the beginning of stimulus to the point of maximum positive amplitude within a time window. P300 scalp distribution is defined as the amplitude change over the midline electrodes (Fz, Cz, Pz), which typically increases in magnitude from the frontal to parietal electrode

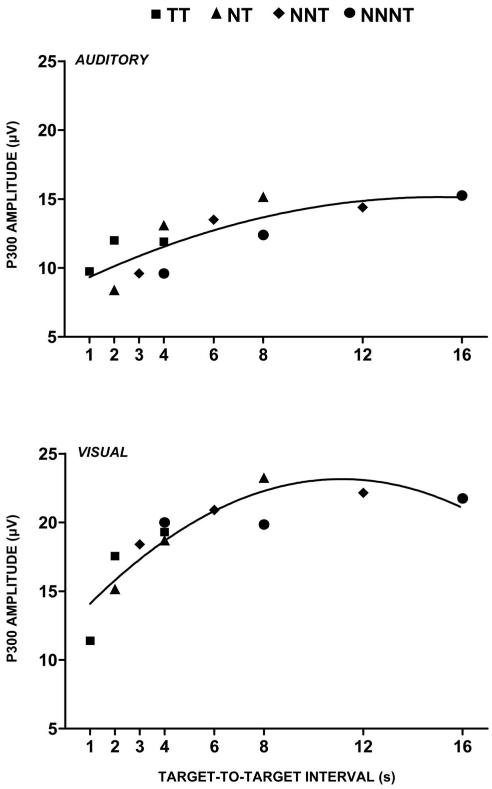


Figure 2.13: P300 amplitude plotted as a function of target-to-target interval (TTI) for the target (T) stimulus in an oddball task across sequences of preceding non-target (N) standard stimuli. The legend defines the symbols used to depict various nontarget and target sequences. The subject is instructed to respond only to the target stimulus. P300 amplitude increases independently of local sequence and global target probability. The regression lines reflect curvilinear best fit for a second order polynomial.

sites [109].

Target-to-target interval

Figure 2.13 is illustrates the influence of time-induced limitations on P300 amplitude as a function of target-to-target interval (TTI) for stimulus sequences defined by the number of non-target (standard) stimuli that occur before the detected target [18]. These findings reflect a major empirical qualification of probability and stimulus sequence on P300 outcomes [135], as TTI determines how quickly resources can be redirected to process target stimuli [58]. Short intervals produce smaller P300 components than longer intervals, with TTIs of 6-8 seconds or greater eliminating probability effects. Temporal limitations therefore may originate from memory trace development governing the event representational quality that underlies P300 generation.

This theoretical interpretation is supported by P300 findings from "single-stimulus" paradigms in which only the target stimulus occurs randomly and variably in time, This task produces P300 components comparable to the oddball paradigm. Thus, even when the target stimulus probability is unitary, the time between events is the primary determinant of P300 amplitude.

P300 amplitude

Early accounts of P300 emphasized stimulus information and probability sequence. Subsequent findings described the role of attentional resource allocation, thereby implying that cognitive demands during task processing influence P300. *Target-to-target interval* (TTI) results demonstrated that component size is small for relatively rapid stimulus presentations, whereas target stimulus items occurring at longer intervals yield maximum component amplitudes. This empirical framework is consonant with the link between P300 and attentional processing of target stimulus events-phenomena that appear related to memory processing.

P300 latency

P300 peak latency is proportional to stimulus evaluation timing, is sensitive to task processing demands, and varies with individual differences in cognitive capability. However, most studies report only a single peak and response time for specific paradigms rather than fostering a wider theoretical framework. Evaluation of normal aging and cognitive impairment have used P300 latency, but fundamental measurement issues on defining individual peaks is complicated by topographic timing variation, single-trial variability, multiple intra-component peaks, and an absence of clinical guidelines.

P3a and P3b subcomponents

Since the initial discovery of the P300, research has shown that the P300 is not a unitary phenomenon. In fact, contains two distinguishable subcomponents: the novelty P3, or P3a, and the classic P300, which has since been renamed P3b [136]. The P3a is a large, positive deflection with a front-central distribution and is typically elicited by novel or non-target stimuli (deviant or distracter) inserted in a series of standard and target stimuli in

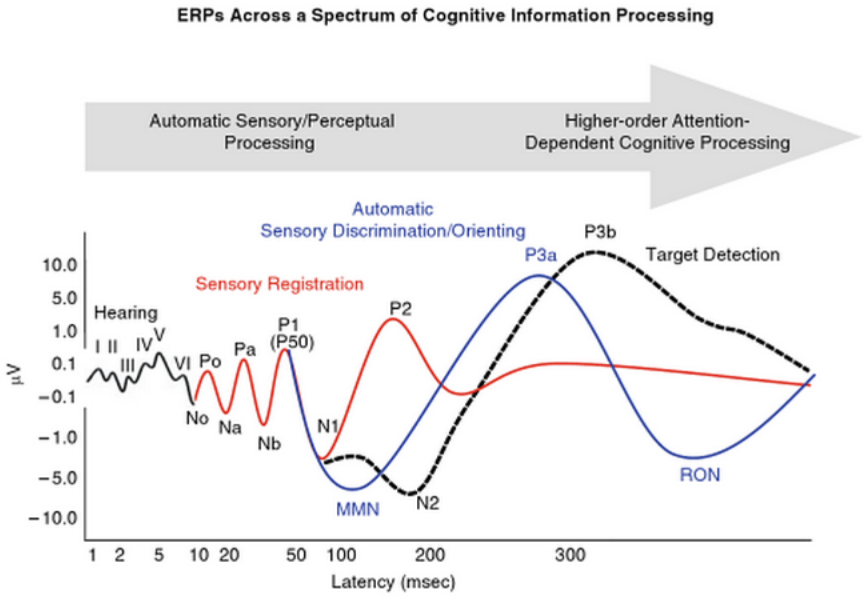


Figure 2.14: "Idealized" event-related potential waveforms representing stages of information processing representing shifts from automatic to higher order controlled processes.

a three-stimulus oddball paradigm, in which behavioural responses to that stimuli are not required.

The P3b (or classical P3) has a more posterior-parietal scalp distribution and a somewhat longer latency than P3a. By contrast, it is strongly associated with voluntary attention to the stimuli in an oddball paradigm that does require behavioural responses. The peak amplitude of the P3b varies with the relative amount of deviant stimuli, as compared to the number of standard stimuli, while stimulus relevance and the availability of attentional resources affect the amplitude as well [40].

What is also important is a distinction between P3a and P3b for both auditory and visual modalities [78]. The P3b component seems to be elicited exclusively by target stimulus, the only stimulus in the sequence required obligatory response. Was found that an infrequently presented non-target tone inserted into the traditional oddball tone sequence, elicited a parietal P3 of smaller amplitude than the target P3. This component is sometimes referred to as a "no-go" P3 since response to infrequent non-target is not required from the subject [104, 105].

ERPs provide a functional measure of neuroelectric brain activity that occurs time locked to a significant event, reflecting successive stages of information processing. Figure 2.14 represents the ERPs across a spectrum of cognitive information processing. As noted in the figure, this high temporal resolution allows for the study of the earliest stages of information processing and the subsequent transitions from automatic sensory-based perceptual processing to higher-order and integrative cognitive operations. Specifically, the amplitude and latency of the successive peaks can be used to quantify the level or the amount of processing resources and the time course of cognitive processing which may vary due to mental state, attentional demand or distraction.

2.4.6 The auditory three-stimulus oddball paradigm

The auditory P300 is typically elicited using a paradigm such that tree different stimuli are used to generate the waveform. The auditory P300 has been considered for a long time an endogenous potential, considered to be the result of a cognitive (internal) event rather than an external event [140]. This cognitive event is a decision that a target occurred. In contrast, an exogenous potential such as auditory brainstem response is the result of an external acoustic event and it is therefore influenced by physical stimulus characteristics [38].

Researchers have shown external acoustic stimulus characteristics, such as intensity, to affect P300 amplitude and latency [21]. Increasing stimulus intensity will result in an increase in P300 amplitude and will decrease P300 latency. P300 waveforms have been shown to be larger in amplitude and shorter in latency at suprathreshold levels (75 dB SPL) compared to threshold levels [98]. Above 75 dB SPL, the amplitude of the P300 does not significantly increase, indicating that the exogenous component is maximized.

When competing trains of stimuli are presented, the P300 wave is most prominent in response to stimuli that the subject is attending to. The direction of attention is usually controlled by the experimenter who requires the subject to perform a task that involves some stimuli and not others. The P300 wave generally occurs only in response to task-relevant stimuli. The perverse desire of subjects to attend to irrelevant stimuli can be prevented by making the assigned task sufficiently difficult in terms of complexity or speed that attention to irrelevant stimuli is not possible. When a subject attends to auditory stimuli in one ear and ignores auditory stimuli in the other ear, an improbable target only elicits a P300 if it occurs in the attended ear [116, 37].

The *three-stimulus oddball* paradigm is a modification of the oddball task

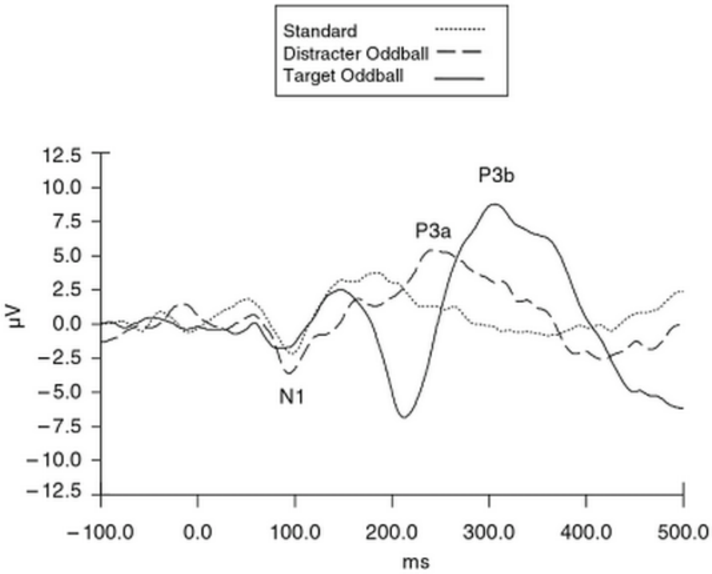


Figure 2.15: Grand average ERP components for both a target oddball stimulus and a non target deviant (distracter). Is represented in picture the variation of ERP amplitude across the N1, P3a, and P3b components in response to the two stimuli (target and deviant) that vary in contextual salience during a three-stimulus oddball paradigm.

in which rare non-target stimuli (deviants) are inserted into a sequence of rare target and frequent standard stimuli. The task given to the subject is usually to notice the presence of target stimulus and to react to it mentally or physically [77, 78].

The oddball paradigm has proven a very versatile tool in testing information processing. This is due to the fact that different ERP components are elicited to the standard and deviant stimuli that can be differentiated by their distinct relationship to the experimental conditions of the oddball paradigm employed. This is due including stimulus *probability of occurrence* as the percentage of stimuli presentation, *stimulus onset asynchrony* (SOA) as the amount of time between the onset of one stimulus and the onset of another stimulus, *interstimulus interval* (ISI) as temporal interval between the offset of one stimulus to the onset of another, and the contextual salience of the stimulus (target or deviant). The three-stimulus oddball task allows for the separate testing of the effects of non target and deviant (distracter) stimuli on behavioural performance (during active task) and may help to elucidate the source of the neural generated effect (scalp distribution).

For example, a common procedure for the *three-stimulus auditory oddball* in which pure tones that typically varies in pitch (e.g., 500Hz vs. 750Hz vs. 1000Hz) and probability of occurrence (e.g. 70% or 80% for frequent standard tone vs. 30% or 20% for infrequent target tone and deviant) are presented randomly with an inter-stimulus interval or stimulus onset asynchrony from $1000ms$ to $2000ms$. Discriminating the infrequent *target* tone from the *deviant* tone and the more frequently occurring *standard* tone, produces a robust P300 components from both stimulus types that vary in amplitude. In each task, the subject is instructed to respond only to the target. That elicits a P3b (P300) component, while the deviant elicits the P3a as shown in Figure 2.15.

2.5 Brain-computer interfaces

This section explains Brain-Computer Interfaces (BCIs) with the technical implementation in order to acquire EEG signals given by an external stimulation (i.e., an acoustic stimulation) and in particular how to extract features and classify them with a mathematical model.

2.5.1 What is a BCI?

In Subsection 2.1.2 on page 10 the role of the central nervous system (CNS) was explained and how it responds to occurrences in the environment by producing appropriate outputs. The natural outputs of the CNS are either neuromuscolar or hormonal. A brain-computer interface (BCI) is a system that measures CNS activity and converts it into artificial output that replaces, restores, enhances, supplements, or improves natural CNS output and thereby changes the ongoing interactions between the CNS and its external or internal environment [154].

Figure 2.16 shows the five kinds of applications that a BCI output might control and it illustrates each of these by showing one possible example.

- A BCI output could *replace* natural output that has been lost to injury or disease. Someone who cannot speak could use a BCI to spell words that are then spoken by a speech synthesizer. Or one who has lost limb control could use a BCI to operate a powered wheelchair [154].
- A BCI output could *restore* lost natural output. Someone with a spinal cord injury whose arms and hands are paralysed could use a BCI to control stimulation of the paralysed muscles with implanted electrodes so that the muscles move the limbs [154].

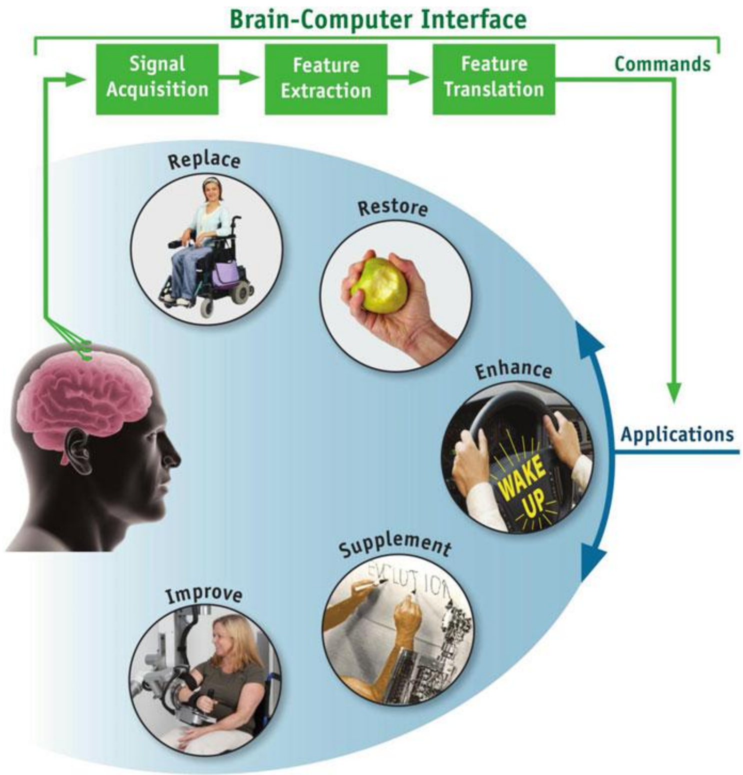


Figure 2.16: Design and operation of a brain-computer interface (BCI) system. Signals produced by brain activity are recorded from the scalp, from the cortical surface, or from within the brain. These signals are analized to measure signal features (e.g., amplitudes of EEG rhythms or firing rates of individual neurons) that correlate with the user's intent. These features are then translated into commands that control application devices that replace, restore, enhance, supplement, or improve natural CNS outputs.

- A BCI output could *enhance* natural CNS output. Someone engaged in a task that needs continuous attention over a long time (e.g., driving a car or performing sentry duty) could employ a BCI to detect the brain activity preceding breaks in attention and then produce an output (such as a sound) that alerts the person and restores attention. By preventing the periodic attentional breaks that normally compromise natural CNS output, the BCI enhances the natural output [154].

- A BCI output could *supplement* natural CNS output. Someone controlling cursor position with a standard joystick might employ a BCI to choose items that the cursor reaches. Or a person could use a BCI to control a third (i.e., robotic) arm and hand. In these examples, the

BCI supplements natural neuromuscular output with another, artificial output [154].

- A BCI output might possibly *improve* natural CNS output. For example, a person whose arm movements have been compromised by a stroke damaging sensorimotor cortex might employ a BCI that measures signals from the damaged areas and then excites muscles or controls an orthosis that improves arm movement [154].

2.5.2 The components of a BCI

A BCI detects and measures features of brain signals that reveal the user's intentions and translates these features in real time into commands that achieve the user's intent (Figure 2.16). In order to do this, a BCI system has four components:

1. *Signal acquisition*
2. *Feature extraction*
3. *Pattern recognition and Classification*
4. *Control signal/output commands*

A BCI includes an operating protocol that specifies how the onset and timing of operation is controlled, how the feature translation process is parametrized, the nature of the commands that the BCI produces, and how errors in translation are handled. A successful operating protocol enables the BCI system to be flexible and to serve the particular needs of each of its users.

The *signal acquisition* component measures brain signals using a particular kind of sensor (e.g., scalp electroencephalographic technique described in Subsection 2.3.1 or intra-cranial electrodes for electrophysiological activity, functional magnetic resonance imaging for hemodynamic activity). It amplifies the signals to enable subsequent processing, and it may also filter them to remove noise such as 60-Hz (or 50-Hz) power line interference. The amplified signals are digitized and transmitted to a computer or a device.

The *feature extraction* component analyses the digitized signals to isolate signal features (e.g., power in specific EEG frequency bands or firing rates of individual cortical neurons) and expresses them in a compact form suitable for translation into output commands.

The *pattern recognition* is provided to the feature translation algorithm and the Classification methods, which converts them into commands that

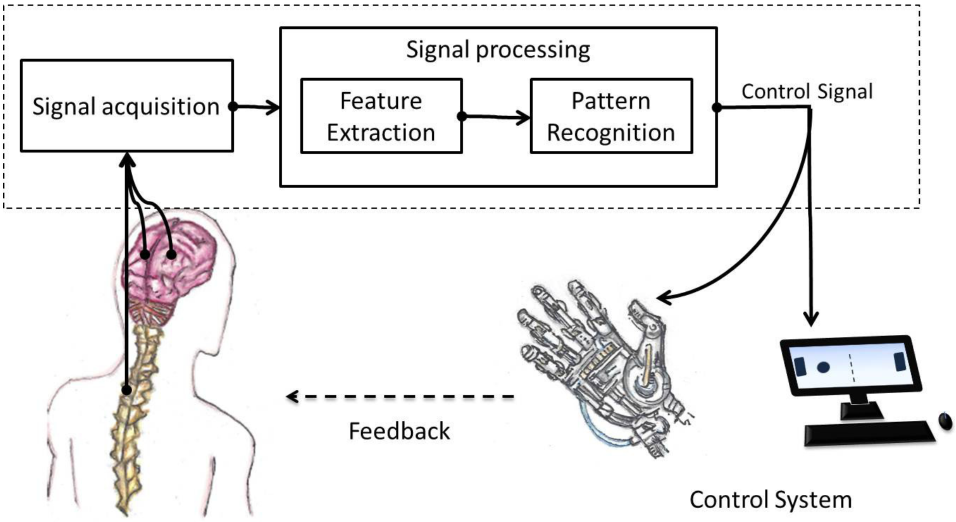


Figure 2.17: Basic components of a BCI. The image illustrates the map between the input and output through the translating algorithm. Signals are acquired by electrodes and then translated into a control signal for an external device (e.g. wheelchair, neuroprosthesis or exoskeleton) using a sequence of processing steps.

achieve the user's intent to do something. For example, a particular evoked potential measure (ERP referred in Subsection 2.4) might be translated into the selection of a letter to be added to a document being composed. The translation algorithm should be able to accommodate and adapt to spontaneous or learned changes in the user's signal features in order to ensure that the user's possible range of feature values covers the full range of device control and also to make control as effective and efficient as possible.

The *commands* or *control signal* that the feature translation algorithm produces, are the output of the BCI. This commands go to the application and there produce functions such as letter selection, cursor control, robotic arm operation, wheelchair movement, etc.

2.5.3 BCI signal acquisition

As discussed in Subsection 2.5.2, translation of intent into action is dependent on expression of the intent in the form of a measurable signal. Proper acquisition of this signal is important for the functioning of any BCI. The goal of signal acquisition methods is to detect the voluntary neural activity generated by the user, whether the signals are acquired invasively or non-invasively. Each method of signal acquisition is associated with an inherent spatial and temporal signal resolution.

At the beginning of the Section 2.3 some possible techniques that are used to acquire signals from brain activity were mentioned. In particular the EEG that is a non invasive technique (Subsection 2.3.1 on page 23) it was treated and in Subsection 2.3.2 the main brain waves that have been categorized into five basic groups as different signals bands present in the EEG signal was discussed. Electromagnetic recording from the brain at rest exhibits endogenous oscillatory activity that is widespread across the entire brain. This activity can be split into bands as referred in Figure 2.9 on page 28.

The voluntary neural activity generated by the user is obtained with the event-related potential (ERP) discussed in Section 2.4 and its main P300 component (Subsection 2.4.5). This component is used in a BCI and is acquired with the EEG in the context of the oddball paradigm (Subsection 2.4.6) as the user's focusing on precise stimulus. The most common way to derive ERP from EEG recording is aligning the signals according to the stimulus onset and then averaging them.

In order to acquire brain signal activity a *bioamplifier* is used that is a variation of an instrumentation amplifier. This device is an electrophysiological device, used to amplify the signal integrity of the brain electrical activity high performance differential amplifiers are used, and signals of interest are in the range of 0.5-100 V over the frequency range of 1-50 Hz.

The amplification process does not only depend on the performance and specifications of the amplifier, but also closely binds to the types of electrodes to attach on the subject's body. Types of electrode materials and the mount position of electrodes affect the acquirement of the signals [15] (see Subsection 2.3.1 on page 23).

2.5.4 BCI signal processing: ARX Models for Features Extraction

The goal of BCI signal processing is to extract features from the acquired signals with a mathematical model and classifying them into logical control commands for BCI applications. Based on this definition, the goal of *features extraction* for BCI applications is to obtain features that accurately and reliably reflect the intent of the BCI user.

The main purpose of the processing and extraction techniques is to characterize an item (i.e., the desired user selection) by discernible measures whose values are very similar for those in the same category but very different for items in another category. From these measures relevant features must be chosen from the numerous choices available since unrelated features can cause the pattern recognition algorithm to have poor generalization, they increase the complexity of calculation, and they require more training samples to obtain a specific level of accuracy.

The methods for extracting features depend largely on the type of neural signals used in the BCI and the characteristics associated with the underlying neural process.

Neural EEG signals features are defined by spatial location and temporal/spectral characteristics. In order to optimize the spatial information, the channels used for BCI control are usually a selected subset of channels. These can be selected with methods such as principal components analysis (PCA) [134] or independent component analysis (ICA) [7], or based on a priori knowledge of the functional organization of the relevant cortical area(s).

Identifying, Selecting and extracting the relevant properties or features of the signals that better describe the EEG signals are essential steps in the design of a BCI. The correct selection of the features is crucial, if the features extracted from EEG are not relevant and do not accurately describe the EEG signals employed, the classification algorithm will have problem in selecting the class or label the user intended.

The feature extraction could be divided in two main groups: *temporal* and *frequential/spectral* methods, a third group can be added as hybrid between temporal and frequency techniques.

Frequency methods are commonly used for the ease of application and computational speed. The different oscillations or rhythms that characterize the EEG signals present variations while performing a mental task or with a steady state evoked potential that a change in the oscillation is highly related to the stimulus frequency. The most commonly used methods are power spectral densities and band powers. It is possible to use another method that performs the Short Time Fourier Transform or the Wavelet transform to have a time-frequency representation of the signal.

In temporal methods, features present a time dependent variation and the changes can be as the ones that occur on P300 wave which depend on the focus of the selected stimulus 300ms later to be generated. The main temporal methods are *AutoRegressive parametric models* (e.g., AR). Finally the signal amplitude method that concatenates the electrodes amplitude into a feature vector, is used as input into the classification algorithm.

As explained in Subsection 2.4 on page 30, event-related potentials (like P300) are buried in the ongoing EEG. Methods are needed to extract the interesting part of the EEG (the P300 in our case) from the recording signals.

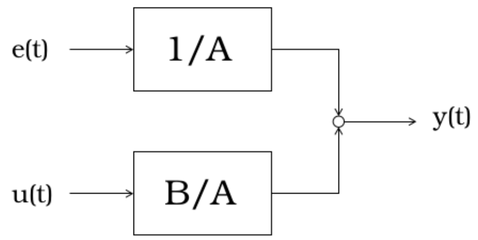


Figure 2.18: Block diagram of an ARX model.

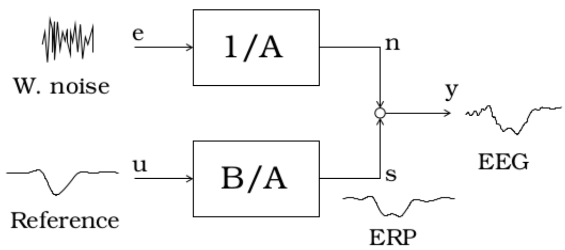


Figure 2.19: Usage of an ARX model for the modeling of ERPs.

The model described here is taken from the B. Dal Seno Phd Thesis work [121]. This method has been used to extract features in this thesis work and it is called: *ARX (AutoRegressive with an eXogenous input)*.

Figure 2.18 shows shows the block diagram of an ARX model. In this diagram, the signal y results from the superposition of a stochastic signal and a deterministic one: The first (the upper one in the figure) is the output of a process with a white noise e as input; in Figure 2.19 is the output of an ARMA (AutoRegressive, Moving Average) system with a fixed input $u(\cdot)$. In formulas:

$$y(t) = \sum_{k=d}^{q+d-1} b_k u(t-k) - \sum_{j=1}^{p} a_j y(t-j) + e(t) \tag{2.7}$$

where a_j and b_k are the coefficients of the ARMA system, q and p are their orders (i.e., the number of coefficients) and d is the delay between the input and the output. Equation 2.7 can be written also as

$$A(z)Y(z) = B(z)U(z) + E(z) \tag{2.8}$$

by using the Z transform [74]; $Y(z)$, $U(z)$, $E(z)$ are the Z transforms of $y(\cdot)$, $u(\cdot)$, $e(\cdot)$ respectively, and

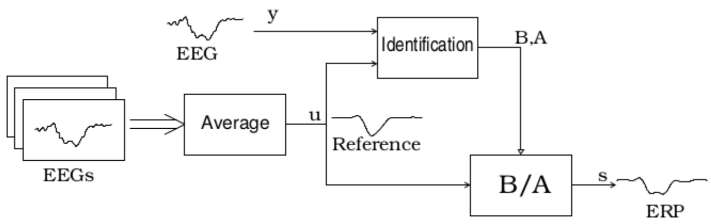


Figure 2.20: Data flow in using an ARX model for ERP extraction.

$$A(z) = 1 + \sum_{j=1}^{p} a_j z^{-j} \tag{2.9}$$

$$B(z) = \sum_{k=d}^{q+d-1} z^{-k} \tag{2.10}$$

When applied to extraction of an ERP from an EEG recording, the recording $y(\cdot)$ can be seen as a superposition of two contributions:

$$y(t) = s(t) + n(t) \tag{2.11}$$

where $s(\cdot)$ represents the ERP component, and $n(\cdot)$ represents the noisy component, i.e., the background EEG. The two components are modeled by the two blocks in the ARX models (see Figure 2.19). The ARMA block filters the reference signal to get the ERP part. The reference input to the ARMA block is a pattern that resembles the characteristics of the ERP to be detected. It is usually obtained by averaging many EEG recordings where the ERP is supposedly present. The rationale behind this is that the ERP component $s(\cdot)$ is similar but not exactly equal to the ERP average. While averaging extracts the ERP component from many recordings where an ERP is known to be present, the objective of the application of the ARX model is to extract an ERP component from a single recording. So, it can be used, for example, to discriminate between stimuli that elicited a P300 and those the did not.

How to use ARX modeling for ERP extraction is shown in Figure 2.20. The analysis works on segments (called epochs) of EEG recordings, long enough to cover the expected duration of the ERP with some margin. An initial batch of epochs of EEG recordings where the ERP is present is used to build the reference signal $u(\cdot)$, by averaging. For every new EEG recording the model in Equation 2.7 is identified, with $y(\cdot)$ being the recording to be analyzed and $u(\cdot)$ the previously computed ERP average. Identification of the ARX model is performed by using a least-squares method, which minimizes the error with a cost function defined as:

$$J = \frac{1}{N}\sum_{t=1}^{N} e(t)^2 \tag{2.12}$$

where N is the number of samples in an epoch. $e(\cdot)$ is the prediction error of the model:

$$e(t) = y(t) - \hat{y}(t) \tag{2.13}$$

where

$$\hat{y}(t) = \sum_{k=d}^{q+d-1} b_k u(t-k) - \sum_{j=1}^{p} a_j y(t-j) \tag{2.14}$$

After the identification, the useful component, $s(\cdot)$, can be computed by filtering the reference signal $u(\cdot)$ with the filter B(z)/A(z), according to Equation 2.7:

$$s(t) = \sum_{k=d}^{q+d-1} b_k u(t-k) - \sum_{j=1}^{p} a_j s(t-j) \tag{2.15}$$

The identification and filtering steps are performed separately for each candidate EEG epoch. For P300 studies, it means one epoch per stimulus.

The resulting signal $s(\cdot)$ contains the ERP when it exists, otherwise is just noise. For a BCI an automatic classification is required. In Subsection **??** on page **??**, our case has been discussed.

Before an ARX model can be used at all, the parameters p, q, and d must be chosen. Normally, the choice is made by analyzing a training data set, i.e., a set of EEG recordings for which it is known whether they contain an ERP or not. The choice of these parameters cannot be made by using a least-square criterion defined in the Equation 2.12, because ever increasing the number of the parameters always makes the model fit better. The magnitude of the contribution to the goodness of the fit of new parameters must be taken into account. *Akaike Information Criterion (AIC)* [3] in the case of ARX models it can be written as

$$AIC = 2n + N log \sigma^2 \tag{2.16}$$

where σ^2 is the variance of $e(\cdot)$ given by the Equation 2.13, n=p+q, and N is the number of samples.

2.5.5 BCI signal processing: Pattern Recognition and Classification methods

After the features have been selected the next step is to translate them into a command. This translation can use regression/classification methods. A classifier is a function that assigns labels to objects. A learning algorithm is a procedure that finds a good classifier given a set of labeled examples.

A suitable classifier function $f : X \to Y$, where X is the feature space ($X \subseteq \Re^n$) and Y a set of possible labels (e.g., $Y = \{1, +1\}$, i.e., a binary classification problem), given a probability distribution $p(\cdot, \cdot)$ defined over X x Y, and a training set of pairs $\langle x_i, y_i \rangle$ where $x_i \in X$, $y_i \in Y$, $i = 1, ..., N$.

In this subsection two of the classifiers used for this thesis work are described.

Logistic Classifier

A *logistic classifier*[83] approximates the probability $P(y|x)$ with a logistic function:

$$P(y = +1|x) = \frac{1}{1 + exp(\omega_0 + \sum_{j=1}^{n} \omega_j x_j)} \tag{2.17}$$

$$P(y = -1|x) = 1 - P(y = +1|x) = \frac{exp(\omega_0 + \sum_{j=1}^{n} \omega_j x_j)}{1 + exp(\omega_0 + \sum_{j=1}^{n} \omega_j x_j)} \tag{2.18}$$

where x_j are the n components of the vector x. The decision of the class to assign to a given sample x is taken by comparing the two probabilities $P(y = -1|x)$ and $P(y = +1|x)$. The parameter vector w can be found by maximizing its log-likelihood

$$L(\omega) = \sum_{i=1}^{N} logP(y_i|x_i, \omega) \tag{2.19}$$

by using gradient ascent. In order to improve the generalization ability of the classifier, a penalization term can be added:

$$L^{\lambda}(\omega) = \sum_{i=1}^{N} logP(y_i|x_i, \omega) - \lambda \|\omega\|^2 \tag{2.20}$$

The additional term penalizes large values of w components. The parameter λ determines how strong the penalty term is.

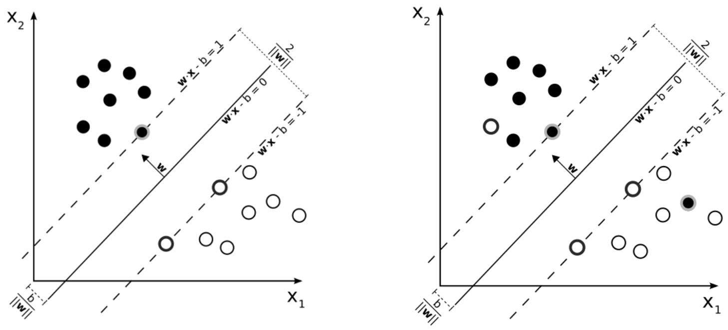


Figure 2.21: SVM: support vectors are surrounded by a circle. On left: the maximum-margin hyperplane in the separable case. On right: the maximum-margin hyperplane in the non-separable case.

Support Vector Machines

A *support vector machine* (SVM) [145] is a supervised learning method used for classification and regression. It was developed by Vladimir Vapnik in the late 1970s, while he was addressing the problem of the generalization of a classifier from a theoretical point of view.

Vapnik found theoretical bounds on the expected risk given the empirical risk. In the simplest case, an SVM is a hyperplane in the space X. This hyperplane separates the space in two regions, one for each labels. Samples are assigned labels depending on which side of the hyperplane they lie (see Figure 2.21). In formulas:

$$f(x) = sign(f^*(x)) \tag{2.21}$$

$$f^*(x) = \langle w, x \rangle + b \tag{2.22}$$

If the training set is such that there exist an hyperplane that separates exactly the positive and negative samples, the SVM maximizes the *margin* (distance) of the hyperplane from the nearest samples. Equivalently, the SVM maximizes the distance between positive and negative samples along the direction ω. Maximizing the margin, the structural risk is minimized. The training samples that are closest to separating hyperplane, i.e., those for which it holds

$$y_i(\langle \omega, x \rangle + b) = 1 \tag{2.23}$$

are called *supported vectors*. The reason is that the problem of finding the maximum margin is equivalent to minimize $\|\omega\|^2$ subject to

$$y_i(\langle\omega, x\rangle + b) \geq 1 \tag{2.24}$$

By introducing *Lagrange multipliers* $\alpha_i, i = 1, ..., N$ it is possible to show [145] that for optimum ω it holds

$$\omega = \sum_i \alpha_i y_i \omega_i \tag{2.25}$$

and $\alpha_i \neq 0$ only for the x_i that are support vectors.

If the training set is such that there is no hyperplane that separates positive and negative samples, some samples are necessarily misclassified by any hyperplane. In this case, minimizing $\|\omega\|^2$ is not enough, but the SVM has to find a trade-off between the maximization of the margin and the minimization of the errors.

A linear classifier proves to be inadequate in the majority of real cases. It is possible to extend the linear SVMs seen so far and make them non-linear in a simple and straightforward way. The trick is to map samples x in a higherdimensional space Θ by means of a non-linear mapping $\Phi : X \rightarrow \Theta$. The separating hyperplane is now to be found in Θ. A good hyperplane is more likely to exist in Θ than in X because the number of dimensions of Θ is greater than that of X, and hence data are more sparse. It is also possible for Θ to be infinite dimensional, but there must be a way to compute inner products in such a space. Inner products are needed because they appear in Equation 2.24. It turns out that is possible to avoid computing inner products with the so-called *kernel trick*.

Putting together the Equations 2.24 and 2.25 is obtained

$$f^*(x) = \langle\sum_i \alpha_i y_i x_i, x\rangle + b = \sum_i \alpha_i y_i \langle x_i, x\rangle + b \tag{2.26}$$

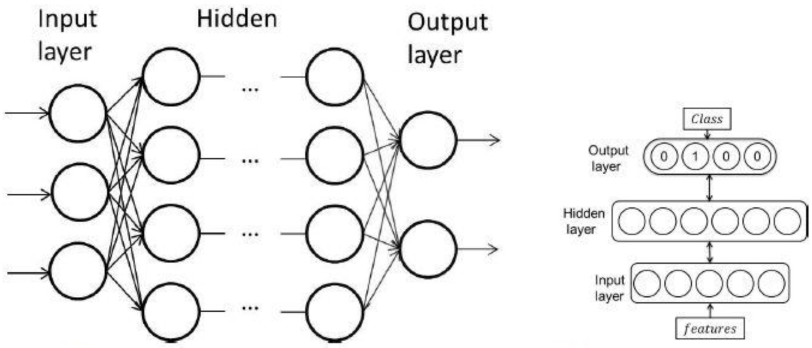
and in the mapped space Θ

$$f^*(x) = \langle\sum_i \alpha_i y_i \Phi(x_i), \Phi(x)\rangle + b = \sum_i \alpha_i y_i \langle\Phi(x_i), \Phi(x)\rangle + b \tag{2.27}$$

It is possible to find a *kernel function* κ such that $\langle\Phi(x_i), \Phi(x))\rangle = \kappa(x_i, x)$, where $\kappa(\cdot, \cdot)$ is much easier to compute than the inner product in Θ. The Equation 2.27 becomes

$$f^*(x) = \sum_i \alpha_i y_i \kappa(x_i, x) + b \tag{2.28}$$

Is possible to consider some *kernel functions examples*:



(a) Arquitecture of multilayer perceptron (b) General and Restricted Boltzmann Machines

Figure 2.22: Neural Networks architectures having Multilayer Perceptron and RBM.

- **Polynomial** $\kappa(x, y) = (1 + \langle x, y\rangle)^2$, where d is a parameter.
- **Gaussian** $\kappa(x, y) = exp(-\frac{\langle x,y\rangle^2}{2\sigma^2})$, where σ is a parameter.
- **Perceptron** $\kappa(x, y) = \tanh(b\langle x, y\rangle - c)$, where b and c are parameters.

The choice of an appropriate kernel for a given data set is still an open issue; there are no predefined rules for selecting kernels.

There are others different classification methods and they can be divided using their classifier properties into: *neural networks*, *non-linear Bayesian classifiers*, *nearest neighbor classifiers* and *combinations of classifiers* [85].

Neural Networks (NN)

Neural Networks (NN) are non-linear classifiers that use assembly of neurons to produce the boundaries. The most used technique is the Multilayer Perceptron [8], that uses an input layer where the features are included, some hidden layers which preocess it and an output layer that defines the output class (Figure 2.22).

A conventional feed-forward artificial neural network (ANN's) is a system constructed by a finite number of basic elements called *neurons*, which are grouped in layers. Every neuron is highly interconnected in the whole topology; the structure has a number of inputs and outputs that depends on the system that will be approximated. A neuron is the basic element in an artificial neural network that simulates biological neurons which receives electrical impulses which are received through its dendrites, from other neuron's axons. The ANN's are applied to approximate a non-linear system as being universal approximations.

An extension to ANN are the *Restricted Boltzmann Machines* (RBM) that have a bidirectional connection between the layers (see Figure 2.22b.).

Non-linear Bayesian classifiers

There are mainly two types of Bayesian classifiers used for BCI systems: *Bayesian classifiers* and *Hidden Markov Models* (HMM). Both these classifiers produce nonlinear decision boundaries.

Bayesian classifiers are used for BCI system which allows them to reject uncertain samples more efficiently than discriminative classifiers [85].

They assign the class to the feature vector with the highest probability. Considering an observed feature vector x, with a Bayesian statistical classifier it is possible to classify it knowing y. The Bayes' rule is used to obtain the *a posteriori* probability $P(y|x)$ that a feature vector has of belonging to a given class. Assuming for example two classes L and R corresponding to imaginary left and right movements of the hand, the a posteriori probabilities of each class are computed using Bayes' rule as in the Equation 2.29:

$$\begin{aligned} P(y|x) = \frac{P(y)P(x|y)}{P(x)} &= \frac{P(y)P(x|y)}{P(x|L)P(L) + P(x|R)P(R)} \\ &= \frac{P(x|y)}{P(x|L) + P(x|R)} \end{aligned} \tag{2.29}$$

Typically it is assumed that the *a priori* probabilities are equal ($P(y) = P(L) = P(R) = 0.5$) since it is supposed the user has no prediction for any movement.

Nearest Neighbor classifiers

In BCI systems classifiers called *k Nearest Neighbor* (kNN) are also used to assign to an unseen point the dominant class among its kNN within the training set. This algorithms may perform efficiently with low-dimensional feature vectors [85]. Another types of Nearest Neighbor classifiers used in BCI system are based on *Mahalanobis Distance* (MDist) to assign a class to a feature vector to the nearest prototype.

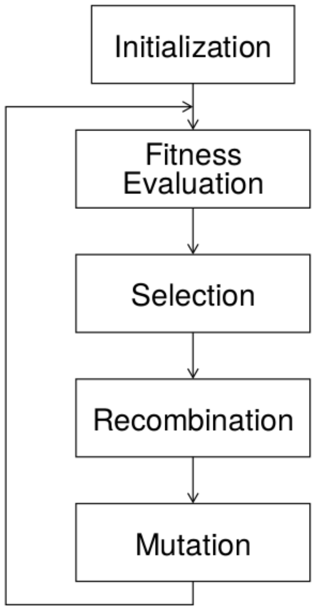


Figure 2.23: General scheme of a genetic algorithm.

2.5.6 Genetic Algorithm

Genetic algorithms (GAs) are a class of optimization algorithms that mimic the way natural evolution works. These algorithms work by considering potential solutions to the problem, evaluating them, and combining parts of good solution in order to find better candidate solutions. The range of problems solved with genetic algorithm is very wide; they have been used in scheduling, budgeting, optimization of networks, and many classical problems of operations research.

The father of genetic algorithms can be considered John Holland, who worked on them in the 1970s at the University of Michigan [64], although a group at the Technical University of Berlin (Ingo Rechenberg, Hans-Paul Schwefel, and Peter Bienert) worked on *evolution strategies* [113] at the same time, an approach similar to GAs, but different in some important aspects. The Atlantic Ocean separated the two groups, which worked independently and unaware of each other for a while. Work in the field continued since, and many variants have been developed [114].

Candidate solutions for the problem are encoded in chromosomelike data structures (called *chromosomes*) which often are just binary strings. Genetic algorithms work on a subset of all the possible solutions, which is called *population.* A genetic algorithm begins with an initial population of chromosomes, which are normally chosen randomly as in Figure 2.23. At every iteration of the algorithm, first all the solutions represented by the chromosomes are evaluated with the respect to the optimization problem. This is operation is in fact the evaluation of the so-called *fitness function.* The fitness function is a measure of the goodness of the parameters encoded in a given chromosome. Fitness values are used to select individuals from the population. The actual selection process may be done in different ways, and an individual may be selected more than once, but in any case fitter individuals have greater chance to be selected. Examples of selection are: the best n individuals; roulette wheel, where individuals are picked at random with a probability proportional to their fitness; tournament, where many independent tournaments between randomly-chosen individuals are performed, the winner of a tournament being the individual with the best fitness.

Recombination and mutation are then applied to the individuals selected in the previous step. *Recombination* (also *crossover*) is applied to the selected population in pairs: randomly-selected parts of the two chromosomes are exchanged, so as to form new, different individuals. Normally, recombination is applied only with a given probability. *Mutation* is the flipping of bits of the chromosomes (when they are binary strings). Typically, mutation is applied to all the bits of chromosomes with a very low probability (less than 1%). After mutation has been applied, a new population is ready, and the algorithm restarts from the evaluation. In GA terminology, a *generation* is the execution of evaluation, selection, recombination, and mutation.

Generation after generation, the fitness of the population increases, and thus better and better solutions are found. The process is terminated by some criterion. It could be something like until the optimum is found, but there are two problems: Maybe there is no test for optimality, or maybe the time for finding the optimum is too much, and obtaining a good yet suboptimal solution is enough. So, normally a genetic algorithm terminates after a predefined number of generations, or after a good enough solution has been found, or when no improvement has been seen for some generations.

Genetic algorithms have been defined as a class of algorithms, because even after choosing a selection scheme, a termination criterion, and the all various parameters (e.g., mutation probability), the result is still a schema of an algorithm and not an actual algorithm. The encoding of solutions in chromosomes depends on the problem at hand, and a new encoding must be devised for every new problem. The fitness function is at the heart of GAs, and it contains the description of the original optimization problem, rewritten in terms of chromosomes. Thus, there is at least one fitness function for every optimization problem, given the encoding. After defining an encoding and a fitness function, all elements are in place to run an actual genetic algorithm. Sometimes, though, the encoding for a particular problem makes use of structures that are more complicated than a plain string of bits. In such cases, recombination and mutation operators must be rewritten accordingly.

The algorithm used for this thesis project was taken from Dal Seno Doctoral Thesis [121].

2.6 Evolution of the BCI technology

BCIs offer their users new communication and control channels without any intervention of peripheral nerves and muscles. Hence, many researchers focus on building BCI applications, in the hope that this technology could be helpful for those with severe motor disabilities. Various BCI applications have been developed recently thanks to significant advances in the field of EEG-based BCI.

EEG signals are used by most BCI applications, because they offer an acceptable signal quality that combines low cost and easy-to-use equipment. Thanks to BCI applications, the quality of life of severely disabled people can be improved. Moreover, BCI applications potentially represent a powerful tool for revealing hidden information in the user's brain that cannot be expressed.

The main target population for BCI applications mainly falls into two classes. The first group includes Complete Locked-In State (CLIS) patients who have lost all motor control, because at a terminal stage of ALS or they suffer severe cerebral palsy. The second group comprises Locked-In State (LIS) patients who are almost completely paralyzed, but with residual voluntary movement, such as eye movement, eye blinks, or twitches with the lip.

Despite the use of the BCI by people who have the most need, they are increasingly used by healthy people in neuromarketing and video games as a tool to reveal affective information of the users, which cannot be so easily reported through conventional interfaces. Likewise, BCI can be used for some people that suffer from neurological disorders such as schizophrenia or depression.

Nowadays, there is a vast number of very different BCI applications, such as word processors, web browsers, brain control of a wheelchair or neuroprostheses, and games. Despite the most recent significant advances in BCI technology, there are still many challenges in employing BCI control for real-world tasks [94]. BCI applications for communication deal with

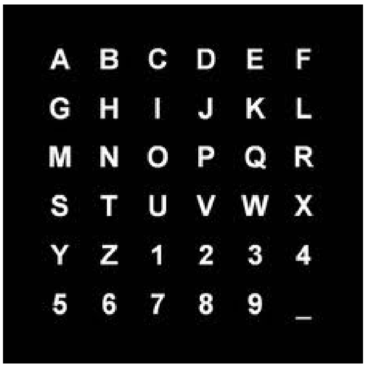


Figure 2.24: Original P300 speller. Matrix of symbols displayed on a screen computer which serves as the keyboard or prosthetic device.

severe communication disabilities resulting from neurological diseases. This kind of application probably represents the most pressing research in the field of BCI, because communication activity is essential for humans.

Applications for communication purposes outline an operation that typically displays a virtual keyboard on screen, where the user selects a letter from the alphabet by means of a BCI. The distinguishing element in each approach is usually the BCI and the type of control signal. P300-based BCIs have been proven sufficiently suitable for ALS patients in the early and middle stages of the disease [90]. Recent progress with P300-based speller have allowed the development of commercial applications available to general public [68], while for the auditory BCI there are not any system available yet for online communication.

2.6.1 ERP-based visual BCI systems

One of the best-known P300 spellers was designed by Farwell and Donchin in 1988 [47]. In this speller, the 26 letters of the alphabet with other symbols and command are displayed on-screen in a 6x6 matrix (see Figure 2.24) with randomly flashing rows and columns. Then, the user focuses attention on the screen and concentrates successively on the characters to be written, while the EEG response is monitored. Two P300 are elicited for each looked-for element on the matrix, when the desired row or column flashed, thereby allowing the system to identify the desired symbol. The Farwell-Donchin speller gets an acceptable spelling rate of about 2 characters per minute.

The speller provides an high rate and high accuracy, but its precision can be improved by reducing perceptual errors when a P300 response is elicited due to flashing rows or column adjacent to the target symbol, which is the

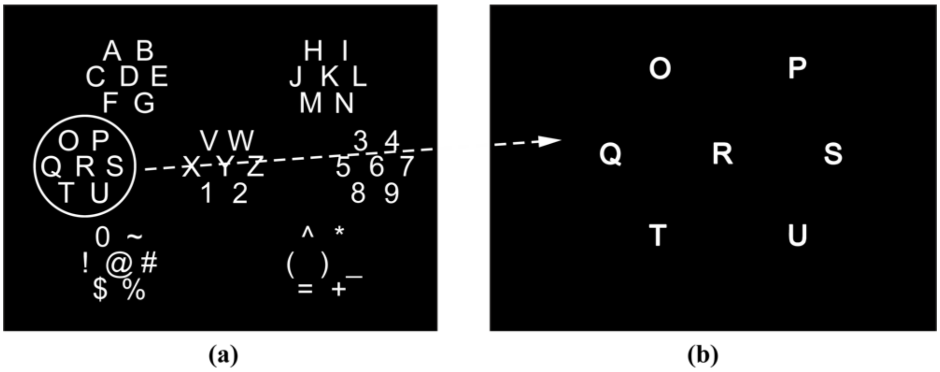


Figure 2.25: The proposed region-based paradigm for the improved P300 speller: (a) The first level of intensification where each group contains up to seven characters; and (b) One region is expanded at the second level.

major source of error. Hence in 2009 a new letter distribution was proposed to overcome this problem [48] as shown in Figure 2.25.

The idea is to have several regions flashing instead of using rows and columns. The characters are placed into a two-level distribution. At the first level, the characters are distributed into seven groups, each with seven characters, which are also flashes randomly. The group containing the target character is found by P300 detection. At second level, the characters in the detected group are repositioned and the level one procedure is repeated, and so on until the target character is final selected.

In 2010 Townsend et al. [144] presented a newly enhanced BCI based on a checkerboard paradigm instead of the standard row/column paradigm introduced by Farwell and Donchin. In this approach, the standard matrix containing targets was superimposed on a checkerboard and showed a significantly higher mean accuracy.

In 2011 Ahi et al. [143] improved the Farwell-Donchin P300 speller by introducing a dictionary to decrease the number of missclassifications in the spelling. The dictionary was used for checking the candidate word proposed by the classifier of P300 responses. In case of misspelling, the dictionary gave a certain number of suggestions from which the system could select. Additionally, in order to reduce the probability of misspelling due perceptual errors, the usual letter position in the matrix was changed according to the analysis of word similarities in the constructed dictionary.

2.6.2 ERP-based auditory BCI systems

All previous P300 spellers are based on the recording of visual event-related potentials. However, there is no sense in using visual stimuli in cases of severely paralyzed patients with impaired vision or poor control of eye movements (as ALS patients at the end of the disease). Here are cited the main studies and experiments on auditory P300-based BCI system during the years.

Audiostream

In 2004 Hill et al. [61] developed a brain-computer interface that uses auditory stimuli describing a paradigm that allows a user to make a binary decision by focusing attention on one out of two concurrent auditory stimulus sequences of “target” and “non-target” beeps. It was shown that an unrelated user’s EEG data can be classified with an high level of accuracy with no previous training to direct conscious attention to provide a useful basis for a BCI.

4-choice

In 2006 Sellers and Donchin [120] evaluated the effectiveness of a BCI system that operates by detecting a P300 elicited by one of four randomly presented stimuli (i.e., YES, NO, PASS, END). The participant’s task was to attend to one stimulus and disregard the other three. Stimuli were presented auditorily, visually, or in both modes. It was demonstrated that the event-related potentials elicited by the target stimuli could be discriminated from the non-target stimuli.

Auditory speller (Klobassa)

In 2009 Klobassa et al. [79] investigated the efficacy of the use of six environmental sounds to operate a 6x6 P300 Speller. It was demonstrated that an auditory P300 BCI is feasible with a reasonable classification accuracy and an achievable rate of communication with a participant who has limited visual modality.

Auditory speller (Furdea)

In 2009 Furdea et al. [53] proposed an auditory stimulation used in order to make P300 spellers suitable for ALS-LIS patients.

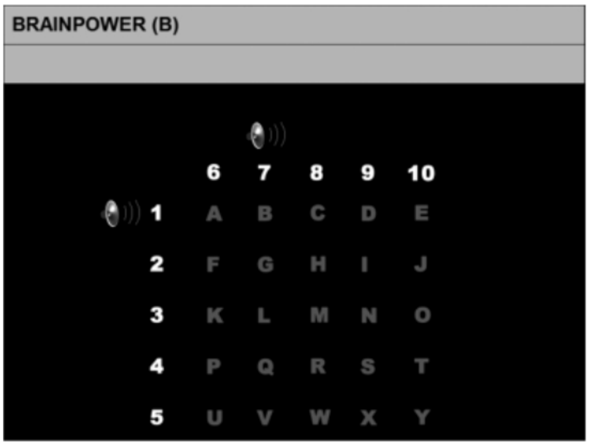


Figure 2.26: The 5x5 visual support matrix for the auditory ERP speller. The numbers surrounding the matrix aimed at facilitating finding the target coordinates. Each number corresponded to an auditory stimulus. For instance, to select the letter "B" the user was required to focus attention on auditory stimulus "one" during the first interval of the trial and on auditory stimulus "seven" during the second interval of the trial.

In his experiment a 5x5 matrix was used in both visual and auditory ERP spellers to reduce the trial duration as a result of the longer stimulus presentation times required for the auditory modality. Because the stimuli are auditory rather than visual, the flashes are replaced with presentation of auditory stimuli that are coded to particular rows and columns of the matrix (Figure 2.26). Each character's position in the matrix was coded by two auditorily presented number words: one corresponding to the row and one corresponding to the column. To select a particular target character, the participant had to attend to the two target stimuli representing the coordinates of the character in the matrix. In addition to auditory presentation of numbers, the matrix, referred to as visual support matrix. Using the oddball paradigm, users were required to focus their attention on the numbers coding the target character by counting how often the numbers were presented.

Auditory oddball BCI for binary choices

In 2010 Halder et al. [117] proposed an auditory BCI based on a three-stimulus paradigm. This paradigm is similar to the standard oddball but includes an additional target (i.e. two target stimuli, one frequent stimulus). This BCI system offers communication with binary choices (yes/no) independent from vision. As it requires only little time per selection, it may constitute a reliable means of communication for patients who lost all motor function and have a short attention span. The intention of this basic BCI approach is to provide a means of communication for those users who cannot achieve a reliable level of control with one of the previously described visual and auditory BCI systems.

In this study they were able to show that a BCI with a three-stimulus oddball paradigm is feasible and it can offer either a high reliability or communication speed achieving an high accuracy and competitive bit-rates. Until today no other study has been carried out a new method to improve the accuracies using auditory stimuli only for ALS patient communication.

Chapter 3

Equipment and Methods

'When the power of love overcomes the love of power, the world will know peace."

Jimi Hendrix

As said in first chapter in Section 1.2 the main purpose of the thesis is to reproduce the study proposed by Halder in 2010 building a portable auditory BCI system in order to test on healthy participant the method functionality in more comfortable way, using our EEG signal processing techniques and offline system classification methods. In Chapter 3 the materials and methods used to build the portable auditory BCI system are described in detail.

As introduced in the first chapter, the main thesis project purpose is to develop a portable auditory brain-computer interface P300-based in order to allow Yes-No communication to Amyotrophic Lateral Sclerosis patients. In order to do this we have relied on the specification provided by the psychologist Mauro Marchetti [129]. The experimental work of the thesis is structured in mainly three parts: the first one regards the design of the oddball paradigm and its implementation in the design of the auditory brain-computer interface application based on that specifications; the second part focuses mainly on the implementation of the Auditory P300 BCI in the "Progetto ON" in parallel with the Visual P300 BCI speller [46], and the improvement of the previous application drivers that allow the communication between devices; the third part of the thesis concerns the clinical in-ear headphones design and the electronic circuit implementation in order to synchronize in time the acoustic stimuli delivered to the participant with the EEG signals during the training session. This step was performed in collaboration with three students in Biomedical Engineering at Politecnico di Milano for their Bachelor thesis followed by us during their work [25]. They have also written the user manual revised in the Appendix A.

At the end the portable application is tested on healthy participants to evaluate its functionality. The EEG data and the audio stimuli are acquired and then sent to a classifier in order to test the audio protocol for classification.

All the work and the experiments were carried out at the Artificial Intelligence & Robotic Laboratory of Politecnico di Milano - Como campus.

The data acquired on tablet were elaborated and classified off-line. This process was performed on a server under the ownership of Politecnico di Milano, giving back the classifier parameters directly on the portable device used for the BCI for the classification.

In this chapter we report the design specification made by the psychologist (Section 3.1) and all the equipment used and developed to build our Auditory BCI system.

3.1 Design Specifications

The main design specification defined by the psychologist Mauro Marchetti for his study to us, is to develop a complete interface in order to deliver a randomized sequence of acoustic stimuli to subjects based on the Three-stimulus oddball paradigm (see Section 2.4.6 on page 39), similar to the one proposed by S.Halder et al., (2010) (Figure 3.1).

S. Halder er al. delivered to healthy subjects a randomized sequence of acoustic stimuli composed by frequent standard tones (pink noise tones) and two targets (pure tones). They performed three tasks where the two target stimuli for the binary selection differed in *pitch*, *loudness* and *location*. The results of the study showed that the Information Transfer Rate (ITR) (defined as the speed of communication expressed in bits/min) is higher for the pitch task.

For this thesis we consider simple pure tones as standard tones (e.g., 1000 Hz with a duration of 80 ms instead of pink noise). In order to make a binary selection we consider two simple words (e.g., YES or NO) as target/deviant stimuli instead of pure tones. The choice of these two simple words is made for their ease of recognition in making a binary selection.

In order to do this, the application developed has to be usable for implementing at least the two deviants oddball paradigm, but with the possibility of implanting the presentation of different sequences of acoustic stimuli.

The application developed should be able to:

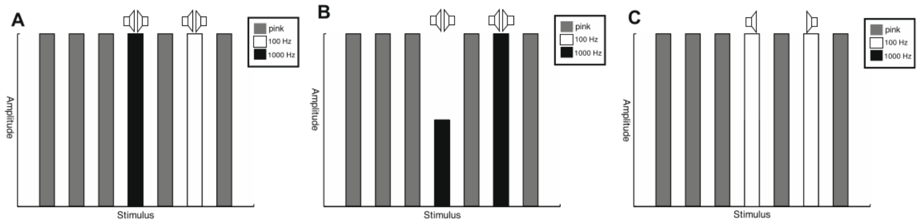


Figure 3.1: Schematic illustration of a single sequence of the three different tasks. In Graph A (pitch task) the stimuli consisted of two targets one at 100 Hz (white bar) and another at 1000 Hz (black bar). The second task (loudness task, Graph B) comprised two targets at 1000 Hz with an amplitude of either 75 dB or 60 dB. Graph C illustrates the third task (direction). The two targets (both at 100 Hz) differed in the direction from which they were presented to the participant (either the right or left channel of stereo headphones). All three tasks comprised five presentations of pink noise per sequence which served as standard tones (grey bars).

1) load the chosen audio files (e.g. wave file) from the SD card.

2) possibility to chose for each stimulus the modality of presentation: STEREO, MONO LEFT, MONO RIGHT.

3) possibility to manipulate the number if trials of acoustic stimuli presentation, and the ratio of presentation as the number of ripetition of each single stimulus:

   - standard frequent sound ratio .7

   - the deviant 1 word *yes* ratio .15

   - the deviant 2 word *no* ratio .15

4) possibility to define the interval time of stimuli presentation. The two possibility of interval choice should be the *Stimulus Onset Asynchrony* (SOA) and the *Interstimulus Interval* (ISI). The ISI or SOA choices should have other advanced options of timings defined between two consecutive stimuli:

   - possibility to define a fixed time interval in milliseconds

   - possibility to define more than one interval (up to five) in milliseconds that are presented randomly during the stimuli presentation

   - possibility to define an interval of values (minimum and maximum) in milliseconds between which randomly selecting values

5) possibility to save a configuration and easily load it in a further session

6) before delivering an acoustic stimulation to the participant, the application should force the experimenter to insert participant ID, to select one of the saved configuration, and to select the session number.

7) at the end of the session, the file should be saved automatically and the EEG file should have:

   ParticipantID_paradigmName_sessionN_date_hour_minute

3.2 System Architecture

This section contains the main elements used to build our portable auditory brain-computer interface system. In order to acquire the EEG signals, the owner hardware of Politecnico di Milano have been used as for the portable Speller P300 ("Progetto ON" [46]).

The Figure 3.2 shows the connection scheme between devices for our auditory BCI system implementation. In order to build our BCI system, the main component of the EEG acquisition hardware is the electroencephalograph. It retrieves brain activity that comes from the electrodes positioned according to the 10-20 system (Figure 2.7 on page 26). This EEG signals are then amplified with a specific EEG amplifier. In our case the device is the BE Light [126].

The EEG amplifier is connected to a network device with a bidirectional optical fiber which guarantees signals isolation and maintains high quality signals transmission. This network device is connected to an Access Point with an Ethernet cable with a static IPv4 (192.168.171.212) in order to have a directly access to the EEG amplifier in the Local Area Network by the application. This connection permits the communication between the EEG amplifier and the Tablet also connected to the access point through DHCP via WiFi or with a static IPv4 (properly set in the range of 192.168.1.1-255 with a Subnet mask 255.255.255.0).

The EEG signal processing of brain activity stimulation by acoustic sequence is directly elaborated in real-time by the application installed on the tablet. The stereo audio signal before being sent to the participant is split with a splitter. This signal consists of the acoustic stimuli sequence that are sent (with a properly designed in-ear headphones) to the participant to avoid the incorrect calibration of the electrodes. At the same time the stereo audio signal is sent to a designed electronic circuit which correctly adapts it to the trigger port of the EEG amplifier converted into a mono signal.

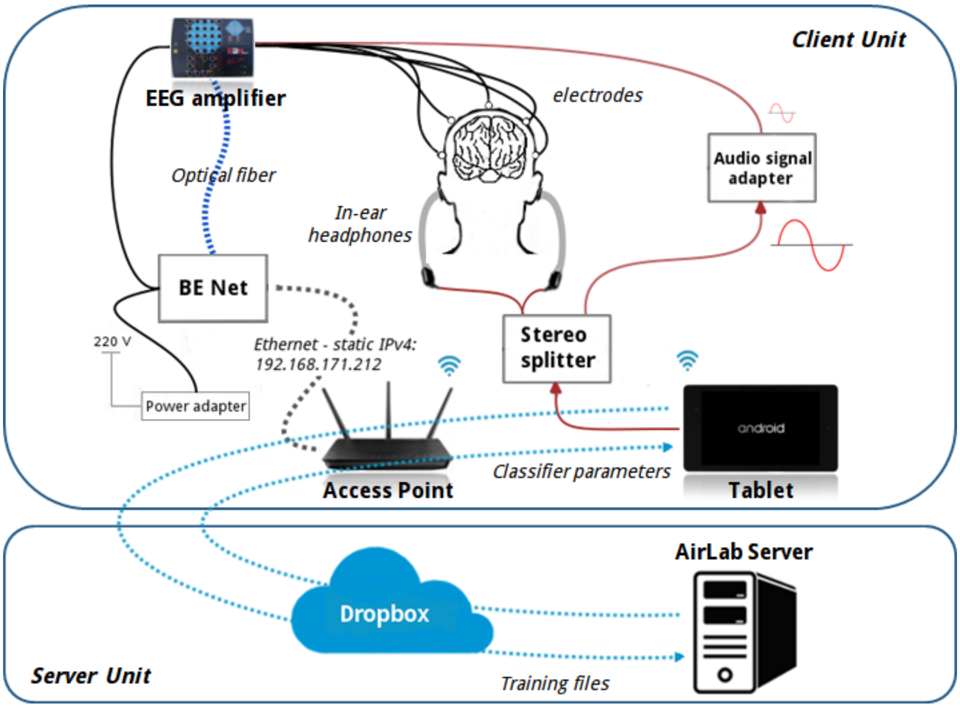


Figure 3.2: *Auditory BCI system. Connection scheme between devices for the training session. On top is represented the client side where the training is performed. On bottom image the server unit for the elaboration of the training files.*

This triggered signal is synchronous with the acquired EEG data stored in a proper file directly on the tablet while the stimuli are delivered to the participant.

At the end of the acoustic training session, the files saved on the Tablet are uploaded to a synchronized Dropbox shared folder with a server. The training recordings are elaborated and then the classifier parameters to that folder are uploaded.

3.3 BE Light EEG amplifier

The BE Light is a compact system for acquisition EEG signals and this unit is represented in Figure 3.3.

The main purpose of the device is to acquire bioelectric signals from electrodes with the stimulation of brain activity. The amplification system captures the biological signals with appropriate electrodes or transducers, amplifies them in voltage (μV order), provides anti-aliasing filtering in

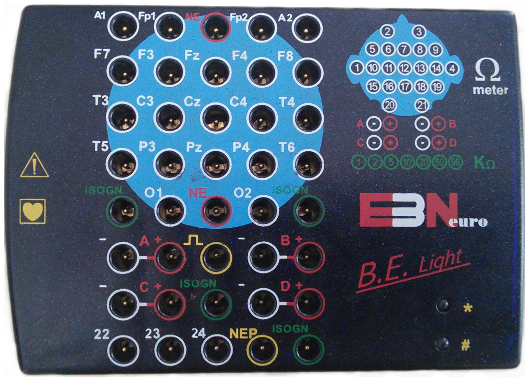


Figure 3.3: BE Light by EBNeuro [126].

order to optimize the digital conversion, converts them into digital form (analog/digital conversion) and then sends the numeric data with the optical fiber to the host system. The device host system (the tablet in our auditory BCI system) reads the data and processes them according to its own logic. With a specific application software is possible to program the amplifier functionality, determining which channels to acquire, the sampling rate, etc.

The BE Light uses the optical fiber to isolate the the amplifier and the patient. The amplifier is not designed to cut the 50 Hz line noise frequency so all the power supplies and all other components that process or deliver AC current must be placed at least 50 cm - 1 m from the subject during the experiments in order to have negligible effects on signals quality. The device applies to all unipolar, bipolar and polygraphic channels a 1st order 0.1 Hz High-Pass filter. The frequencies over 1 KHz are cut by a Low-Pass anti aliasing filter applied to the EEG and polygraphic channels, while anti alias Low-Pass filters are applied to the bipolar channels for frequencies over 2 KHz.

The AC/DC conversion uses 16 bits SAR A/D and the sampling frequency can be chosen between 128, 256, 512, 1024, 2048, 4096 and 8192 Hz. In our experiments we use a 512 Hz sampling frequency.

The BE Light system provides the possibility to be equipped with the option “Ohmmeter”. This option allows us to monitor the resistance value contact of the electrodes through a special matrix of bright LEDs. As we can see in Figure 3.3 there are 40 patient input channels to the amplification system. There are 21 EEG unipolar inputs with common reference mainly used for the 10-20 EEG system (see Figure 2.7 on page 26) with AC coupling. This inputs are numbered from ”1” to ”21” and accept only the signals that come directly from the survey points.

The channels “22”,“23”,“24” are 3 polygraphic unipolar inputs with their own reference with AC or DC coupling (set by software). The amplified signal is the difference between each of these electrodes and the signals polygraphic reference electrode connected to the NEP input. The ‘two ‘NE” (input Neutral Electrode) are channels for the common reference of the 21 unipolar inputs (AC coupling). They are connected together. The 8 inputs for 4 bipolar inputs with AC coupling, are the “A”, “B”, “C”, “D” and allow to amplify the potential difference between the electrode connected to the “-” input with the one connected to the “+” input. The 4 inputs for patient isolated ground marked as “ISOGN” are internally connected to each other and represent the contact ground points (patient ground). The input for the separated reference of the 3 unipolar inputs (AC or DC coupling) marked as “NEP” (Neutral Electrode Polygraphic), allow the connection with the electrode that acts as a reference for the 3 polygraphic unipolar inputs. The calibration input signal is represented with the channel marked with a dedicated symbol. This allows to retrieve the calibration signal generated from the internal acquisition box. It can be used to control the goodness of the internal amplification.

3.4 EEG electrodes

In order to acquire the EEG signals our EEG acquisitions have been performed using electrodes connected directly on the BE Light. In Figure 3.4 on the right image the Ag-AgCl electrodes disks plugged directly into the EEG amplifier are represented.

If all the 19 channels are interested for the auditory BCI experiments the 21 electroencephalographic cap can be used. This cap is composed by 19 electrodes placed on participant according to the 10-20 system (see Figure 2.7 on page 26) while other 2 electrodes are used for the ISOGN and NE ports on the BE Light.

3.5 BE Net

The BE Net [126] (Figure 3.5) is a device developed by the EBNeuro as the BE Light. It is directly connected to the power supply and it provides power also to the BE Light. It establishes the connection in the LAN (Local Area Network) with a static IPv4 (192.168.171.212) to the Access Point with a standard Ethernet cable. The digital data converted by the BE Light are

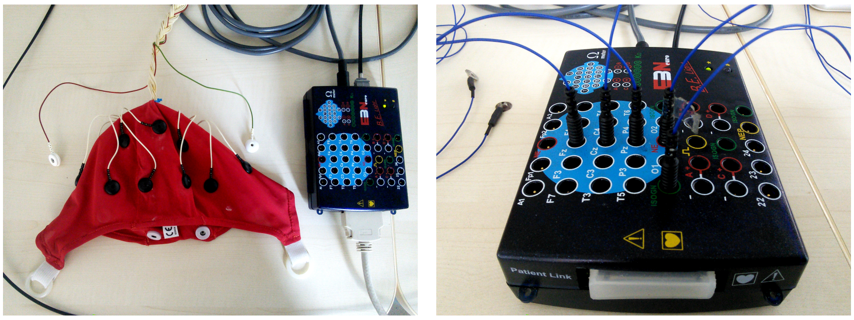


Figure 3.4: Electrodes used for the experiments. On left image is represented the electroencephalographic cap with 21 electrodes (19 for the 10-20 system and 2 electrodes for the ISOGN and NE ports on the BE Light). On right the Ag-AgCl electrodes disks plugged directly into the EEG amplifier are represented.

Figure 3.5: BE Net by EBNeuro [126].

sent to the BE Net through an optical fiber that isolates the EEG signals and makes them available on the local net in order to be accessed from the host device.

3.6 Access Point

To build our acoustic BCI system to have a generic access point with an Ethernet port and a Wi-Fi connection system is necessary. The access point is used to build a LAN (Local Area Network) in order to allow the communication between the EEG amplifier and the host device (Tablet). The IPv4 assignment in the local network can be configured with a DHCP (Dynamic Host Configuration Protocol) for the Tablet via Wi-Fi connection. It is also possible assign a static IPv4 configuring it with a subnet mask 255.255.255.0 leaving the static IPv4 132.168.171.212 for the EEG amplifier connected physically to the access point through the BE Net with a straight

Ethernet cable.

The access point should be also connected to the Internet with a router with the proper ISP (Internet Service Provider). This is done to allow the communication between the application installed on Tablet and the server unit to share the training files and the classifier parameters on Dropbox folder.

3.7 Tablet

As we can see in Figure 3.2, the host device that communicates with the EEG amplifier is a tablet. It is connected to the Access Point with the Wi-Fi technology based on the IEEE Standard 802.11 (see Section 3.6 for details). The application developed installed on it allows to control all the BCI system functionality. Once established the connection with the BE Light, it is possible to check the correct impedance with "Ohmmeter" function and visualize the EEG signals processed on the application with a simple auditory stimulation. With the tablet is also possible to set all the participant parameters, change the desired preferences and set all the proper configurations before the acoustic training session can start.

The training session generates a stereo audio signal composed by a sequence of audio stimuli. This signal before reaching the participant ears with a customized headphones (see Section 3.8) is split with a simple stereo splitter.

With the device connected to the Internet is possible to send the training files generated automatically to the server unit that elaborates them and returns the classifier parameters to the system.

The application for the auditory BCI is developed for Android devices. In Figure 3.6 the two host devices tested during all the development phase are shown. Two tablets with different screen size in order to design an elastic application interface adapted for both screen devices are chosen.

The first one is the Asus Nexus 7 2013 Wi-Fi version. It has a 7" LED Backlight WUXGA (1920x1200) Screen IPS Panel equipped with a CPU Qualcomm® Snapdragon™ S4 Pro 8064 32-bit Quad-Core 1.5 GHz, GPU Adreno 320 and 2 GB RAM. On it the last version of Android™ Lollipop 5.1.1 is installed.

The second host device tested is the ASUS TransformerPad. It has a 10.1" LED Backlight WXGA (1280x800) Screen IPS Panel equipped with a CPU Intel® Atom™ Z3745 32-bit Quad-Core 1.6 GHz with Intel Hyper-Threading Technology, GPU Intel Gen 7 (Ivy Bridge), 1 GB RAM. On it the Android™ KitKat 4.4.2 is installed.

Figure 3.6: Host devices used in the auditory BCI system. On the left: Asus Nexus 7 2013 with Android^{TM} Lollipop 5.1.1 (7" LED Backlight WUXGA 1920x1200 Screen IPS Panel). On the right: ASUS Transformer Pad with Android^{TM} KitKat 4.4.2 (10.1" LED Backlight WXGA 1280x800 Screen IPS panel).

3.8 In-ear headphones

As said at the beginning of this chapter, the in-ear headphones were designed by three students under our supervision. The BE Light has inside for each acquisition channel a sophisticated EEG signal amplifier that are not designed to cut the 50 Hz noise frequencies. It is possible that during the training session the audio signal synthesized directly in the subject ear may interfere with the electrodes placed on the patient scalp during the EEG signals acquisition. This is due the magnetic field generated by the headphones. A specific in-ear headphones called "clinic headphones" that maintain a suitable distance from the electrodes placed on the subject to avoid this phenomena are designed from students.

With this type of headphones, internal components formed by cone, coil and magnet are separated with a tube in order that only the silicone ear plugs are near the electrodes. The main drawback is the reduction of the sound intensity then the original SPL measured as the headphones sensitivity is attenuated. This is caused by the dispersion/absorption of the sound propagation inside the medium before reaching the ears (see Section 2.2.1 on page 13 for details).

To design the in-ear headphones their physical characteristics were hypothesized as close as possible for the correct propagation of the acoustic signal in the ears during the training session. Measures were made to obtain this results and materials that do not attenuate to much the acoustic signal for the pitch perception are used. The Loudness level (see Section

Table 3.1: In-ear MDR-EX15LP made by SONY® technical specification.

Type	Closed, dynamic
Driver	Unit 9mm
Sensitivity	100 dB/mW
Power Handling Capacity	100mW
Impedance	16 Ohm (1KHz)
Frequency Response	8-22,000 Hz
Magnet	Neodymium
Cord	Y-shape
Plug	Gold-plated L-shaped Stereo Mini
Weight (Without Cord)	Approx. 3g

2.2.3 in Chapter 2) defined by the application on Tablet for each channel (see Section 4.6 on page 88 for details) was measured and tested.

The in-ear MDR-EX15LP made by SONY® was used and adapted to this purpose. The technical specification are defined in Table 3.1. In order to realize the customized headphones a special oxygenation mask tube (OS3xx model) is adapted between the in-ear amplifiers and the silicone ear plugs. This tube is made of PVC (PolyVinyl Chloride) and its length (one meter) was chosen as a compromise between the sound attenuation inside the medium in order to avoid the magnetic field interference generated with the electrodes. In Figure 3.7 on the right image the in-ear headphones developer are represented.

Students using an environmental sound level meter have obtained some interesting results. They have measured the Sound Pressure Level (SPL) (see Section 2.2.1 for details) obtaining the sound attenuation inside of the tube. In Table 3.1 the in-ear Sensitivity is specified as $100dB/mW$ declared by the manufacturer. This values can be associated to the Loudness perceived by the subject at maximum volume level of the audio synthesis.

To measure the SPL the sound level meter was placed inside of a glass jar padded completely with a sponge and cotton as in Figure 3.7 (left image). This to avoid the possible sound dispersions and obtain the SPL values as real as possible. The first experiment made was to measure the effective headphones Sensitivity without the tube and they have realized an average reading of 97 dB/mW, very close to the one defined in the headphones specification. This experiment was replicated with a one meter tube obtaining on average 72 dB/mW. So it can be possible to define that the sound attenuation measured inside the tube is about 25 dB/mW. In Section 4.6 on

Figure 3.7: On the left image there is the sound level meter placed inside of a glass jar padded completely with a sponge and cotton used to measure the in-ear headphones SPL. On the right image there are the in-ear headphones designed with the PVC (PolyVinyl Chloride) tube.

page 88 is explained how the main setting interface for delivering stimuli is developed. With the project specification defined (see Section 3.1 on page 64) the values (75 dB, 65 dB and 55 dB) of the volume level sensitivity are defined for each channel using this type of measures, mapping the effective float level in Java code (from $0.0f$ to $1.0f$) with the corresponding values measured with the sound level meter.

3.9 Audio Signal Adapter for EEG synchronization

During the training session, the acoustic sequence sent to the participant stimulates the EEG signals. These signals are acquired by the EEG amplifier as described in Section 3.3. The application during this phase save each EEG signals from each electrode directly on device in a specific file (see Section 4.12 on page 113).

In order to make a correct classification when doing experiments with ERPs induced by external stimuli (see Section 2.4.6 on page 39), it is very important to have the EEG recording synchronized with the acoustic signal. Relying on the input dynamics specification of the EEG amplifier, we designed a simple electronic scheme and realized the electronic circuit for this purpose. This circuit allows to adapt the Tablet audio signal output into the BE Light input trigger port.

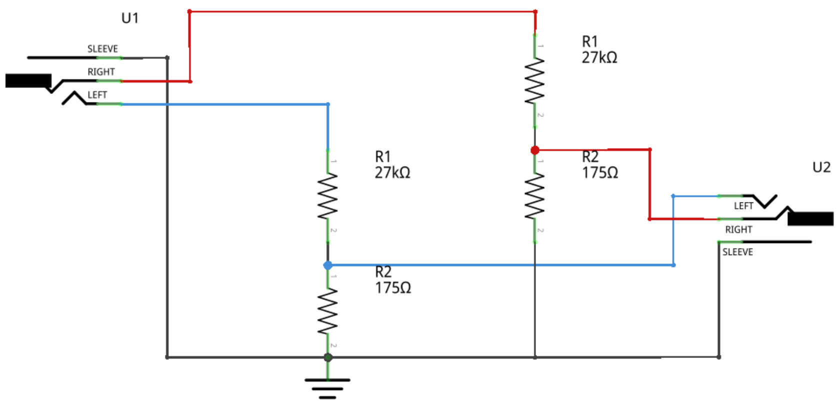


Figure 3.8: Audio signal adapter electric scheme for the EEG synchronization. U1 is the 3.5 mm input jack where the tablet audio signal enters in the circuit. U2 is the output 3.5 mm jack where the audio signal goes to the EEG amplifier.

As explained in Section 3.3 on page 67 the BE Light has three polygraphic unipolar inputs (22, 23, and 24 ports) with a common separated reference (NEP port). Via software we have enabled the 22 port as the audio input considering the NEP port as the ground of the signal in the EEG amplifier. In Table 3.2 Gain/Resolution, Dynamics and Noise specification for the BE Light are defined.

In Figure 3.8 the electric scheme designed is represented. The input to the circuit is the Tablet stereo output audio signal (considered with a maximal dynamics as $1V$) connected with a 3.5 mm audio jack. To obtain the right maximal dynamics defined in Table 3.2 as the output circuit, a simple *voltage divider* is designed for each channel (mono left and mono right) as in the Equation 3.1.

$$U_{2(left)} = U_{1(left)} \frac{R_2}{R_1 + R_2}; U_{2(right)} = U_{1(right)} \frac{R_2}{R_1 + R_2} \tag{3.1}$$

Knowing the input and the output of the circuit it has been possible to dimension the resistors obtaining $R_1 = 27K\Omega$ and $R_2 = 175\Omega$. The two stereo output have to be converted into a mono signal in order to be interfaced to the BE Light 22 and NEP port (yellow connector for the NEP port and the black one for the 22 port), so they are maintained separated and then connected together using the jumpers given by the EBNeuro. The signal conversion is done to trigger each acoustic stimulus delivered in different audio channels (right or left) during the training session in the EEG amplifier, that has one input channel (see the right image in Figure 3.9).

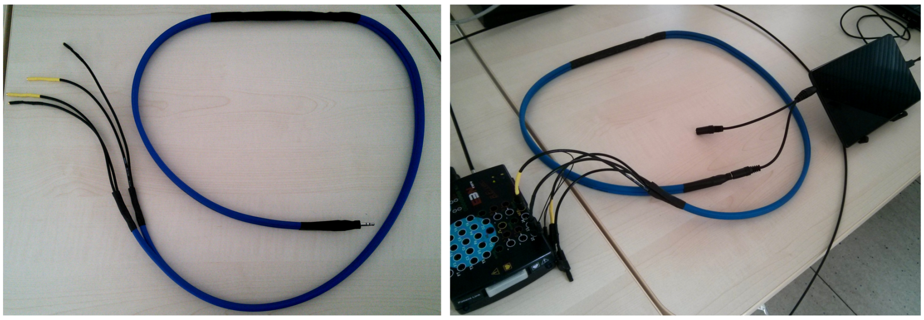


Figure 3.9: On the left image there is the audio signal adapter cable developed for the EEG synchronization. The input is the 3.5 mm stereo jack, while at the output there are the two stereo channels separated with their respective inputs for the NEP and the 22 ports. On the right image an example of the cable plugged in the EEG amplifier is represented: the NEP and the 22 inputs of each channel are connected together with the jumpers in order to convert the stereo signal into a mono signal.

Table 3.2: Specifications of Gain/Resolution, Dynamics and Noise of the BE Light.

Gain/Resolution	Dynamics	Noise (0.1Hz - 70Hz)
High: $\frac{1}{8}\mu V/bit$	$8000\mu V$	$\leq 0.3\mu Vrms$
Low: $2\mu V/bit$	$128mV$	$\leq 5\mu Vrm$

After the audio circuit realization, before sending an audio signal to the EEG amplifier, the output voltage was measured and tested with an oscilloscope in order to verify the correct design. This because the standard tone at 1000 Hz that is sent to the in-ear headphones is not in the dynamics of the BE Light. Therefore it was decided to generate with the classic 1000 Hz (as standard tone) a pure tone not audible by human hearing at 20 Hz, but acquired in the EEG amplifier.

The output dynamics has been measured with a sinusoidal continuous signal at 20 Hz. At the tablet output with a maximum volume (circuit input at U1 in Figure 3.8) we measured with an oscilloscope a 720 mV peak to peak value. At the EEG amplifier input (circuit output U2 in Figure 3.8) the peak to peak value was 10 mV correctly inside the BE Light dynamics. Words are perfectly inside this dynamics because the frequency of speech is about 200 Hz for the yes-no words.

3.10 Server unit

The training files stored on the device are sent to a Dropbox shared folder with an account of Politecnico di Milano. This operation is obtained transferring the registered session (see Section 4.12 on page 113) coupled with the respective stimulation sequences from the Android application installed on Tablet. For this purpose we chosed to use the Android API provided by the Dropbox developers.

A Unix server (with a pubblic IP located in AirLab at Politecnico di Milano - Como Campus) is used to obtain the classifiers parameters (called *weights*) with special genetic algorithm for feature extraction and classification (see Chapter 5 on page 117 for details). The training files processing is performed on Octave code [130]. Once the parameters are obtained, they are uploaded to the shared Dropbox folder and downloaded on tablet.

It is possible to use any Unix machine to build the server satisfying the following requirements:

- Full internet connection (24 hours on 24 hours), needed for the Dropbox functionality.
- 2 GHz Dual Core Processor or more (3 GHz Dual Core advised)
- 2 GB Ram (4 GB advised).
- Unix OS (Debian advised, Tested on Xubuntu 11.10/Ubuntu 12.04 and Linux Mint 13 Maya).
- *Cron* daemon or similar.
- *Dropbox* installed as *root*.
- *Octave* ($\geq$ 3.6 with Java plugin)

To run the scripts for data processing and obtain the *weights* for the application it is sufficient that the system uses a scheduler (*cron*) to repeatedly run a script to check if another instance is already running and eventually start the process.

We have configured our server in order to check if there are any training files every 30 minutes (files not trained yet). The server has this specifications: Intel® Dual Core Pentium® 4 CPU @ 3.00 GHz, 1024MB RAM, Linux Mint 13 Maya OS installed.

Chapter 4

Application and modules

"I used to live in a room full of mirrors; all I could see was me. I take my spirit and I crash my mirrors, now the whole world is here for me to see."

Jimi Hendrix

The most of the thesis work was to develop an acoustic training interface for delivering acoustic stimuli to subjects while their EEG is recorded with a P300 stimulation. A special Android application it has been developed expanding the "Progetto ON" [46] created by Info-Solution S.P.A. [127].

This section contains the main software realization and all the techniques performed to build the application and the driver optimization. In particular this section explains the development environment used to build the application and the libraries implemented for the system's functionality. It explains how the Java classes and drivers are adapted in order to allow the communication between Tablet and the EEG amplifier also for the Auditory P300 BCI and the main acoustic yes-no BCI menu functionality and all relatives implementations. In the last part of this section (Section 4.12) explains how the training files are automatically generated during the EEG signals acquisition and saved on the host device. The methods chosen to compile them and send it to the server classifier are treated.

4.1 Development Environment

The application has been developed in Android OS with Android Studio [125]. This development environment is built on IntelliJ IDEA Community edition, the popular Java IDE by JetBrains [128].

All the code developed, the test phases and the debug of the application during the acoustic training session were carried out on a DELL Precision

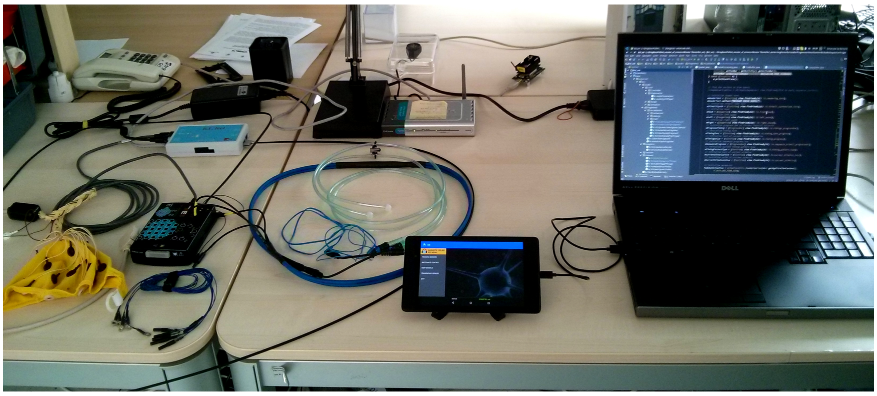


Figure 4.1: Development environment in the AirLab in Politecnico di Milano - Como campus.

M6500 with Intel® Core™ i7 CPU Q 820 @ 1.73 GHz, 8 GB DDR3 @ 1333 MHz and a GPU NVIDIA® Quadro® FX 2800M. Ubuntu Studio 15.04 is the OS installed [131].

In Figure 4.1 the environment used to develop and test the application is represented.

4.2 Java classes and drivers adaptation

This section explains how drivers used for the Visual P300 BCI are adapted to establish the connection between the tablet and the EEG amplifier in order to execute commands to the BE Light by the application also for the Acoustic P300 BCI.

The first important thing made was to implement the driver classes in the main Android Activity [28] class called ***MainBciAudioActivity***. The interfaces methods to the BE Net device (see Section 3.5 on page 69) are initialized with a class called ***Communication*** that extends a thread. This class manages connection establishment, status and refresh. Also triggers the acquisition modes and sets the EEG hardware configuration based on the configuration status (see Section 4.4 on page 83). Here all the acquisition channels are added (see Section 4.8 on page 99) with their hardware codes and dynamic ranges defined by EBNeuro[126].

The ***BENet*** driver class is initialized on the static IPv4 192.168.171.212, with a specified boot port (7023), control port (7024) and data port (7025) also passing both pointers of the Auditory P300 BCI and the Visual P300 BCI Activities.

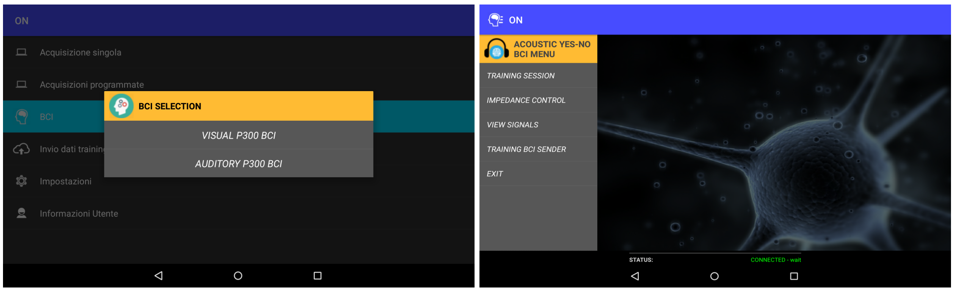


Figure 4.2: Rendering of "Progetto ON" main menu. On the left: representation of the main menu in the application with the possibility to choose the visual P300 BCI or the Auditory P300 BCI. On the right: Acoustic yes-no BCI menu.

When all the parameters are verified the socket creation is initialized. The first firmware file uploaded is the one that manages the connection (conv_ctrl.jet). The second one is the mDSP firmware that manages the connection and the communication with the BE Light (EBX_MALL.jet). The last one is another sDSP firmware file uploaded on the BE Light device (EBX_SSA3.jet). At the end the DSP on BE Net to communicate with BE Light is configured getting the device hardware informations and capabilities.

Another class implemented and adapted in the Auditory P300 BCI Activity is the circular buffer of samples class driver called ***SampleBuffer*** that includes methods for accessing EEG data, filtering and data saving. This class is initialized with 512 samples in the buffer.

The buffer class driver containing the signal epochs and the related stimulation code is the ***TrialBuffer*** class. This class is initialized with a maximum of 120 epochs in the trial.

4.3 Acoustic yes-no BCI menu

As said at the beginning of this chapter the acoustic yes-no BCI application was developed expanding the "Progetto ON" [46] based on a Visual Speller P300.

As shown in Figure 4.2 the main menu of the application it has been redesigned with the possibility to choose the old Visual Speller P300 and the Auditory P300 with a simple customized dialog.

Selecting the Auditory P300 BCI in the pop-up (the left image in Figure 4.2), a new Android Activity is created and then the Acoustic yes-no BCI menu is initialized (the right image in Figure 4.2). In this Activity all

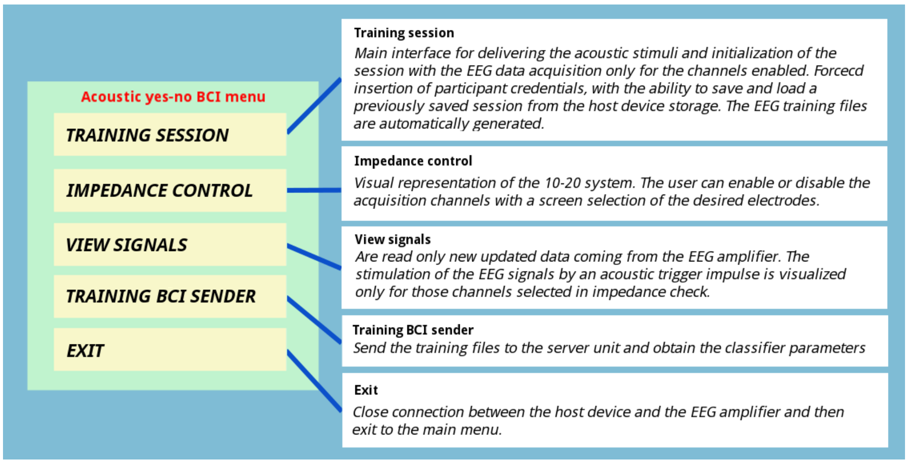


Figure 4.3: Acoustic yes-no BCI menu scheme. Brief explanation for each item selected in the menu.

the drivers that permits the communication between the Tablet and the EEG amplifier are initialized too (see Section 4.2 on page 80 for details). With a progressbar animation and screen messages the user can monitor if the connection has been successful. As we can see in Figure 4.2 when the connection is successfully established the status is set to "CONNECTED - wait" in green color. (see Section 4.4 for a more detailed explanation for the update connection status).

The Acoustic yes-no BCI menu basic scheme is represented in Figure 4.3. After the connection has been carried out between devices the user can navigate through the menu.

4.3.1 Training session

The main interface for delivering the acoustic stimuli to the participant while the EEG is recorded is the "TRAINING SESSION", selectable in the menu list. In this special interface it is possible to create a new session entering patient credentials, load the acoustic stimuli from the tablet storage, select the proper audio synthesis settings and set all the parameters according to the three-stimulus oddball paradigm (see Section 2.4.6 at page 39). Here it is possible to save the current patient configuration on the storage and load it afterwards. During the auditory session the EEG training files are automatically generated and saved in a specific folder.

4.3.2 Impedance control

The "IMPEDANCE CONTROL" selection starts the visual representation of the 10-20 system where the user can enable or disable the acquisition channels with a screen selection of the desired electrodes. In this screen the impedance for each electrode selected in real time on a rendered human head, showing texts of the real impedance values in Ohm are also elaborated and visualized. These values are refreshed every 2 seconds with a proper thread that controls the request of values from the EEG amplifier and the screen elaboration directly on tablet (see Section 4.8 on page 99 for details). Before any data processing the electrodes must be defined.

4.3.3 View signals

Directly from the current screen visualization the user can select "VIEW SIGNALS" button in the lateral menu list. Here are elaborated all the parameters according to the user electrodes selection. Only the new updated data coming from the EEG amplifier are read and the stimulation of the EEG signals by an acoustic trigger impulse is visualized only for the channels selected. The dynamic data acquired and the signal processing visualization in real time are elaborated with a proper algorithm.

4.3.4 Training BCI sender

"TRAINING BCI SENDER" starts a new Android Activity pausing the Acoustic yes-no BCI menu and the devices connection. The interface was designed to contain both the Visual P300 and the Auditory P300 training files. The user can navigate through the menu and select the desired training files for a specific participant subdivided in session order, move them in the archive, delete or send them to the server unit in order to obtain the classifier parameters.

The "EXIT" button closes the connection and the communication between the host device and the BE Net and then go back to the main application screen.

4.4 Update connection status

The update connection status helps the user to understand the behaviour of the EEG amplifier during the work of the application. The states are defined according to the following parameters explained in Table 4.1.

Table 4.1: Status connection of the BCI system.

Status	**Label**	**Color**	**Description**
CONNECTED	*wait*	*green*	*The connection between the host device and the BE Light is successfully established.*
CONNECTED	*readImp*	*green*	*Switched BE Light to impedance analysis mode.*
CONNECTED	*readData*	*green*	*The BE Light is ready to send and receiving data.*
CONNECTED	*readDummy*	*green*	*Instead of reading data directly from BE Light, read data from a file stored on device.*
-	*pause*	*yellow*	*The connection has been established to BE Net device and it is in standby due to transition or application being moved to background.*
UNCONNECTED	*eegmissing*	*red*	*The connection has been established to BeNet device, but it can't send and receive commands from BE Light, thus the device is considered unconnected.*
UNCONNECTED	-	*red*	*We have no connection to BE Net. The connection is restored as soon as possible.*

Three types of status color have been chosen to inform the user while using the application. The red indicates the unsuccessful connection between the tablet and the BE Light. In this state the application tries to restore the connection as soon as possible flashing the firmware to the EEG amplifier.

A special case is considered when the connection has been established to the BE Net device, but it can not send and receive commands from the BE Light, thus the device is considered unconnected. This state is labelled as "eegmissing".

The green color defines the successful connection between the tablet and the BE Light. This status is obtained once a firmware protocol is updated to the BE Net and the static IPv4 192.168.171.212 is set. In this case the status is labeled as "wait" and the BE Light is ready to perform any request made from the host device.

There are two sates of the BE Light considered as special cases when the connection is already established. The first one is when the EEG amplifier works as "Ohmmeter" (see the Section 3.3 on page 67) and this status is labeled as "readImp". The second case is when the BE Light is ready to send and receiving data. In this state it is required the EEG signals reading and processing during the acoustic training session stimulation (this phase is explained in detail in Section 4.12). This status is labeled as "readData".

There is a standby status considered to a default state when the connection has been established to the BE Net device. This status is labeled as "pause" and it has a yellow color which represents some transition or the application Activity being moved to a background state after a while.

4.5 Fragment layout switching algorithm

The layout of the Auditory P300 BCI interface is initialized with the creation of a new Android Activity [28] class called ***MainBciAudioActivity***. As mentioned in Section 4.3, at the beginning of the Activity all drivers that establish the communication between tablet and the EEG amplifier are initialized. The drivers developed for the Visual P300 BCI are adapted also to communicate with the Auditory BCI in order to maintain the two interfaces completely independent. Once the connection is established it is possible to select the desired item in the menu (see Figure 4.3 on page 82).

The Activity implements on the left side a navigation drawer menu [32] viewable with a right swipe. It is automatically hidden when an element is selected. The navigation drawer permits the user to navigate easily in the application without having to return to the main menu when he want to change the view. The main Auditory P300 BCI Activity layout scheme is represented in Figure 4.4.

The main layout of this Activity acts as a container and it is composed of Fragments [29] placed dynamically in it. With a selection of the desired item in the Acoustic yes-no BCI menu, each Fragment is inflated with its own layout in the container with an identifier.

Each Fragment has a life cycle and all elements, objects and functions created with it are destroyed with the Fragment itself when switching from one layout to another. In order to inflate each Fragment and visualize its layout in the Auditory P300 BCI Activity an appropriate algorithm was designed which allows to interact with each Fragment and maintains unchanged all the parameters that the user set during the application usage, despite the exchange between Fragments. This algorithm is implemented

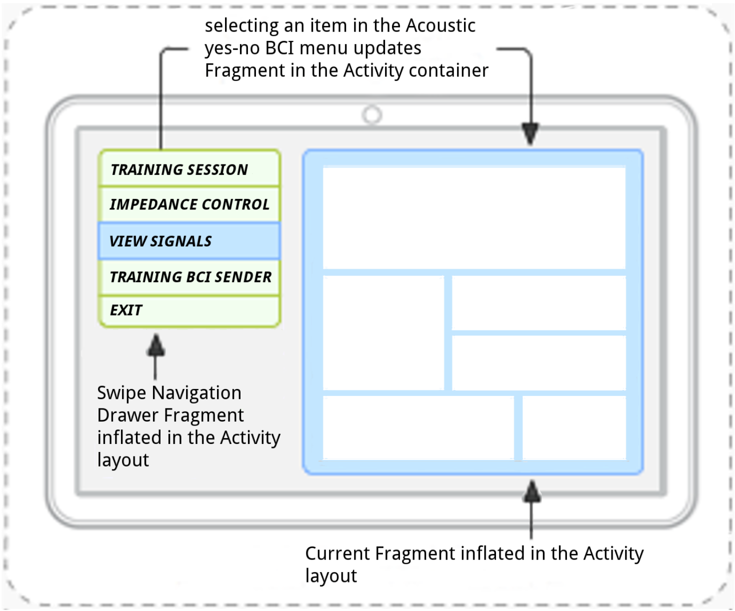


Figure 4.4: Main Auditory P300 BCI Activity layout composition. The layout is composed by a swipe navigation drawer fragment statically inflated in the main layout. This is called with a right drop from the left side of the screen and the dynamic Fragment update happens selecting an item in the menu. On item selection the menu automatically disappears on the screen left side and the fragment is inflated in the layout.

in the default Fragment called ***MainBciAudioFragment*** that is inflated inside the main Activity.

Suppose that the user has set all the parameters in the Training Fragment and then he wants to view the impedance or the signals behaviour once again before the acoustic session. At the Impedance check selection in the menu, the current Training Fragment is replaced by the Impedance Fragment destroying all settings set before. The idea of the algorithm is to keep always alive in some way the training Fragment once it is inflated. This because it is the main layout in which the user interacts more and set all parameters for the acoustic stimulation. In this way is possible to jump one layout to another without losing any previous configuration.

The algorithm works only on “TRAINING SESSION”, “IMPEDANCE CONTROL” and “VIEW SIGNALS” selection in the Acoustic yes-no BCI menu. Android OS allows to handle Fragments with a stack, so is possible to push or pop them from that stack. Another important feature is to hide a a Fragment while another is running over it, leaving the previous hidden on the stack. This allows to process only the current layout without processing the hidden layout by the GPU, while all the parameters initialized of the hidden Fragment are maintained. At Fragment creation it is possible to set a String label that identifies them as a tag, so it is possible to recall that label from the stack.

The model algorithm is defined in Listing 4.1. The Fragments switching is performed only when the connection is established (see Section 4.4 for details) in order to maintain the BELight always ready to satisfy the demands made by the application and preventing the communication errors.

Listing 4.1: Fragment layout switching model algorithm between Training Session, Impedance Control and View Signals in the Acoustic yes-no BCI menu.

```
switch(fragment_tag) {
    case "training": {
        if (no fragment in the stack) {
            push training on top;
        } else if (one fragment in the stack) {
            if (impedance in the stack) {
                pop impedance from the stack;
                push training on top;
            } else if (view signals in the stack) {
                pop view signals from the stack;
                push training on top;
            } else if (training in the stack) {
                stay on training;
            }
        } else if (two fragments in the stack) {
            if (training hidden && impedance on top) {
                pop impedance from the stack;
            } else if (training hidden && view signals on top) {
                pop view signals from the stack;
            }
        }
    } break;
    case "impedance": {
        if (no fragment in the stack) {
            push impedance on top;
        } else if (one fragment in the stack) {
            if (impedance in the stack) {
                stay on impedance;
            } else if (view signals in the stack) {
                pop view signals from the stack;
                push impedance on top;
            } else if (training in the stack) {
                hide training from view;
                push impedance on top;
            }
```

```
        } else if (two fragments in the stack) {
            if (training hidden && impedance on top) {
                stay on impedance;
            } else if (training hidden && view signals on top) {
                pop view signals from the stack;
                hide training view;
                push impedance on top;
            }
        }
    } break;
    case "view_signals": {
        if (no fragment in the stack) {
            push view signals on top;
        } else if (one fragment in the stack) {
            if (impedance in the stack) {
                pop impedance from the stack;
                push view signals on top;
            } else if (view signals in the stack) {
                stay on view signals;
            } else if (training in the stack) {
                hide training from view;
                push view signals on top;
            }
        } else if (two fragments in the stack) {
            if (training hidden && impedance on top) {
                pop impedance from the stack;
                hide training view;
                push view signals on top;
            } else if (training hidden && view signals on top) {
                stay on view signals;
            }
        }
    } break;
```

4.6 Auditory P300 Training interface

The Training session is the main interface to initialize a new acoustic training session. The interface is designed to have a high usability in order to guide the user to set all the parameters in a proper manner for a right acoustic stimuli sequence generation based on the Tree-stimulus oddball paradigm (see Section 2.4.6 on page 39 for details) and on the specifications defined by psychologist Mauro Marchetti (see Section 3.1 on page 64).

The Training Fragment is initialized by the ***MainBciAudioFragment*** class inflating a new Fragment class called ***BciAudioTrainingFragment***. Its layout is composed by many other Fragments inflated dynamically at each

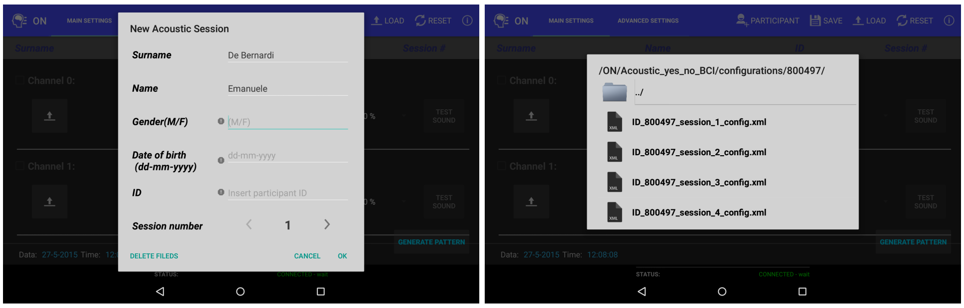


Figure 4.5: Renderings of the acoustic training session in order to initialize a proper clean session. In the left figure there is a popup window appeared on "add participant" selection. In this window the user inserts the credentials and the session is initialized. The right figure shows an example where the user can load a previous session saved on SD card for the initialization of the session.

creation directly in Java code. This to obtain high performances during the usage and to avoid problems of the user interface slowdown during the data processing.

Initially the interface is locked and the "SAVE" Image Button in the menu is disabled. The user before starting a new clean session, must insert the participant credentials or confirm a previous session saved on SD card in a specific folder.

In Figure 4.5 there is a simple rendering for the session initialization. In the left figure there is a popup window appeared on "ADD PARTICIPANT" selection. This is a customized Dialog created with the initialization of a Fragment class called ***NewSessionDialogFragment***.

In this window the user inserts the credentials and the session is initialized on "OK" button selection. The right figure shows an example where the user can load a previous session saved on SD card for the initialization of the session (see Section 4.7 on page 94 for details). The user can select the desired ID folder where inside there are the .xml files associated to that ID divided in order of session and select one of them. In this window it is possible to delete a folder or an xml file listed just holding down them and confirm the operation. At session creation the "SAVE" Image Button is enabled (see Section 4.7 on page 94 to see in detail how the saving and loading algorithms work).

The Fragment class implements a *ViewPager* [36] and a *FragmentStatePagerAdapter* [30] in an another class initialized at its creation called ***BciAudioMain*** where two Tabs in the layout called with the ActionBar method [27] are also implemented. Each Tab has associated a specific Fragment

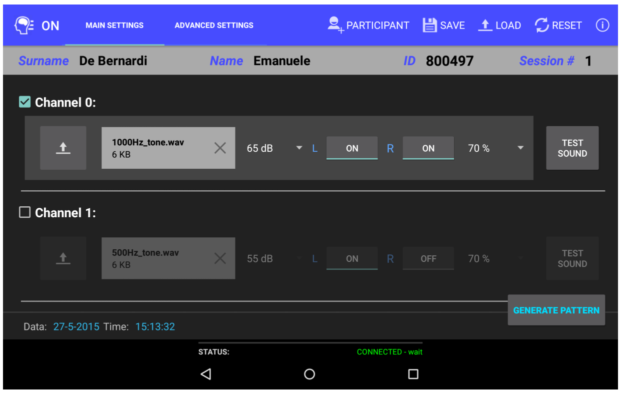


Figure 4.6: Rendering of the Training Session. In this figure is represented the Main settings layout.

already inflated in memory managed by a Fragment class called ***BciAudioFragment***. When the desired Tab is pressed, the Fragment associated to the current selected Tab is the current visualized in the layout container defined by an id in the ViewPager. One Tab is called *MAIN SETTINGS* and visualize a Fragment that is the main screen visualization, while the other one is called *ADVANCED SETTINGS* and visualize another Fragment containing the advanced options for the training session. The interface allows also to switch between these two Fragments with a swipe to the right and to the left. In Figures 4.6 and Figure 4.7 renderings of the main settings and the advanced settings Fragments is represented.

In the OptionMenu there is the "RESET" ImageButton. Clicking on new instance is created and an AlertDialog Fragment called ***ResetSessionDialogFragment*** is visualized. In this layout is asked to restore all the parameters in the training interface at their initial state.

4.6.1 Main Settings layout

The Main Settings layout is composed by an array of eleven CheckBoxes, an array of eleven Buttons to synthesize the sound loaded in the associated channel, and an array composed by eleven Fragments inflated dynamically through Java code as audio channels. Those Fragments are a repetition of a unique Fragment class called ***BciAudioChannelFragment*** because they share the same layout.

The Main Settings layout is designed like an audio mixer where the user can select a channel, load an audio file (.wav file from the SD card in a specific folder) and deliver it with the in-ear headphones to the participant

Table 4.2: Mapping between the float volume levels and the Loudness expressed in dB.

Float level	*Loudness level*
0.2f	55 dB
0.5f	65 dB
1f	75 dB

(see Section 3.8 on page 72 for details). With an up-down scrolling on screen is possible to visualize all channels.

Each audio channel before running has to be activated with the associated CheckBox. An ImageButton in the audio channel layout creates a new instance which opens a window. The Fragment class initialized here is the ***AudioBrowserDialogFragment*** that is a customized Dialog which shows a list of all the .wav audio files found in the default folder: *"ON/Acoustic_yes_no_BCI/sounds/"*. After the selection of an audio file, the customized Dialog is dismissed. The audio is loaded in the current channel selected its string path location, its name and its size are visualized in a small gray layout as shown in Figure **??**. When the audio file is loaded into the channel, also the other components of that channel are activated. Spinner contains a list of three values of Loudness Level (see Section 2.2.3 on page 18 for details) as the Intensity volume perceived from the participant in the ear expressed in decibel [dB]. These values are mapped in Java code as float from $0.0f$ to $1.0f$ according to the output volume measured from the in-ear headphones explained in section 3.8 on page 72. In Table 4.2 are defined these mapping values associated.

It is possible to delete the current audio file in the channel holding down the "X" ImageButton. This operation set all the parameters at their beginning state.

Then there are two Switches that define the directionality of the audio file loaded. The audio is synthesized in STEREO with a selection of both Switches, while if it is selected only one of them the output will be MONO LEFT or MONO RIGHT.

After setting the parameters defined before it is possible to test the current audio loaded in the channel with a long selection on the Button "TEST SOUND" (see Section 4.11 on page 108 for details).

Another Spinner defines the probability of occurrence as the percentage of stimuli presentation defined in the Three-stimulus Oddball Paradigm (see Section 2.4.6 on page 39 for details) in order to make possible by the application to understand if in the current channel selected there is a frequent standard tone (70%) or a target/deviant tone (15%).

It is possible to select all the eleven channels and set them in a customizable way, but for a proper sequence generation for the paradigm must leave selected only three channels: one channel for the frequent sound and the other two for the target and the deviant sounds. Deselecting a channel maintains all the parameters defined in order to select it later.

4.6.2 Advanced Settings layout

The Advanced Settings is initialized when the Tab associated in the Training Fragment is pressed. When it happens the layout of the class ***BciAudioStimuliHeaderFragment*** is visualized.

This layout is composed by others four Fragments controlled and inflated dynamically through Java code. Also this layout can be scrolled up-down and a simple render is shown in Figure 4.7.

The specification defined by Mauro Marchetti for the SOA (Stimuli Onset Asynchrony) and the ISI (Interstimulus Interval) with their advanced settings timings defined in Section 3.1 are accessible and editable from user here in the Advanced Settings layout.

The main elements in this layout are the SOA, the ISI and the FixSTD CheckBoxes. The selection of the SOA or ISI automatically checks the FixSTD CheckBox.

In this layout the first Fragment class called ***BciAudioStimuliNumberFragment*** is inflated. It is the one referred to the number of stimuli selection. An EditText is defined here where it is possible to edit and update the default value set as seven stimuli. Clicking on it, a new instance and a popup window is is initialized as in Figure 4.8. The window created is a Dialog with a customized layout inflated with the Fragment class called ***EditDialogFragment***. In this layout there are the current number taken from the EditText that can be incremented/decremented by one unit clicking on the arrows or just sliding the progress bar to the right or to the left. The "DEFAULT VALUE" button set the current value visualized on the Dialog to the default and the "OK" confirm the new one updating the Edit Text.

The other three Fragments inflated are the advanced timings properties for the SOA or ISI selection. They are the *Fixed Interval*, the *Intervals* and the *Random Intervals* Fragments. As defined in Section 3.1 on page 64 these Fragments are chosen in this way:

- ***BciAudioStimuliFixedIntervalFragment*** class allows to define a fixed timing value between two consecutive stimuli in the sequence.

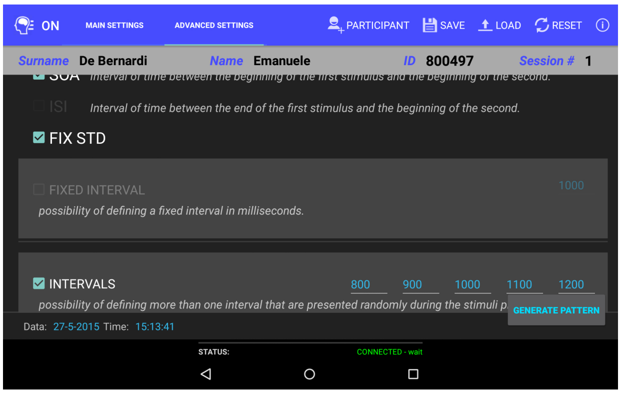


Figure 4.7: Rendering of the Training Session. In this figure is represented the Advanced Settings layout.

- ***BciAudioStimuliIntervalsFragment*** class allows to define five timing values and set them in a randomized order by the program between two consecutive stimuli during the sequence.

- ***BciAudioStimuliRandomIntervalsFragment*** class allows to define a minimum and a maximum timing value and set a randomized timings caught between these two limits in the sequence between two consecutive stimuli.

Also in the Advanced Settings layout, in order to have a best accessible interface for defining correct timings and the correct randomized sequence generation, initially all these three Fragments inflated are disabled.

An algorithm enables/disables them and their elements. When the user selects the SOA or the ISI CheckBox the complementary timing pattern type is disabled (i.e., if SOA is selected ISI is not accessible until the SOA is deselected and vice versa) and all the three Fragments inflated as advanced settings are initialized. When the user deselects the timing pattern, the system verify if the timings values are defined or not. If not, they are updated to their default values, otherwise the value entered previously is restored.

Each Fragment (Fixed, Intervals and Random) inflated in the Advanced Settings layout contains a CheckBox and EditTexts (initially disabled for all the three Fragments) to allow the insertion of the updated timings. When the user checks the desired advanced interval timing, the EditTexts of the current Fragment selected are enabled and the other two Fragments with their elements are completely disabled. At the selection of an Edit Text a

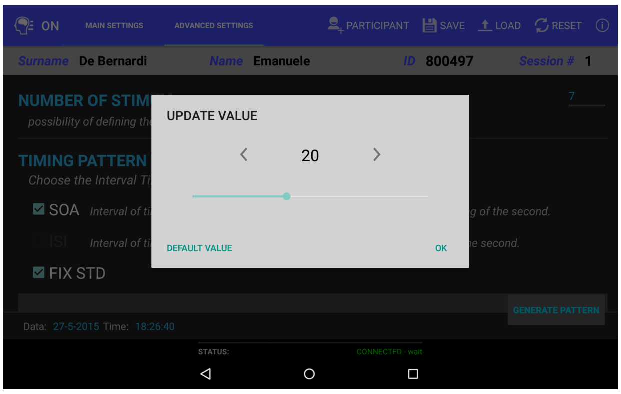


Figure 4.8: Rendering of the appearance of a customized Dialog on EditText selection. The user can update the value clicking on the arrows incrementing/decrementing by one unit or sliding the progress bar.

new instance with a Dialog Fragment (***EditDialogFragment***) composed by a customized layout is initialized like in Figure 4.8 and the current value can be updated.

4.7 Session management

4.7.1 Save session algorithm

In the ActionBar [27] of the Training interface layout there is the “SAVE” Image Button as shown in Figure 4.6 on page 90. After the session initialization (see the beginning of Section 4.6) and all the desired settings in the Main and Advanced Settings, the user can save the session in a specific .xml file in a default folder on SD card just clicking on it. After that, a new instance is created and an AlertDialog is visualized on screen like in Figure 4.9 where the user can confirm or cancel the operation. The class that performs this operation is called ***SaveSession***.

The ***saving session algorithm*** is based on SharedPreferences [33] that is an interface for accessing and modifying data. When the user select the “OK” button it starts the algorithm for the current user ID and the current session set calling the *onSavePreferences()* function in the referenced Java class.

According to the SharedPreferences the algorithm creates for each elements defined in the Training interface a key string that is associated to the current state of the element. This pairing is then saved in an editor called from SharedPreferences.Editor. The type of the element (i.e., String,

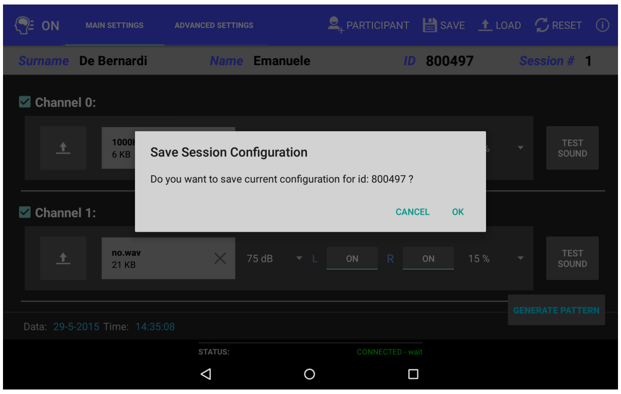


Figure 4.9: Rendering of the appearance of an AlertDialog on Save session ImageButton selection. The user confirm or cancel the operation.

Integer, Boolean, Float) is associated to its value and then the editor is committed.

The path *"ON/Acoustic_yes_no_BCI/configurations/"* is defined on the SD card as the default folder where the SharedPreferences files are saved. In this path a subfolder named as the patient ID defined in the current session (i.e., *"/configurations/800493/"*) is automatically created. Before being created, it is verified that the folder is already existing. If not that folder and the output .xml file named as:

ID_**idvalue**_session_**N**_config.xml

is created. In that file are inserted all the pairs made by the editor with the Java command *objectOutputStream.writeObject(sharedPreferences.getAll())*. The variables *idvalue* and N are the participant ID and the session number for the current session.

4.7.2 Load session algorithm

When the user wants to restore a previous session clicks on "LOAD" ImageButton in the Training interface menu. As shown in the right image in Figure 4.6 on page 90, it possible to select the desired session for the desired participant ID.

Clicking on the ImageButton a new instance is created that shows a customized Fragment Dialog with the class ***LoadSessionDialogFragment*** where its layout contains a list of ID folders inside the default path *"ON/Acoustic_yes_no_BCI/configurations/"*. The user here can select a desired folder or delete it with all the configuration files inside, just holding it down and confirming the operation. If the desired ID folder is selected, all configuration files inside are visualized in a list. It is also possible to delete a configuration file holding it down and confirming or simply selecting it to restore all the items configuration saved.

Also in this case the SharedPreferences is used to restore all the elements associated with their corresponding state saved on the .xml file. This operation is performed by the class ***LoadSession*** and it is a little more complex than the saving one. A sort of information retrieval on the .xml file selected is performed in order to get the type of items and then restore them. This operation is performed by the ***loading sessions algorithm***. When the .xml file in the customized Dialog is selected, the class that deals with the items restoration gets the entire path and the name associated to that file in its constructor function. Then the *onLoadPreferences()* function is called where the file is imported with the *ObjectInputStream* in order to load the primitive types.

The global context in the Activity is used to get the SharedPreferences associated to the file name. In order to restore the state for all the elements, the SharedPreferences.Editor is used recalling the key String associated to the states saved in the .xml file.

A *Map<String,?>* element is initialized so as to associate to it all the pairings between the strings and objects from the input file. For each entry value it is obtained the current *Key String* and the current *Object Type*. When the type for the current entry is checked the pairing result is written in the SharedPreferences.Editor. When all the entries are processed the editor is committed and the preferences are restored for each element in the Training interface.

In Listing 4.2 there is an example of the algorithm for saving and loading. For this example the patient surname string and SOA CheckBox boolean state are saved and restored from the .xml file on SD card.

Listing 4.2: Listing of saving and loading session algorithms examples.

```
...
// define the default path
private final static String default_path = "ON/Acoustic_yes_no_BCI
    /configurations/";
...
/*
 * saving session algorithm
 */
private void onSavePreferences() {

    // define the saving file name
```

```java
String pref_file = "ID_" + getID() + "_session_" + getN() + "
    _config";
// define SharedPreferences
SharedPreferences save_session;
// initialize shared preference object
int mode = getContext().MODE_WORLD_WRITEABLE;
save_session = getContext().getSharedPreferences(pref_file,
    mode);
// use an editor to maps the key value with the state object
SharedPreferences.Editor editor = save_session.edit();
...
// pairing example for saving patient surname string
editor.putString(key_surname, getSurname());
...
// pairing example for saving SOA CheckBox boolean state
editor.putBoolean(key_SOA, getSOAstate());
...
// commit
editor.commit();
...
// create a new path for the ID
String ID_path = default_path + getID() + "/";
// associate the path to a file
File configuration_path = new File(Environment.
    getExternalStorageDirectory().getPath() + ID_path);
// check if the path exist
if (!configuration_path.exists()) {
    // create directory if does not exist
    configuration_path.mkdirs();
} else {
    Log.d(TAG, "folder:␣'" + ID_path + "'␣already␣exists!");
}
// create a new file to generate the xml
File file = new File(Environment.getExternalStorageDirectory()
    .getPath() + ID_path, pref_file + ".xml");
// create ObjectOutputStream
ObjectOutputStream output_file = null;
try {
    // create output file and write the file in the directory
    output_file = new ObjectOutputStream(new FileOutputStream(
        file));
    output_file.writeObject(save_pref.getAll());
} catch (IOException e) {
    e.printStackTrace();
} finally {
    try {
        if (output_file != null) {
        // close output file
        output_file.flush();
```

```java
                output_file.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    /*
     * Loading session algorithm
     * @config_path:    get the full path of the file selected
     * @file_name:      get the name of file selected
     */
    public void onLoadPreferences(String cofig_path, String file_name)
         {
        // import file
        File file = new File(config_path);
        ObjectInputStream input_file = null;
        try {
            // define SharedPreferences
            SharedPreferences load_session = null;
            // get the preferences
            int mode = getContext().MODE_WORLD_WRITEABLE;
            load_session = getContext().getSharedPreferences(file_name
                ,mode);
            // create the editor to restore mapping between key
                strings and values of the objects
            SharedPreferences.Editor editor = load_session.edit();
            editor.clear();
            // read all entries from file loaded
            Map<String, ?> entries = (Map<String, ?>) input_file.
                readObject();
            // for every values
            for (Map.Entry<String, ?> entry : entries.entrySet()) {
                // get all values from input file xml
                Object object = entry.getValue();
                // get all key Strings from input file xml
                String key_string = entry.getKey();
                // put boolean values to shared preferences
                if (object instanceof Boolean) {
                    editor.putBoolean(key_string, (Boolean) object);
                } else if (object instanceof Integer) {
                    // put integer values to shared preferences
                    editor.putInt(key_string, (Integer) object);
                } else if (object instanceof Float) {
                    // put float values to shared preferences
                    editor.putFloat(key_string, (Float) object);
                } else if (object instanceof String) {
                    // put string values to shared preferences
```

```java
                editor.putString(key_string, (String) object);
            }
        }
        // commit the editor
        editor.commit();

        /* restore the preferences */
        ...
        // example to retrieve patient surname string
        patient_surname = load_session.getString(getKey_surname(),
             "");
        // set the value
        setSurname(patiet_surname);
        ...
        // example to retrieve the boolean state of SOA switch
        SOA_state = load_session.getBoolean(getKey_SOA(), false);
        // set the value
        setSOA(SOA_state);
        ...
    } catch (ClassNotFoundException | IOException e) {
        e.printStackTrace();
    } finally {
        try {
            if (input_file != null) {
                input_file.close();
            }
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
}
```

4.8 The 10-20 system implementation

The application "Progetto ON" originally developed for the Visual Speller P300 [46] was provided with the activation and processing of the data acquired from four EEG channels (Fz, Cz, Pz, Oz) and one EOG channel (see Figure 2.7 at page 26). During the development of the Auditory P300 BCI the acquisition channels to the 10-20 system it was decided to be expanded enabling all the nineteen unipolar channels (Fp1, Fp2, F7, F3, Fz, F4, F8, T3, C3, Cz, C4, T4, T5, P3, Pz, P4, T6, O1, O2 as in Figure 3.3 on page 68 on the BELight device) keeping activated the bipolar channel EOG as before (A-, A+ as in Figure 3.3 on page 68 on the BELight device).

The main work in this phase was to allow the user to choose and select the desired acquisition channels directly on the screen and processing data only

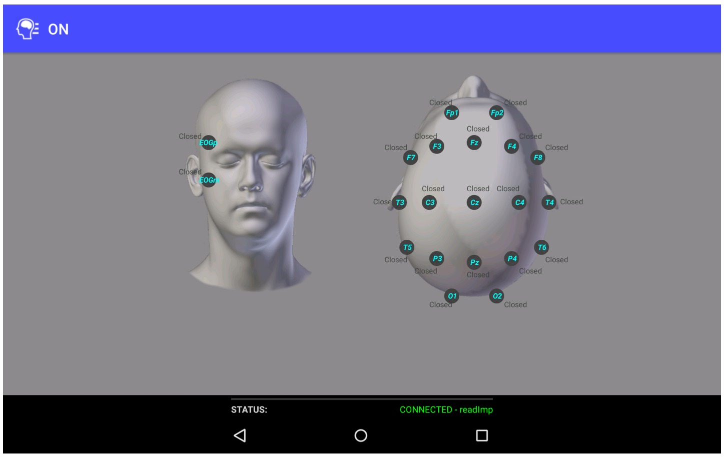


Figure 4.10: Rendering of the impedance control panel. The left image represents the EOG electrodes visualization rendered on a frontal head model. The right image represents the 10-20 system visualization rendered over a top head model.

for those channels. To this purpose, drivers and all the specific functions that recall the interfacing channels developed before are modified and adapted.

4.8.1 Impedance control

Selecting the "IMPEDANCE CONTROL" button in the Acoustic yes-no BCI menu (see Section 4.3 on page 81 for details) the Fragment called ***BciImpedanceFragment*** for the visual representation of the 10-20 system is inflated following the Fragment layout switching algorithm described in Section 4.5 on page 85. In this Java class the ***SamplesBuffer*** and the ***Communication*** classes pointers are passed in the constructor function and the main thread that handles the BELight states is set to *readImp* status. This status is the equivalent of the impedance analysis mode.

The main layout of this impedance control Fragment is composed by a Java class that extends a SurfaceView [35] called ***BciImpedanceRender***. This class provides a dedicated drawing surface embedded inside a view hierarchy. This is the graphical interface that works with a thread class called ***BciImpedanceThread*** that animates the elements updating the visualization changes of the electrodes states on screen device after their activation by user selection. Figure 4.10 represents the graphical object that contains a front head model rendered on the left part of the screen with the EOG electrodes visualization while on the right the complete 10-20 electrodes system is rendered over a top head model.

Through the *Communication* thread all the channels are read. Initially all the 21 electrodes (2 electrodes for the EOG and 19 for the 10-20 system) are in a default “closed” state in grey color. It means that all the electrodes are not acquiring signals from their referred channels.

4.8.2 Gesture detector algorithm

The user can select the desired electrode directly on the screen by holding it and set it in an “open” state. This selection is defined by an algorithm that finds the user input gesture on the screen called *Gesture Detector algorithm* defined in Listing 4.3. The GestureDetector is called by an overridden function in the SurfaceView defined as *onTouchEvent()* that passes an event as the touch on the screen.

In this algorithm initially the coordinates (x and y pixels) of the touch input on the screen are retrieved with the MotionEvent function. Each electrode point rendered on the screen has a 20 pixel radius. The algorithm checks for each electrode if the current coordinates of the pixel selected are inside its centroid of radius 20 pixel. Then the current state between “closed” and “open” is verified and it is classified to be set in the opposite current state.

After the screen selection of the electrode, other states are considered to evaluate the electrode impedance connected to the subject head skin. They are “verybad”, “bad”, “decent”, “good”, “optimal” all mapped with a specific color from red to green. These states are evaluated with some threshold values in this way:

- verybad (red): $< 100\ K\Omega$
- bad: $< 50\ K\Omega$
- decent: $< 15\ K\Omega$
- good: $< 10\ K\Omega$
- optimal (green): $< 5\ K\Omega$

Listing 4.3: Gesture Detector algorithm. Initially all the electrodes are in a default closed state. The user can set them to an open state by a MotionEvent detector that obtains the coordinates of pixel touched and the centroid of points with radius 20 pixels is checked for correct selection classification.

...

```java
private final GestureDetector gestureDetector = new
    GestureDetector (new GestureDetector.SimpleOnGestureListener()
     {
    public void onLongPress(MotionEvent event) {
        // get x pixel coordinate touched
        int x_touched = (int) event.getX();
        // get y pixel coordinate touched
        int y_touched = (int) event.getY();
        /* for all the electrodes plotted check if the coordinates
             of pixel selected are inside of the centroid (center
            of the electrode) of radius 20 pixels.*/
        for (int i = 0; i < electrodes; i++) {
            if (x_touched >= x_elec_pos - 20)
                && (x_touched <= x_elec_pos + 20)
                    && (y_touched >= y_elec_pos - 20)
                        && (y_touched >= y_elec_pos + 20) {
                // check if the current electrode is checked or
                    not
                if (!checked) // if not
                    // set it to open state
                    setElectrodeChecked(i, true);
                else
                    // set it to closed state
                    setElectrodeChecked(i, false);
            }
        }
    }
});

@Override
public boolean onTouchEvent(MotionEvent event) {
    return gestureDetector.onTouchEvent(event);
}
...
```

4.9 Dynamic data processing

4.9.1 Signals visualization

This section explains how data are acquired and dynamically processed depending on the activation of the channels with the user selection. Before requesting data to the circular buffer driver class, at least one electrode must be selected. It is then possible to select "VIEW SIGNALS" button in the Acoustic yes-no BCI menu in order to check the data elaborated only from those channels selected. A new Fragment called ***BciViewSignals-Fragment*** is inflated over the current layout following the Fragment layout

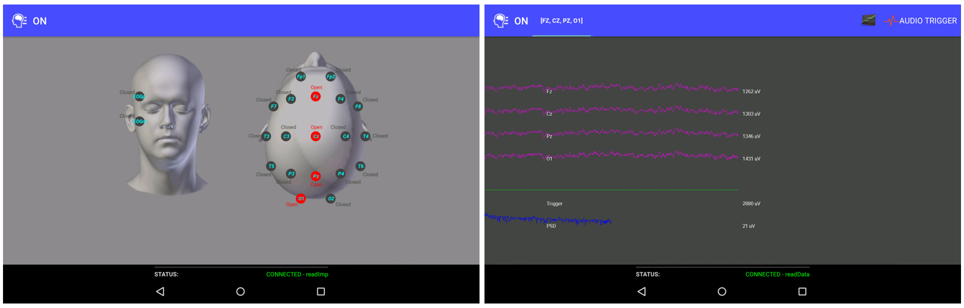


Figure 4.11: An example of the main central electrodes selection (Fz,Cz,Pz,O1) with the corresponding signals elaboration.

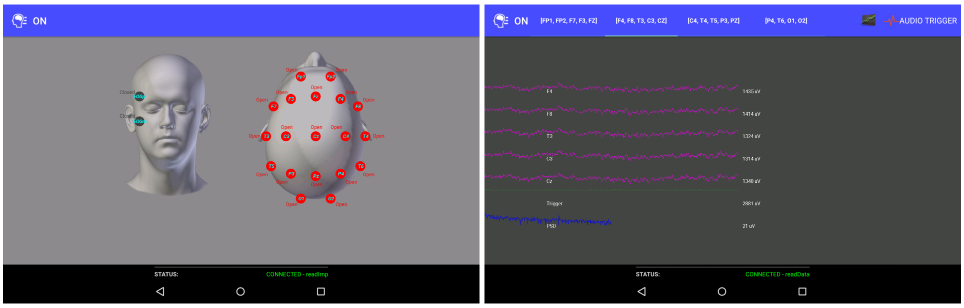


Figure 4.12: An example of all electrodes selection (10-20 system) with the corresponding signals elaboration.

switching algorithm (see Section 4.5 on page 85 for details).

In this Java class the ***SamplesBuffer*** and the ***Communication*** classes pointers are passed in the constructor function and the main thread that handles the BELight states set it to *readData* status.

This Fragment inflates another Fragment class called ***BciDynamicViewSignalsTabHost*** used to retrieve the electrodes selected by user from the impedance Fragment and draws only the data signals processed associated to those electrodes. All the desired data obtained from the *SamplesBuffer* are subdivided automatically in groups of five signals each. This in order to allow the correct visualization of all signals on the tablet screen.

The class uses an algorithm that gets the electrodes selected, subdivides them in groups of maximum five signals each, and finally adds a tab for each group created labelled with the current name of electrodes visualized. The number of tabs is defined with the ActionBar [27] method with the number of groups created by the algorithm.

This class at its creation implements the rendering of the signals labelled in the first Tab selected. Another class called ***BciViewSignalsRender***

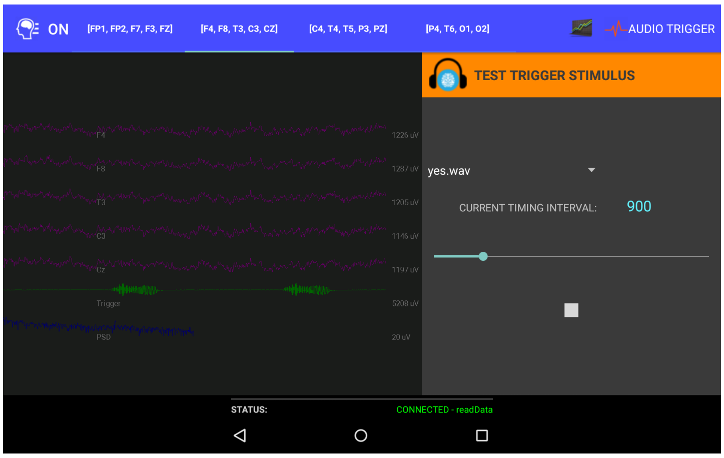


Figure 4.13: Advanced menu for audio trigger player in view signals Fragment.

(that extends the SurfaceView [35] method), provides a dedicated drawing surface with another thread class called ***BciViewSignalsThread*** that controls the animation. At each tab selection only the maximum five signals added to the current group on the screen by a proper thread are rendered. In Figure 4.11 an example of the main central electrodes selection (Fz, Cz, Pz, O1) with the corresponding signals elaboration is shown, while in Figure 4.12 an example of all electrodes selection (10-20 system) with the corresponding signals elaboration is shown as a comparison.

A unique function model is created to draw signals in the rendering class. The main thread that controls the animation of signals asks to the *SamplesBuffer* all samples data (all the channels). Then passes them to the rendering class in order to compute and to update the desired data for the current group of signals labelled in the current tab selected by user. After this, data updated are rendered on screen.

As we can see in Figures 4.11 and 4.12 the class that extends SurfaceView defines the data signals in a magenta color. After the tab selection the current label and the peak-to-peak value are computed and visualized.

The trigger value is the representation of the peak-to-peak signal for the input audio source in the EEG amplifier that comes directly from the tablet and adapted. This input signal is adapted with a designed cable (see Section 3.9 on page 74 for details). During the visualization of signals, the subject can hear a simple pure (or complex) tone or a word through the in-ear headphones.

4.9.2 Trigger visualization

The interface is developed also to do this operation in the view signals Fragment. This class implements a NavigationDrawer[32] hidden on the right side of the screen. Clicking on the "AUDIO TRIGGER" button in the ActionBar, the player console appears as in Figure 4.13. The layout of this extended interface is inflated from the ***BciViewSignalsPlayTrigger*** class. Initially the *onCheckChannels()* is called to retrieve automatically the name of each audio file loaded in the respective audio channels (frequent sound, deviant 1 and deviant 2) previously set in the training interface and they are associated to a specific array list in the Spinner menu visualized in the figure.

The acoustic synthesis of the audio selected is executed by a thread called ***BciAudioTriggerThread*** that runs in loop until it is decided that it should be terminated. The thread is initialized pressing the play icon in the interface and the current acoustic stimulus selected is synthesized in loop with a delay interval defined. Doing this it is possible to visualize directly in the application the audio trigger onset and its relative brain signals stimulation in time. The thread is stopped by clicking on the stop icon and the stimulus will not be sent both to the participant and into the EEG amplifier.

When the "VIEW SETTINGS" (the signals icon in the ActionBar) is pressed a customized Dialog appears. Here it is possible to select the current Power Spectral Density (PSD) associated to a desired data channel processed with a Fast Fourier Transform (FFT). Initially the PSD is calculated on the data that comes from the first electrode rendered on screen.

4.10 Timings pattern generation algorithm

When the desired electrodes are selected and the correct visualization of signals stimulated by the trigger audio signal is performed, it is possible to go back to the TRAINING SESSION trough the Acoustic yes-no BCI menu if the session is already initialized or to create a new one if not (see Section 4.6 on page 88 for details).

In order to generate a proper acoustic session with the oddball paradigm all the elements have to be set in a correct way. In Figures 4.6 on page 90 it is possible to see that in the OptionsMenu there is an ImageView with an "i" symbol. At its selection a new instance with a customized Dialog is initialized showing the *User Guide* where it is explained in detail how to set every elements.

Clicking on the "GENERATE PATTERN" button all the parameters are processed by the software and the sequence of stimuli and timings between them is automatically generated.

The application developed does not initialize the pattern generation algorithm if all the elements required for the generation (training interface and the electrodes selection) have not been defined correctly. This check is to avoid errors or crash report by the application.

In order to obtain an audio stimulation a check algorithm function called *onCheckConfiguration()* has been developed in the main Fragment class ***MainBciAudioFragment***. As defined in Section 4.6 on page 88 (under the Main Settings section) it is possible to select all the eleven channels and set them in a customized way. For a proper sequence generation only three channels are needed: one for the frequent sound and the others for the two deviants. At beginning the check algorithm verifies this condition (at least three channels). For each channel selected the system verifies the correct audio loading, the loudness selection, the almost one volume switch selection and the percentage of occurence definition in the sequence (70% for standard and 15% for deviants). Also the pattern type (SOA or ISI) and its advanced timings settings are verified. In order to do this, the application alerts the user.

After parameters confirmation, a function called *onWaitConfiguration()* is executed. This function runs an AsyncTask creating a new background thread that waits 500 milliseconds at each iteration. This is done in order to achieve the best loading configuration of channels and parameters avoiding errors (see Section 4.11 on page 108 for more details).

The software reads each selected channel and initializes a specific Java class called ***AudioParameters*** where all the audio elements are prepared for the correct acoustic sequence synthesis (see the next Section 4.11 for details). Another class called ***AudioConfigurator*** is initialized. In this class all values are stored in specific arrays. A function called *onSession-Config()* get the timing intervals for the pattern type defined (SOA or ISI) and initializes other classes to generate the sequence of stimuli and timings between them.

The first class initialized is the ***Session***. It generates the expected number of frequent sounds and deviants for the stimuli sequence saving them in arrays accessible from the other classes.

The second class initialized is the ***SequenceSyntheser*** that elaborates the randomized sequence of stimuli from user input parameters. The *Session* class passes its pointer in its constructor. This sequence generated with a sequence of integers defined as 0;1;2 in a randomized order is mapped. 0

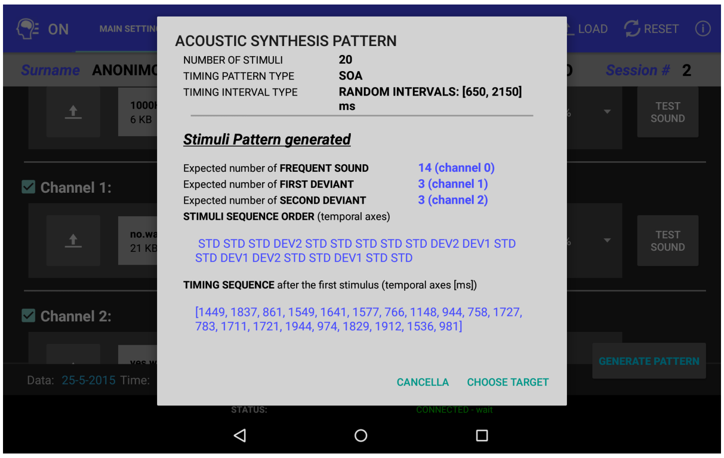


Figure 4.14: An example of pattern generation algorithm results shown to the user. A stimuli pattern generated for the oddball is shown in detail. In this figure an example of the algorithm summary is visualized before the acoustic training session is initialized. This customized dialog shows on top the parameters setted in the interface. The expected number of frequent sounds, the first and the second deviants expected with their relative channels associated to load them by the software are defined. At the end the stimuli sequence order generated by the algorithm in temporal axes and the timing sequence after the first stimulus expressed in milliseconds are shown.

means the frequent sound, while 1 and 2 are referred to the first and second deviants. This sequence is stored in an array accessible from the other classes.

The last class called is the ***TimingSyntheser***. This class generates intervals between each stimulus for the entire sequence. In its constructor the *Session* class passes its pointer in order to access to the randomized sequence of integers. For each integer of the sequence array, the current timing (fixed, intervals or random intervals) is checked and it is generated a timings array N-1 long where N is the number of elements in the sequence array like in the Example 4.10.1.

Example 4.10.1. *Final timing sequence example.*

$$sequence = [s_1, s_2, s_3, ..., s_N];$$
$$timings = [t_1, t_2, t_3, ..., t_{N-1}];$$
$$final\_sequence = [s_1, t_1, s_2, t_2, s_3, t_3, ..., s_{N-1}, t_{N-1}, s_N];$$

If the current timing pattern type is the *fixed interval*, a timings array composed by the same value after each element s_i (where i = 1,...,N) of the sequence is generated. Else if the *intervals* is selected, a timings array

array composed by a randomized order of values associated to the intervals pattern defined is generated. Else if the *random intervals* is selected, a timings array composed by a randomized values between the two defined is generated. The timings array built is accessible from the other classes.

After the *onWaitConfiguration()* function execution a new instance that initialize a Dialog Fragment called ***AudioElaborationDialogFragment*** with a customized layout is created.

In Figure 4.14 an example of pattern generation algorithm is shown to the user. This Dialog shows on top the parameters set in the interface. The expected number of frequent sounds, the first and the second deviants expected with their relative channels associated to load them by the software are defined. At the end the stimuli sequence order generated by the algorithm in temporal axes and the timing sequence after the first stimulus expressed in milliseconds are shown.

4.11 Synthesis of the acoustics stimuli

This section explains in detail how the audio is processed and synthesized in the application. Audio processing in Android devices is performed by two main libraries called *SoundPool* [34] and *MediaPlayer* [31].

MediaPlayer class can be used in general to control playback of audio files and streams. This class is used to synthesize the audio guides that inform the subject about the status of the session through the in-ear headphones.

A SoundPool is a collection of samples that can be loaded into memory from a resource inside the APK or from a file in the file system. The SoundPool library uses the MediaPlayer service to decode the audio into a raw 16-bit PCM mono or stereo stream. This allows applications to ship with compressed streams without having to suffer the CPU load and latency of decompressing during playback. Being the oddball defined with a sequence composed by stimuli with short duration that have to be loaded and played quickly in an efficient way, this library it has been chosen also for the low latency property.

The Training Fragment (Section 4.6 on page 88) in the Main Setting layout has eleven channels. The Fragment class that creates in its layout these channels is the ***BciAudioFragment***. In each channel it is possible to load a the preferred audio tones or words for binary selection from the SD card file.

To this purpose two model classes are created to synthesize the stimuli in the channels selected. One class called ***AudioParameters*** controls all parameters defined by the user for each channel selected with the SoundPool class. All objects created in this class are defined as arrays composed by eleven elements (like the number of channels) in order to store for each channel its volume level i.e, the loudness defined and a SoundPool identifier that is passed to another class to play sounds. When the class is initialized in its constructor function the pointers of the Activity context and the *BciAudioFragment* class are passed to the AudioParameters class.

Its public functions defined in this class is mainly associated to the channels parameters. The *onCheckSwitches()* function obtains the integer pointer to the channel focused. Here the channel selection and which one of the two switches are selected by defining Boolean variables as *mono_left*, *mono_right* and *stereo* are verified.

Another function called *setChannelVolume()* obtains the current pointer to the channel and initializes the AudioManager that provides access to the volume by Android OS. Here the loudness set by the user for that channel pointed is obtained and associated to a Float array.

Once these parameters are defined, the SoundPool is initialized with a public function called *onResetParameters()*. A sound ID is associated to the channel pointed, the audio path of that channel is loaded and a *setOnLoadCompleteListener()* function setting a positive Boolean variable it is waited. In this way all .wav files are associated to their ID and they are loaded in memory before the audio sequence is sent. This in order to perform a correct sounds synthesis avoiding buffering problems during the sequence of stimuli.

Other public functions are defined to set the audio loaded path in the channel pointed, and the probability of occurrence defined by user that is passed to ***Audio*** class (initialized in the ***AudioConfigurator*** to compute the sequence of the stimuli) is obtained. A specific public function defined here is *onGuideVolume()* in order to prepare the volume for the audio guides during the session to the participant. The audio volume is initialized for MediaPlayer class and it is made available to be modified with the device hardware buttons. The other class that performs the audio synthesis is the ***AudioSynthPlayer***. This class synthesize the audio for each channel selected using the configuration acquired by the ***AudioParameters*** class. To calculate the duration of the current stimulus the MediaPlayer class is used. When this class is created, in its constructor, the SoundPool created in the AudioParameters class passes the pointers, and also the main Activity passes its context.

The main public function that can be called from the other classes is the *onDirectionSynth()*. The current channel and its sound ID pointers are obtained. The MediaPlayer is prepared and when the stimulus starts calling *getCurrentPosition()* function is verified. The duration of the stimuli with the *getDuration()* function is calculated. From the AudioParameters class the Boolean variables *mono_left*, *mono_right* and *stereo* are obtained and for the current state another function *onPlay()* is called, the current sound ID is passed and at the end the stimulus is played.

The Training Fragment (Section 4.6 on page 88) in the Main Setting layout has eleven Buttons; each of them is associated an audio channel. When the audio is loaded in to a channel it is possible to synthesize it to test the correct functionality of the loudness and the directionality through the in-ear headphones to the subject. To perform this operation the ***BciAudioFragment*** is the Fragment class that creates in its layout these eleven Buttons. When one of these Buttons is held longer, it is verified if the loudness and at least one of the two switches of the current channel are set passing its Integer index to the AudioParameters and AudioSynthPlayer classes, so the sound in the current channel is synthesized.

In Section 4.10 on page 105 how the sequence and timings between stimuli are generated was describe and how the audio stimuli are synthesized with an algorithm in function of the two elements array generated in the SequenceSyntheser and TimingSyntheser classes is described.

As said in the previous section, in Figure 4.14 on page 107 there is an example of pattern generation algorithm results that shows to the user a summary of the sequence delivering. After the "SELECT TARGET" selection, is checked that at least one electrode is selected from user in impedance control Fragment. A new AlertDialog ***AudioUserGuideDialogFragment*** is created where a default audio is sent to the subject through the in-ear headphones (via MediaPlayer). This audio message informs to focus the attention on a precise stimulus that has to be defined by user.

In Figure 4.15 the Dialog Fragment appeared when the audio message it has been completely reproduced is represented. The class initialized here is the ***PlayDeviantDialogFragment*** and its customized layout is composed by a ListVew where automatically displaying the two deviants set. These deviants are obtained verifying the channels selected in BciAudioFragment class and for the three classes selected the two that have set the 15% in the percentage of occurrence are processed. The name of the audio files in these channels are displayed and the indexes of these channels are passed to an array composed by two elements.

At the selection of one of the two stimuli, it is synthesized to the subject through the in-ear headphones the one selected (via SoundPool class). This stimulus is set to MainBciAudioFragment class as the current target

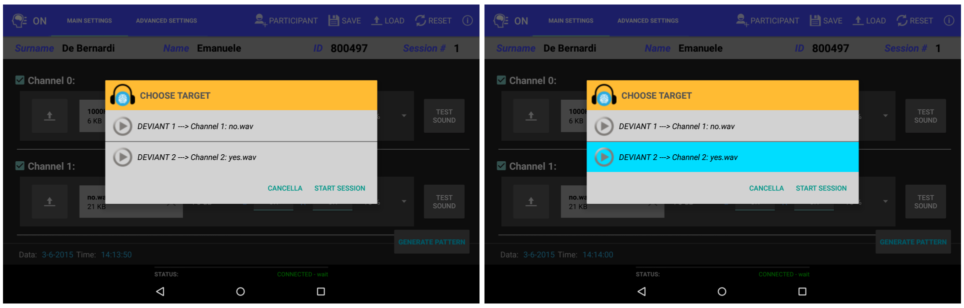


Figure 4.15: Current target selection. The left image shows a Dialog Fragment appearing to the user for defining the current target stimulus and the deviant one. The right image shows the definition of the target stimulus.

to generate the training files for the classification.

When the “START SESSION” button in the dialog is pressed (Figure 4.15), the current Dialog Fragment is dismissed and a another AlertDialog Fragment is initialized with the class ***AudioUserGuideDialogFragment***. During the visualization the subject is informed about the initialization of the session loading and playing a default .wave file (via MediaPlayer class) through the in-ear headphones.

In MainBciAudioFragment class a new Fragment called ***BciAudioSequencerFragment*** is initialized with the Fragment layout switching algorithm (see Section 4.5 on page 85 for details) hiding the TrainingFragment layout during its initialization. At its creation the Communication thread pointer, the TrialBuffer pointer, the SamplesBuffer pointer and the Activity context are passed in its constructor function.

This is the core class for delivering the acoustic sequence to the subject while the EEG signals selected in the impedance Fragment are acquired, processed and two training files are generated (see the next Section 4.12 for details).

Its layout is shown in Figure 4.16. The elements defined are animated with a class thread called ***BciAudioSequencerThread*** initialized after the definition of all the elements in the Fragment class (precisely in the *onResume()* function). Here also the ***AudioSynthPlayer*** class is initialized.

In the thread constructor function are passed the Activity pointer, the MainBciAudioFragment, BciAudioSequencerFragment, AudioSynthPlayer and Communication classes pointers. In order to obtain the right interval between each stimulus synthesized for all the entire sequence another parallel thread is created. This new thread controls the audio in background that gets the current interval retrieved from the array generated in the Tim-

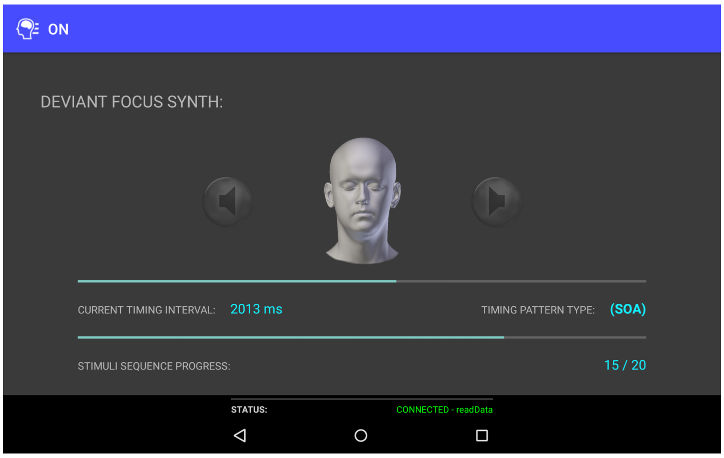


Figure 4.16: Layout for sequence of stimuli delivering in the training session. This layout is animated and these animation are controlled by a thread. Over the head is shown only the current target stimulus played. The two icons speakers at the sides are animated in function of the current stimulus directionality in the sequence. The first ProgressBar is animated upgrading up to the timing shown below and the second one is updated up to the total stimuli of the sequence.

ingSyntheser class.

The delay is computed in real time during the acoustic synthesis of each stimulus in the audio thread. Initially the timing pattern type (SOA or ISI set by user selection in the interface) is obtained. According to this each delay is computed considering the current timing get from the array and the duration of the current stimulus in the sequence with a function called *onTimingStimuli()*. If SOA is selected the current delay is considered as the current timing interval, while if ISI is selected the current delay is the sum of the current stimulus duration and the current timing. In order to create the entire sequence with the correct timings, the delay computed between each stimulus is defined as the sleep state of the audio thread.

While the audio thread is in sleep state, a new Handler for the UI (user interface) updating is defined. In this handler at each iteration the labels of the current deviant selected are visualized.

A new parallel thread that maintains the icon speakers in the "on" state (with the correct directionality) is created. This thread sets the icon speakers to a full state for all the duration of the current stimulus delivered.

Another thread on the runnable function is created to update the current timing visualization with a ProgressBar in order to visualize when the next stimuli will be delivered.

4.12 Training files

4.12.1 Files generation

When the ***BciAudioSequencerThread*** class is initialized, the *onCreateFileConfiguration()* function is executed. This function generates a .txt file which contains in every row the single information for each stimulus played in the sequence associated to the EEG stimulation.

The .txt file is automatically compiled and generated in a default directory on the SD card: "*/ON/Acoustic_yes_no_BCI/training/**currentID**/*". Inside this folder a subfolder named as the current ID where the log file is stored is automatically created. This file generated is named as:

id_surname_Acoustic_yes_no_BCI_session_**N_formatDate**.txt

where the values *id*, *surname* and *N* are the current participant id, surname and the number of session defined. The *formatDate* at its definition is composed with the function *SimpleDateFormat* expressed as "*dd-MM-yyyy_HH:mm:ss*".

The first row of the file is compiled with the timestamp. Next rows are iteratively compiled at each stimulus delivered in the main acoustic thread. At the beginning of the audio thread a *System.currentMillis()* is called to monitor all the sequence temporal duration and monitor how much time passes after each stimulus delivered from the beginning. Here a *onTrial()* function is called to examine the current stimulus delivered at each iteration of the audio thread and its properties on a row in the log file. In Table 4.3 there is a description of each column compiled when a stimulus is delivered.

In the *onResume()* called in the thread class also other functions contained in the ***SamplesBuffer*** class are executed. This in order to access to the EEG data and thus obtain only the data from the electrodes selected by user. The main function called here is the *toggleOutEdf()* that obtains a boolean parameters to write another file which corresponds to a specific file format with an .edf extension (EDF file [49]). This file is named as the .txt file and saved in the same folder. In this function another class called ***EdfParser*** provides a static methods to write the data acquired in the .edf file format calling statically the *createEdfFileStream()* function.

4.12.2 Files management

In order to obtain the classifier parameters these two training files must be sent to the AirLab server with the Dropbox application installed on the Tablet (see Section 3.10 on page 77 for details). A special interface in

Table 4.3: Single row composition for each stimulus delivered in the training log file. In this table each element of the row with its description is defined.

Element	**Description**
1st	*Trial for the Auditory BCI always fixed to* ***1***
2nd	*The current stimulus numbered in the sequence (**from 1 to the total number of stimuli**)*
3rd	*The current stimulus coded:* - ***1*** *for freq sound,* - ***2*** *for first deviant* - ***3*** *for second deviant*
4th	*The current stimulus coded is the target* - ***1*** *target* - ***0*** *not target*
5th	***0*** *value by default*
6th	***0*** *value by default*
7th	*The **timing in milliseconds** of the current stimulus elapsed from the beginning of the sequence*

the “Progetto ON” application has been modified in order extend also the training files generated from the Auditory BCI.

In the Acoustic yes-no BCI menu there is an item called “TRAINING BCI SENDER” (as in Figure 4.3 on page 82). At its selection a new Activity class called ***BciResultSender*** is launched with its layout inflated while the ***MainBciAudioActivity*** is maintained opened in background. In Figure 4.17 is represented the main interface for delivering training files.

The new Activity implements a NavigationDrawer[32] at its creation and its layout is hidden initially. The user can choose to visualize the training files between the visual or the auditory BCIs by selection and a new instance calls the DialogFragment ***TrainingFolderDialogFragment*** class. It inflates a customized layout that shows the list of the ID folder inside a default training path passed. If the user wants to visualize the Visual BCI trainings generated, the default path “/ON/CopySpeller/” is passed, while if the user wants to visualize the Auditory ones, the “*/ON/Acoustic_yes_no_BCI/*” is the path passed as default. Selecting the desired ID folder all the training files generated inside that folder are listed.

The list of files visualized is obtained by the class ***ExplorerList***. Here is applied a filter to find all the data recording with the .edf extension in the current folder and then all of them are displayed as a list.

In the right image in Figure 4.17 it is possible to see how files are viewed

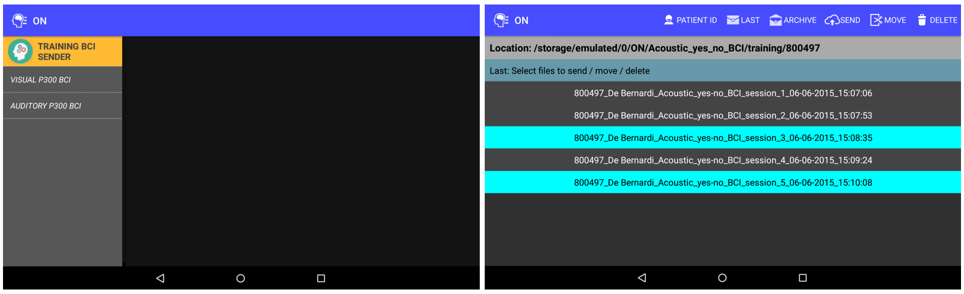


Figure 4.17: Main interface for delivering training files on DropBox folder. In the left image there is a screenshot where is possible to choose between the Visual BCI or the Auditory BCI training files visualization. The right image shows the complete interface for the training files management.

in the application. The main items in the OptionsMenu are also visualized. If the "PARTICIPANT ID" ImageButton is selected, appears again the DialogFragment where is possible to select another ID folder that passes its path and others training files generated associated to that ID are visualized.

It is also possible to archive files just selecting them and then click on the "MOVE" ImageButton. This operation moves both the .edf and the .txt files in the Archive folder.

Another possible operation is to delete data selecting files and clicking on the "DELETE" ImageButton. An AlertDialog appears to confirm the operation and both the .txt and .edf files with the same name are permanently deleted.

After the selection of the desired files, clicking on "SEND" ImageButton the application gets both the .edf and .txt files. If the tablet is correctly online and connected to the DropBox folder synchronized, a ProgressDialog is visualized with a ProgressBar update until all the selected files are not sent properly. The class ***DataServInt*** is imported to do this operation. From the Activity class the *newAuthConnection()* function in DataServInt class is called, a new thread class ***uploadThread*** is initialized and finally the files are uploaded to the server.

Chapter 5

Classification protocol

"Music is my religion."

Jimi Hendrix

As explained in Subsection 2.5.5 on page 50, the goal of BCI signal processing is to extract features from the acquired signals with a mathematical model. This is the first step called *feature extraction*. The second important step is the translation of these features into a specific command also with a mathematical model. This step is called *classification* and the BCI must decide whether a record signal belongs to two or more possible classes. In our case the P300-based BCI classifies each stimulus as target or non target.

In this chapter, our genetic algorithm (taken from [91]) tested with our Auditory BCI is described.

5.1 Feature extraction

The system used for the automatic feature extraction of P300s is based on a genetic algorithm. This algorithm operates on very simple features extracted to be used for the classification of P300 epochs, with almost no preprocessing. The epochs are classified in an affective way without developing the usual chain of information enhancement based on preprocessing, feature extraction, and classification. In our case, features are encoded in variable-length chromosomes, where each encodes one feature, and the fitness of an individual is given by the performance of a classifier trained on the encoded features. A classifier operating on P300 features is selected by GA.

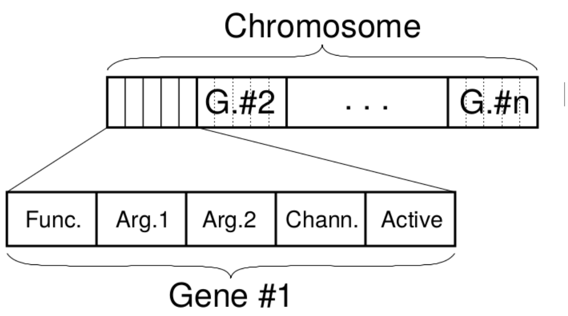


Figure 5.1: Structure of a chromosome encoding features.

5.2 Logistic Classifier

In this particular implementation, features are "tuned" on the use of a simple logistic classifier as already explained in Section 2.5.5 on page 50. A *logistic classifier*[83] approximates the probability $P(y|x)$ with a logistic function:

$$P(y = +1|x) = \frac{1}{1 + exp(\omega_0 + \sum_{j=1}^{n} \omega_j x_j)} \tag{5.1}$$

$$P(y = -1|x) = 1 - P(y = +1|x) = \frac{exp(\omega_0 + \sum_{j=1}^{n} \omega_j x_j)}{1 + exp(\omega_0 + \sum_{j=1}^{n} \omega_j x_j)} \tag{5.2}$$

where x_j are the n components of the vector x. The decision of the class to assign to a given sample x is taken by comparing the two probabilities $P(y = -1|x)$ and $P(y = +1|x)$. The parameter vector w can be found by maximizing, by using gradient ascent, its log-likelihood, with a term added to penalize large values of ω components:

$$L^{\lambda}(\omega) = \sum_{i=1}^{N} logP(y_i|x_i, \omega) - \lambda\|\omega\|^2 \tag{5.3}$$

5.3 Encoding

The chromosome of each individual in a population encodes a set of features, and its logical structure is shown in Figure 5.1.

A chromosome contains a variable number of genes, with an identical structure, and each gene is formed by five elements. The first three elements define a feature: the first one is an integer designating one feature extractor out of a predetermined set, while the two following elements encode two real-valued parameters for such an extractor. Feature extractors are functions

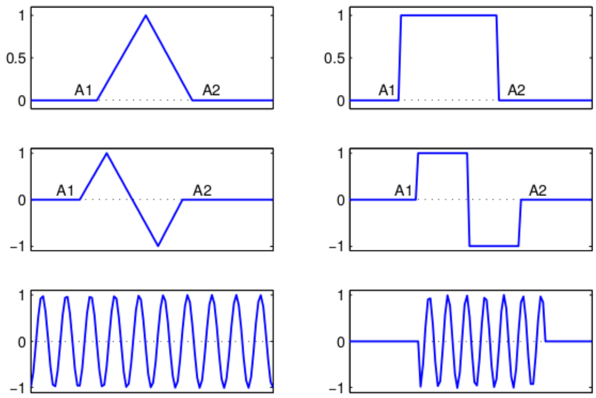


Figure 5.2: The weight functions encoded in genes.

with three arguments: a signal from which a feature is extracted, and the two parameters encoded in genes; these parameters are within the range $[0, +1)$, and their actual meaning varies from extractor to extractor. The fourth element of a gene is an integer number, which identifies the EEG channel where the feature encoded in the gene is to be extracted from. The last element of a gene is a Boolean flag that determines whether the gene is active or inactive. Inactive genes are not used to compute the fitness of a chromosome. Their role is of a genetic reserve, as they can be turned on in a later generation by mutation. The position of a gene within a chromosome is not significant.

Six different feature extractors are used, which share a very simple scheme: the input signal is multiplied by a weight function, and the result is integrated over time. In other words, feature extractors compute the cross-correlation between the input and a weight function. If we call $s(\cdot)$ the EEG signal from the channel the feature is to be extracted from, a feature extractor constructs a weight function $u(\cdot)$ from the parameters specified by the gene elements and then computes the resulting feature x with the formula:

$$x = \sum_{t=1}^{T} u(t)s(t) \tag{5.4}$$

The six weight functions used by the six feature extractors are shown in Figure 5.2. The feature that uses the weights shown in the top-right box is proportional to the average of the input signal over an interval; the extremes of the interval are determined by the two parameters (A1 and A2 in the figure) encoded in genes. The weights in the top-left box produce a similar effect, but the samples at the center of the interval weight more. The functions in the middle row compute the differences between two adjacent intervals; again the extremes are encoded in genes. The functions in the bottom row compute the cross-correlation with a sine wave; genes encode frequency and phase of the sines. The interval where the bottom-right weight function is not zero is fixed, and it goes from 0 to 600 ms after the stimulus, i.e., it is centered around the P300. These last two functions permit to do a sort of frequency analysis of the signal.

5.4 Fitness and Selection

The fitness of a chromosome is determined by measuring the performance of a logistic classifier on the features it encodes. To have a fair estimate of the performance, a 4-fold cross-validation scheme on the training set is used, and the mean performance on the 4 folds is used as the fitness. The actual criterion used to evaluate the "performance" depends on the kind of data. For data recorded, the number of correctly predicted stimuli is used, with a little bonus for stimuli that can be correctly predicted with less than the maximum number of repetitions. Let us call l the number of correctly predicted stimuli out of a total of n, N the number of repetitions in the data set, and $r_i, i = 1...n$, the number of repetitions needed for the prediction of the letter i. The fitness$f$ is then given by:

$$f = \frac{1}{n}(l + \frac{1}{l}\sum_{i \in I} \frac{N - r_i}{N + 1}), \tag{5.5}$$

where I is the set of correctly predicted stimuli. The second term in the parentheses computes an index, averaged over the l correct letters, that grows with the decreasing of r_i; this index is always strictly less than 1, and therefore it contributes to the fitness less than a single correctly predicted letter. In this way, a higher number of correct stimuli is always preferred to a lower number of repetitions needed for correct prediction.

Repetitions are taken in their natural order, and r_i is computed in way such that if a letter is correctly predicted by using the first r_i repetitions, then it must be correctly predicted also by using the first $r_i + 1, ..., N$ repetitions.

The fitness function of our genetic algorithm can be easily changed without modifying anything else in the algorithm. This permits to adapt the fitness computation to a different BCI task, and if there are no letters to spell, other measures like accuracy, recall, or mutual information can be used.

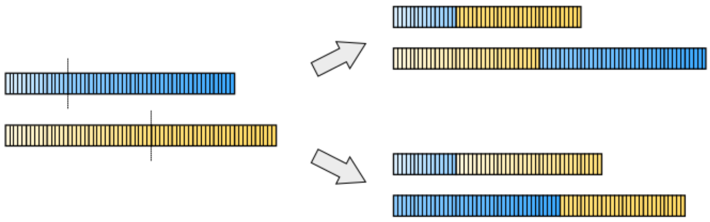


Figure 5.3: Crossover operator. The two possible way of applying it are shown.

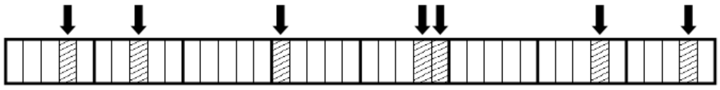


Figure 5.4: Mutation operator. Gene elements selected for mutation are marked.

In a genetic algorithm, fitness is used to select the most promising individuals for the next generation. The selection mechanism employed is *tournament* selection with *elitism*, a standard setup in genetic algorithms, with no particular adaptation. In tournament selection each individual of the new population is selected by setting up a tournament: a fixed number k of individuals are chosen at random from the old population, and the one with the highest fitness is declared the winner and will get in the next generation. Elitism is the practice of keeping the fittest individual or individuals in the new generation, even when selection discarded them (e.g., because they never participated to any tournament), or mutation and selection modified them.

5.5 Genetic Operators

After selection, the selected population undergoes crossover and mutation. These two operators have been slightly modified in order to adapt them to the non-standard chromosome structure we employed. Figure **??** shows how crossover works. Crossover is applied to pairs of chromosomes in the selected population (chosen at random) with a probability of 0.7; both chromosomes are split in two sections at a gene boundary in a random way, and then the four sections are recombined. Because the order of genes in a chromosome is not important, one section from one chromosome can be coupled with either section from the other one, and so there are two different way of doing crossover. Which way to use is randomly chosen each time, and it is important to use both ways, as this choice increases the mixing of the genetic material.

Crossover may be applied to individuals with a common ancestor, and so they may share some genes. In this case, it is very likely that at least one of the new chromosomes contains duplicated genes, and many duplicates accumulate with time. These duplicates are ignored for fitness evaluation.

Mutation (see Figure 5.4) operates on gene elements; for each element in each gene, a random choice is taken whether to mutate it, independently from each other, but with the same (small) probability, which is 0.005 in this algorithm. Elements are modified differently accordingly to their type. For a discrete element (extractor, channel, and active flag), mutation modifies it by choosing one of the other admissible values for that kind of elements, at random. For a continuous element (the two extractor parameters), a perturbation is added according to a Gaussian distribution; if the result lies outside the admissible interval [0, 1), it is wrapped around, e.g., a value of 0.95 which is perturbed by 0.07 does not result in a new value of 1.02, which is not legal, but it is wrapped to 0.02.

The use of normalized extractor parameters is useful because the way mutation works. When mutation is applied to the gene element that encodes the feature extractor, the parameters are always legal also for the resulting new feature extractor; moreover, in some cases the old and the new weight functions are similar, and this helps the GA.

5.6 Population Size and Stop Criterion

The size of the population is constant throughout a GA run. The initial population is completely random; the length, i.e., the number of genes, for each chromosome is extracted from a geometric distribution with mean 20. The actual values for gene elements are taken from uniform distributions over the whole range of legal values for each element.

The last component to complete the GA description is the stop criterion. We relied only on the number of generations, after some initial experiments where we noticed that in all runs no improvements could be seen in both the fitness of the best individual and the mean fitness of the population after 1015 generations. Figure 5.5 shows how the fitness of a population evolves in a typical GA run; it is evident that the maximum fitness reaches a plateau after only 78 generations, and population fitness tends to stabilize around the 12th generation. In any case, a check on the fitness growth is made after each run, so as to be sure that evolution has actually stopped: if the

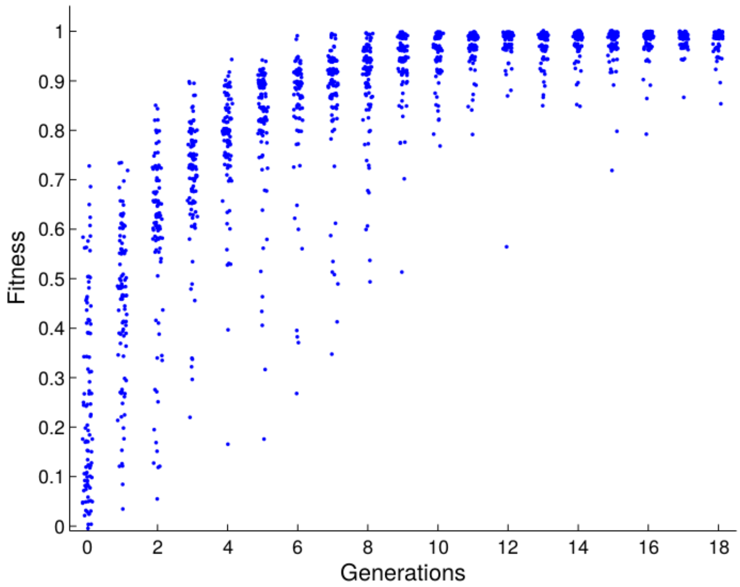


Figure 5.5: Evolution of fitness over time (generations) for the whole population in a GA run. Noise has been added to the point positions so as to make visible also overlapping points.

maximum and mean fitness has been constant for the last 34 generations, evolution is considered finished.

5.7 Feature Set Validation

After the end of each GA run, the performance of the individuals with a high fitness is validated on a test set, never used before by the GA. This validation is done on the individuals with a fitness at least 99% of the fitness of the best individual in the last generation. Evaluating more than one individual and not just the best one results in a more robust assessment of the effectiveness of the method.

For each individual, the features encoded by its chromosome are extracted from all the training data (i.e., the data used for fitness evaluation), and a logistic classifier is trained on them. The same features are extracted from the test set, and the classifier is evaluated on them. The classifier can also be used online, together with the feature extractors it was trained on. The feature extractors and the trained logistic classifier are very fast to apply, and they can be used online in real-time.

Chapter 6

Training Test and Results

"It's time for a new National Anthem. America is divided into two definite divisions. The easy thing to cop out with is sayin' black and white. You can see a black person. But now to get down to the nitty-gritty, it's getting' to be old and young - not the age, but the way of thinking. Old and new, actually... because there's so many even older people that took half their lives to reach a certain point that little kids understand now."

Jimi Hendrix

The goal of this study was to test and evaluate our portable brain-computer interface application design based on the three-stimulus oddball paradigm (see Subsection 2.4.6 on page 39) that allows a binary selection. In order to test the training session functionality and test the classification protocol (described in Chapter 5) with our Auditory P300-based BCI, data have been collected and validated offline. This chapter contains results obtained in order to show the application behaviour and the offline classification based on the tree-stimulus oddball paradigm. All results are validated relying on the goodness of our classifier, performing tests on target stimuli classification in different acoustic sessions for each subject.

Section 6.1 explains how we have designed the audio trigger detection for each stimulus in the sequence, Section 6.2 explains how data are acquired, while Section 6.3 contains all the results computing the accuracies for the target/non-target classification and the communication speed expressed in bits/min.

6.1 Threshold detection algorithm

As explained in Section 3.9 on page 74, in order to make a correct classification when doing experiments with ERPs induced by external stimuli, it is

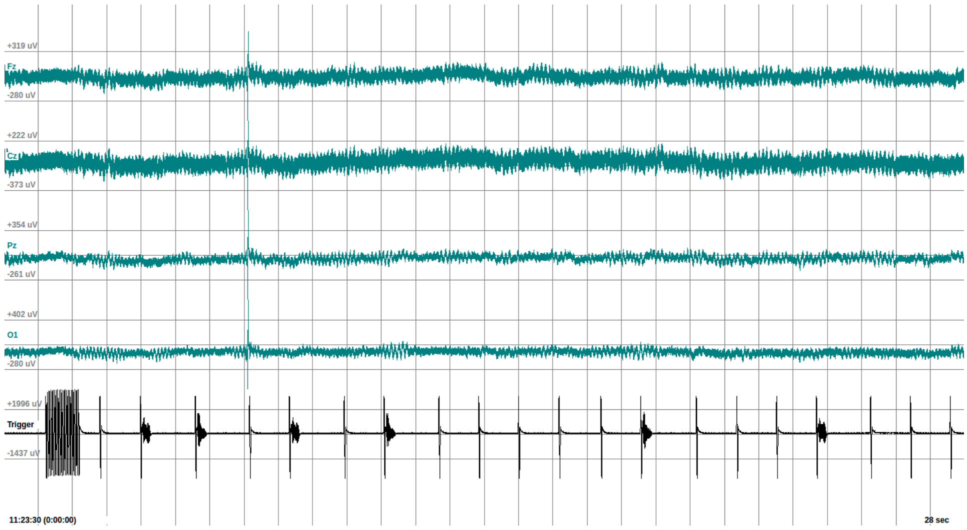


Figure 6.1: Representation of the edf file content. The EEG tracing of F_z, C_z, P_z, and O_1 channels is represented in dark green color, while the tracing of the audio trigger is represented in black color. In this example the session is composed by 20 randomized stimuli: 14 standard tones, 3 deviants and 3 targets. The ISI between each stimulus is 1000 ms and the total duration of the session is about 28 seconds.

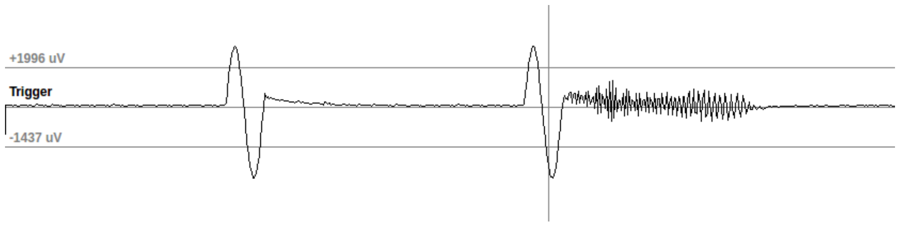


Figure 6.2: Trigger signal example for the threshold detection algorithm. The left sinusoid represents the 20 Hz added before the 1000 Hz standard tone, while the right one represents the 20 Hz added before the deviant/target word.

very important to have the EEG recording synchronized with the acoustic signal. During all the sessions, the audio trigger signal is recorded from a specific channel of the EEG amplifier (see Section 3.9 on page 74) with the EEG data.

An edf file [49] is generated automatically during the session, where the subject information and the EEG data with the audio signal are stored (see Subsection 4.12 on page 113 for details). Each acoustic stimulus is identified with a stimulation code and is said to the system when it is a target or a non target in a text file. Figure 6.1 shows an example of the edf file content generated by the application. The tracing of the EEG data is represented in dark green color while the tracing of the audio trigger is represented in black color. The edf and the text files are elaborated by the server in order to train, test and classify the target stimuli. In the code executed on server, a trick in order to recognise each acoustic trigger signal that elicits the P300 wave is actuated. Each audio file loaded in the training interface (standard tone and deviants stimuli), is composed at its beginning by a 20Hz sinusoid

$$T = \frac{1}{20Hz} = 0.05ms \tag{6.1}$$

long (see Figure 6.2). The server that process data, receive the edf files where the header (specified with the standard composition [49]) has a tag added at the end in order to recognise when the file has to be processed for the Visual BCI or for the Auditory BCI. For example we have implemented a tag defined as "20Hzfall". As we can see in Figure 6.1, in particular the tracing of the audio trigger, the application before sending the stimuli to the subject, generates a sinusoid at 20 Hz with unitary amplitude one second long called *pre-stimulus*. During this time frame (about 6 seconds) the threshold algorithm for trigger detection is computed, and the genetic algorithm is performed after this learning time. Is implemented a double moving window with length

$$L = \frac{512Hz}{20Hz} = 26samples, \tag{6.2}$$

each, where 512 Hz is the *sample-frequency* used to process the EEG data. The area of the pre-stimulus defined as a single period of a 20 Hz tone is computed as the sum of each windows length until the learning time period (26 samples) defined as *tuning threshold*. The threshold is defined as the 65% of the total area for the trigger detection:

$$TH = prctile(area_{tuning}(threshold_{tuning}), 65). \tag{6.3}$$

6.2 Data acquisition

As explained in Section 3.1 on page 64, we have developed the training interface based on the specifications given by the psychologist Mauro Marchetti [89]. We have collected several data with our BCI ON application. The EEG was recorded with Ag/Agcl electrodes (see Figure 3.4 on page 70) positioning them in the F_z, C_z, P_z, and O_1 in order to capture the P300 waves on target stimuli. Each channel was referenced and grounded to the mastoid and the impedances were kept below $5K\Omega$ (see A.3 on page 153 in the user manual for details). The EEG was recorded and amplified with the BE Light A.2.4, notch filtered at 50 Hz and sampled at 512 Hz.

To test the application training behaviour, data from three subjects [25] have been acquired. All the experiments have been performed in the AirLab of Politecnico di Milano - Como Campus.

6.2.1 Experimental design

Subjects were seated in a comfortable chair approximately less than one meter from the EEG amplifier and the electrodes were positioned over their scalp (see the User Guide in the Appendix A in Section A.3 on page 153 for details). The headphones were placed in the ear to present the auditory stimuli. Participants were cued about the current target stimulus to pay attention to, and the beginning and the end of the session was synthesized directly by the application through an English female voice. During the experiments participants have been instructed to be very relaxed and possibly to remain with their eyes closed in order to increase their concentration. To optimize the target detection it was suggested to count each target word.

The experiment consists in 20 randomized stimuli organized in sessions for each participant. In each session we present pure tones (70% of the sequence) and two words (each 15% of the sequence), considering 17 non targets stimuli (14 standard tones and 3 deviants) and 3 target words for each session. Participants have to concentrate on the designed target reported directly with the application in the in-ear headphones before the beginning of each session, and discard the other deviants and the standard tones. We have considered a standard pure tone at 1000 Hz with 80 ms duration, and the two deviants words *yes* and *no* each 500 ms long. All stimuli were delivered at 75 dB in stereo (both ears). The target designed for selection was alternated between word yes and no from one session to another (e.g., sessions 1, 3, 5, 7, 9, 11 selection of word yes; sessions 2, 4, 6, 8, 10, 12 selection of word no). Sessions were designed to elicits both the P3a (elicited by deviant word) and P3b (classic P300 elicited by target word) components, accustoming the subject to the same timing delivering (ISI with a fixed time of 1000 Hz between each stimulus). This is done in order to test the correct classification of the target word with a confusion matrix. Before each session, each sound is tested to the participant in order to verify that the he could differentiate the acoustic stimuli.

Table 6.1: Confusion matrix.

		Prediction outcome		
		p	**n**	**total**
actual value	**p′**	True Positive	False Negative	P′
	n′	False Positive	True Negative	N′
	total	P	N	

6.3 Auditory BCI performance

This section contains the results obtained with data collected during the experiments. Accuracy and communication speed are obtained for each participant and then averaged to get the performance of our auditory BCI system. In order do this we have computed for each subject a *confusion matrix*

A confusion matrix, also known as a contingency table or an error matrix [137] is a specific table layout that allows visualization of the performance of an algorithm. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. A confusion matrix is represented as in Table 6.1 and its elements are defined as:

- *true positive (TP)*: is the number of correct predictions that an instance is positive (i.e., a target)
- *true negative (TN)*: is the number of correct predictions that an instance is negative (i.e., a non target)
- *false positive (FP)*: is the number of incorrect predictions that an instance is positive (i.e., a target)
- *false negative (FN)*: is the number of incorrect of predictions that an instance negative (i.e., a non target)

From the confusion matrix it is possible to define the *Accuracy (AC)* (used for our tests) as the proportion of the total number of predictions that were correct:

$$AC = \frac{TP + TN}{TP + TN + FP + FN} \quad (6.4)$$

The communication speed was defined as the number of bits transmitted per run [107]:

$$B = log_2 N + P log_2 P + (1 - P) log_2 (\frac{1 - P}{N - 1}) \quad (6.5)$$

with N being the number of possible targets (in our case is 2) and P being the probability of the correct classification (defined as the *accuracy*). To compute the ITR (Information Transfer Rate in minutes) we have to consider the total number of stimuli for each trial (in our case 20) multiplied by the averaged interval timing between stimuli. This value have to be multiplied by the total number of trials and then converted to minutes by division of 60 obtaining the time t in minutes needed for a selection with the given number of trials. Then the bitrate given in the Equation 6.5 is divided by t in order to obtain the ITR.

6.3.1 Results

To compute the probability of the correct classification for our BCI system, we have considered a test set of 10% of the training set for each subject. The first experimental test of our BCI system was performed on three students and their training sets considered different but homogeneous. In Table 6.2 the training set for each subject considered is shown.

Table 6.2: Training set considered for each subject. The total number of stimuli, the target and non target are defined for the training set.

Subject	Training set	Total stimuli	Targets	Non Targets
1	12	240	36	204
2	10	200	30	170
3	4	80	12	68

The training set for each subject is composed by a trial with 20 standard tones, 3 deviants and 3 targets per session. The system for the first subject should classify 36 target words on a total of 240 stimuli; for the second subject should classify 30 targets on a total of 200 stimuli; for the last subject should classify 12 targets in a total of 80 stimuli. The elements of the confusion matrix for each subject computed by the genetic algorithm is defined in Table 6.3.

Table 6.3: Elements of confusion matrices for each subject. True Positives (TP), False Positives (FP), True Negatives (TN), False Negatives (FN).

Subject	TP	FP	TN	FN	Accuracy [%]	Bitrate [Bits/min]
1	36	0	204	0	100	0.25
2	30	0	170	0	100	0.30
3	8	4	68	0	95	2.88

In Table **??** it is possible to see that for the subjects 1, the genetic algorithm has classified 36 stimuli over 36 target and 204 stimuli over 204 non target. The accuracy obtained is 100% and the bitrate is 0.25 bits/min reaching the perfect classification. The same result is obtained for the second subject which has classified 30 stimuli over 30 target and 170 stimuli over 170 non target. The accuracy is 100% and his bitrate is 0.30 bits/min. The last subject has classified 8 stimuli over 12 targets while the true negatives are 68 over 68 non targets reaching an accuracy of 95% and a bitrate of 2.88 bits/min. Accuracies and Bitrates for all subjects are then averaged and reported in Table 6.4.

Table 6.4: Offline classification test results. The bitrate expressed in Bits/min and the accuracy are computed for each subject and then averaged.

Subjects	Accuracy [%]	Bitrate [Bits/min]
1	100	0.25
2	100	0.30
3	95	2.88
Mean	**98.33**	**1.14**

The averaged accuracy between the three subjects is 98.33% and the bitrate is 1.14 bits/min. With this results we can state that for our first testing of the application we have reached an high accuracy and a competitive communication speed as the prerequisite of the thesis work.

Chapter 7

Conclusion

"When I'm sad, she comes to me
With a thousand smiles, she gives to me free
It's alright she says it's alright
Take anything you want from me, anything
Anything."

Jimi Hendrix - "Little Wing"

For patients with impaired vision at the final stage of the disease, it is important to have a BCI paradigm that does not require visual feedback or stimulation. Additionally, an easily attained high level of accuracy is particularly important when working with ALS patients because the average accuracy in any kind of BCI tends to be lower than for healthy subjects.

This thesis has presented methods, validated by experiments, in order to develop a portable auditory brain-computer interface based on P300 to allow Yes-No communication to Amyotrophic Lateral Sclerosis patients. This type of BCI allows to deliver an acoustic randomized sequence composed by two deviant words and standard tones using the three-stimulus oddball paradigm. The training interface, has been realized from the input of Mauro Marchetti, thinking to ease its use by doctors or psychologists that will use the application for their tests.

The first work done in this thesis was to build an oddball paradigm generating a randomized sequence of stimuli with a frequent sound and two deviants. The interface developed is composed by eleven channels that allow to loading wave files such as pure tones, words and complex tones. For each channel the user can define the directionality (stereo, mono right or mono left), the Loudness perceived from the participant (75 dB, 65 dB and 55 dB) with the clinical in-ear headphones developed, and the probability of occurrence (70% for standard tone and 15% for deviants). The user can also set the total number of stimuli of the sequence, and the pattern of stimuli presentation between SOA and ISI with their advanced settings (i.e., a fixed interval, five intervals chosen randomly or random intervals choosing the minimum and the maximum values). The user is forced to insert the participant credentials and the number of the current session, with the possibility to save and load it with all the parameters defined from SD card.

The second step of the work was based on the implementation of the Auditory BCI with the Visual BCI speller adapting the drivers that allow the communication between devices. Drivers are implemented in order to enable the 10-20 system with the possibility to activate the desired electrodes on the tablet screen, visualize only the behaviour of the signals associated to the channels selected and finally the training session is performed by processing only the channels enabled. The training files generation for each participant ID subdivided in session, and a tag (Auditory or Visual) which distinguishes the type of the BCI is added to the file name.

The clinical in-ear headphones developed for the experiments are designed to avoid problems with the correct EEG estimation, in order to maintain a suitable distance from the electrodes positioned over the subjects adding a PVC tube 1 meter long between the in-ear amplifiers and the silicone ear plugs. An audio signal adapter for the EEG synchronization with the acoustic stimuli, has been developed to adapt the dynamic audio signal output of the tablet into the EEG amplifier.

Data acquired from subjects have been used obtaining promising results for the first preliminary tests of the audio protocol, our methods for filtering data, extract the features and classify the target words. We have demonstrated that the oddball BCI paradigm that we have built in the Auditory BCI application, is able to achieve high accuracy of 98.33% and competitive bitrates of 1.14 bits/min.

7.1 Future work

The application interface of the Auditory BCI could to be improved with the saving of the electrodes selection from the user for the current participant. May be a possibility to select/deselect all electrodes with a single button would be useful to improve the application usability.

The application drivers have to be optimized in order to maintain a stable connection between the tablet and the EEG amplifier when the the EEG data are requested in reading/writing from the circular buffer of the

BE Light. Thus avoiding the flashing of drivers during the usage of the application. Having implemented the electrodes selection of the 10-20 system in the Auditory BCI using and the dynamic data processing to elaborate the EEG data, we have to adapt also this feature back to the original Visual BCI speller.

Another step to do with our application is to implement the online phase for the ERP classification in real time with a feedback presentation (e.g., the classified yes or no words presented acoustically to participants, and the corresponding word displayed on the monitor to the user). The most important step will be to examine the use of the application with individuals diagnosed with amyotrophic lateral sclerosis (ALS).

A next step that we want to do is to adapt our application with the Emotiv EPOC / EPOC+, that is a revolutionary Brain Computer Interface with a wireless EEG system that offering high resolution, implementing both the Visual and the Auditory BCIs.

Bibliography

[1] E. D. Adrian and Matthews. B. h. c. *Brain.*, 57.:355., 1934.

[2] Akust. Z. After v.Békésy. *Physics Today.*, volume 8. 1943.

[3] Hirotugu Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control.*, page 716723., 1974.

[4] Horst Bischof Alois Schlögl, Felix Lee and Gert Pfurtscheller. Characterization of four-class motor imagery eeg data for the bci-competition 2005. *Journal of Neural Engineering.*, 2.(4.):L14., 2005.

[5] Horn AK Leigh RJ Buttner-Ennerver JA. Averbuch-Heller L, Helmchen C. Slow vertical saccades in motor neuron disease: correlation of structure and function. *Ann Neurol.*, (4.):641–648, 1998.

[6] van der Molen M. Bashore T. Discovery of p300: a tribute. *Biol Psychol.*, 32.:155–171., 1991.

[7] Sejnowski TJ. Bell AJ. An information-maximization approach to blind separation and blind deconvolution. *Neural Comput.*, 7.:1129–1159., 1995.

[8] Christopher M. Bishop. *Neural Networks for Pattern Recognition.* New York, NY, USA., 1995.

[9] David T. Blackstok. *Foundamentals of physical acoustics.* University of Texas. Austin, Texas.

[10] Curio G. & Müller K.R. Blankertz, B. Classifying single trial eeg: Towards brain computer interfacing. Advances in neural information processing systems 14, Cambridge, MA, 2002, MIT Press, 2002.

[11] Valderrama M. Crpon B. Navarro V. Le Van Quyen M. Botella-Soler, V. Large-scale cortical dynamics of sleep slow waves. *PloS one*, 7.(2.), 2012.

[12] A. Brancucci. Neural correlates of cognitive ability. *Neurosci. Res.*, 90.:1299–1309., 2013.

[13] Eric R. Braverman. Brain mapping: A short guide to interpretation, philosophy and future. *Journal of Orthomolecular Medicine.*, 5.(4.):189–197., 1990.

[14] Ulrich R. F. Spiker D. G. Sclabassi R. J. Reynolds III C. F. Marin R. S. Boller F. Brenner, R. P. Computerized eeg spectral analysis in elderly normal, demented and depressed subjects. *Electroencephalography and clinical neurophysiology.*, 64.(6.):483–492., 1986.

[15] J.D. Bronzino. *Principles of Electroencephalography.*, pages 201–212. CRC Press, Florida, 1995.

[16] Bonnet M. Burle, B. High-speed memory scanning: a behavioral argument for a serial oscillatory model. *Cognitive Brain Research.*, 9.(3.):327–337., 2000.

[17] Shagass C. Evoked brain potentials in psychiatry. *Vaughan.*, 178.(4057.):150–151., 1972.

[18] J. Polich. C. J. Gonsalvez. P300 amplitude is determined by target-to-target interval. *Psychophysiology.*, 39.:388–396., 2002.

[19] Polich J. Cahn BR. Meditation states and traits: Eeg, erp, and neuroimaging studies. *Psychol Bull.*, 132.(2.):180–211., 2006.

[20] Nicola Orio. Carlo Drioli. *Elementi di Acustica e Psicoacustica.*, volume 2. 1999.

[21] Polich J. Cass M. P300 from a single-stimulus paradigm: auditory intensity and tone frequency effects. *Biol Psychol.*, 46.(1.):51–65., 1997.

[22] T. Collura. *A Guide to Electrode Selection, Location, and Application for EEG Biofeedback.* Ohio., 1998.

[23] Hao L. Hart J. Boatman D. Lesser R. P. Irizarry R. Gordon B. Crone, N. E. Electrocorticographic gamma activity during word production in spoken and sign language. *Neurology.*, 57.(11.):2045–2053., 2001.

[24] K. B. Campbell D. H. Loewy and C. Bastien. The mismatch negativity to frequency deviant stimuli during natural sleep. *Electroencephalogr. Clin. Neurophysiol.*, 98.:493–501., 1996.

[25] Matteo Cartiglia. Davide Schiena, Nicolas Alan Casella. Work for the bachelor thesis in biomedical engineering. they have contributed developing the clinics in-ear headphones and the audio signal adapter circuit for the be light input, and writing the user manual.

[26] P. A. Davis. The electrical response of the human brain to auditory stimuli. *Am. J. Physiol.*, 126.:475476., 1939.

[27] Android ActionBar description:. `http://developer.android.com/design/patterns/actionbar.html`.

[28] Android Activity description:. `http://developer.android.com/guide/components/activities.html`.

[29] Android Fragment description:. `http://developer.android.com/guide/components/fragments.html`.

[30] Android FragmentStatePagerAdapter description:. `http://developer.android.com/reference/android/support/v4/app/FragmentStatePagerAdapter.html`.

[31] Android MediaPlayer description:. `http://developer.android.com/reference/android/media/MediaPlayer.html`.

[32] Android NavigationDrawer description:. `https://developer.android.com/design/patterns/navigation-drawer.html`.

[33] Android SharedPreferences description:. `http://developer.android.com/reference/android/content/SharedPreferences.html`.

[34] Android SoundPool description:. `http://developer.android.com/reference/android/media/SoundPool.html`.

[35] Android SurfaceView description:. `http://developer.android.com/reference/android/view/SurfaceView.html`.

[36] Android ViewPager description:. `http://developer.android.com/reference/android/support/v4/view/ViewPager.html`.

[37] Little R. Donald MW. The analysis of stimulus probability inside ad outside the focus of attention, as reflected by the auditory n1 and p3 components. *Can J Psicol.*, 35.:175–187., 1981.

[38] W.; McCallum C. Donchin, E.; Ritter. Cognitive psychophysiology: the endogenous components of the erp. In P.; Koslow S. Callaway, P.; Tueting, editor, *Brain-event related potentials in man.*, pages 349–411. New York: Academic Press, 1978.

[39] M. Snodgrass. E. Bernat, H. Shevrin. Subliminal visual oddball stimuli evoke a p300 component. *Clinical Neurophysiology.*, 112.:159–171., 2001.

[40] T. R. Bashore M. G. H. Coles E. Donchin, D. Karis and G. Gratton. Cognitive psychophysiology and human information processing. In E. Donchin M. G. H. Coles and S. W. Porges., editors, *Psychophysiology: Systems, Processes, and Applications.* The Guilford Press, New York., 1986.

[41] Gruzelier J. H. Egner, T. Eeg biofeedback of low beta band components: frequency-specific effects on variables of attention and event-related brain potentials. *Clinical Neurophysiology.*, 115.(1.):131–139., 2004.

[42] Meagan A. Jenkins et al. *The Journal of Neuroscience.*, 30.(15.):5125–5135., 2010.

[43] S. S. Stevens et al. *Handbook of Experimental Psychology.* John Wiley & Sons, Inc., New York., 1951.

[44] T. H. Schafer et al. Frequence selectivity of the ear as determined by masking experiments. *J. Acoustic Soc. Amer.*, page 490., 1950.

[45] T.W. Picton et. al. Guidelines for using event-related potentials to study cognition: Recording standards and publication criteria. *Psychophysiology.*, 37.:127–152, 2000.

[46] M. Matteucci F. Salice. F. Veronese, H. Saidinejad. Communication oriented brain computer interface in a remote monitoring system for amyotrophic lateral sclerosis. *Assistive Technology: From Research to Practice: AAATE.*, pages 483–488., 2013.

[47] Donchin. Farwell L. Talking off the top of your head: toward a mental prosthesis utilizing even-related brain potentials. *Electroencephalogr Clin Neurophysiol.*, 70, 1988.

[48] K. Fazel-Rezai, R.; Abhari. A region-based p300 speller for brain-computer interface. *Can. J. Elect. Comput.*, 11.:81–85., 2009.

[49] EDF file specification:. `http://www.edfplus.info/specs/edf.html`.

[50] Fletcher and Munson. Equal-loudness contours for pure tones. *American standard for noise measurement.*, 1942.

[51] H. Fletcher. Auditory patterns. *Revers. of Mod. Phys.*, 12.:47–65., 1940.

[52] Society for Neuroscience. *BrainFacts - A primer on the brain and nervous system.* 1121 14th Street, NW, Suite 1010, Washington, DC 20005 USA., 2012. www.sfn.org.

[53] Krusienski D Bross D Nijboer F Birbaumer N et al. Furdea A, Halder S. An auditory oddball (p300) spelling system for brain-computer interfaces. *Psychophysiology.*, 46.(3.):617–625, 2009.

[54] Davis H. & Lennox W.G. Gibbs, F. A. The electro-enchephalogram in epilepsy and in conditions of impaired consciousness. *Arch. Neurol. and Psychiat.*, 34.:1133–1148., 1935.

[55] Patrick GJ. Improved neuronal regulation in adhd: An application of 15 sessions of photic-driven eeg neurotherapy. *J Neurother.*, 1.(4.).

[56] T. Ebrahimi G.N. Garcia and J. Vesin. Support vector eeg classification in the fourier and time-frequency correlation domains. In *Neural Engineering, 2003.*, pages 591–594. Conference Proceedings. First International IEEE EMBS Conference on, 2003.

[57] Berger H. Über das elektroenkephalogram des menschen. *Arch. f. Psychiat.*, 87:527–570, 1929.

[58] Pashler H. Dual-task interference in simple tasks: data and theory. *Psychol Bull.*, 116.:220–244., 1994.

[59] H. J. Michalewski N. Bleich H. Pratt, A. Starr and N. Mittelman. The auditory p50 component to onset and offset of sound. *Clin. Neurophysiol.*, 119.:376–387., 2008.

[60] Ossebaard HC. Stress reduction by technology? an experimental study into the effects of brainmachines on burnout and state anxiety. *Appl Psychophysiol Biofeedback.*, 25.(2.):93–101., 2000.

[61] Lal T Bierig K Birbaumer N Schölkopf B. Hill, N. An auditory paradigm for brain-computer interface. 2004.

[62] S. A. Hillyard and Electrophysiology of cognition. T. E. Picton. Mismatch negativity related to the perception of whispered speech. In F. Plum., editor, *in Handbook of Physiology.* William and Wilkins, Baltimore., 1987.

[63] I. J. Hirish. *The Measurement of Hearing.*, volume McGraw Hill Book Company, Inc. New York., 1952.

[64] John H. Holland. *Adaptation in Neural and Artificial Systems.*, volume University of Michigan Press. 1975.

[65] 2nd Wycoff SJ. Howard CE, Graham LE. A comparison of methods for reducing stress among dental students. *J Dent Educ.*, 50.(9.):542–544., 1986.

[66] W. Huang, A. & Zhou. Blda approach for classifying p300 potential. *7th Asian-Pacific Conference on Medical and Biological Engineering, Springer.*, page 341., 2008.

[67] M. Hyde. The n100 response and its applications. *Audiol. Neurotol.*, 2.:281–307., 1997.

[68] Intendix. `http://www.intendix.com/`. [Online, accessed on 4 April 2015].

[69] Polich J. Updating p300: an integrative theory of p3a and p3b. *Clin Neurophysiol 2007.*, 118.(10.):2128–2148., 2007.

[70] H. H. Jasper and L. Carmichael. Electrical potentials from the intact human brain. *Science.*, 81.(2089.):51–53., 11 January 1935.

[71] H.H. Jasper. The ten-twenty electrode system of the international federation. *Electroencephalography and Clinical Neurophysiology.*, pages 371–375., 1958.

[72] Williams JH. Frequency specific effects of flicker on recognition memory. *Neuroscience.*, 104.(2.):283–286., 2001.

[73] H.W. Hake. J.P. Hega. On the masking pattern of a simple auditory stimulus. *J. Acoustic Soc. Amer.;*, 22.:622., 1950.

[74] Eliahu I. Jury. *Theory and application of the Z-transform method.* Wiley., 1964.

[75] D. A. Kaiser. Interest in films as measured by subjective & behavioral ratings and topographic eeg., 1994. Methodological Issues at: http://www.skiltopo.com/papers/applied/articles/dakdiss2.htm.

[76] Polich J. Katayama J. P300 from one-, two-, and three-stimulus auditory paradigms. *Int J Psychophysiol.*, 1998.

[77] Polich J. Katayama J. Stimulus context determines p3a and p3b. *Psychophysiology.*, 35.:23–33., 1998.

[78] Polich J. Katayama J. Auditory and visual p300 topography from a 3 stimulus paradigm. *Clin Neurophysiol.*, 110.:463–468., 1999.

[79] Brunner P Schwartz NE Wolpaw JR Neuper C et al. Klobassa DS, Vaughan TM. Toward a high-throughput auditory p300-based brain-computer interface. *Clin Neurophysiol.*, 120.(7):1252–1261, 2009.

[80] Sellers E. Cabesing F. Bayoudh S. McFarland D. Vaughan T. & Wolpaw J. Krusienski, D. A comparaison of classification techniques for the p300 speller. *Journal of Neural Engineering.*, 6.:229–305, 2006.

[81] Furdea A Hösle A. Kübler A, Halder S. Brain painting - bci meets art. *Proceedings of the 4th international brain-computer interface workshop and training course.*, pages 361–366., 2008.

[82] Halder S Hammer EM Nijboer F Kotchoubey B. Kbler A, Furdea A. A brain-computer interface controlled auditory event-related potential (p300) spelling system for locked-in patients. *Ann N Y Acad Sci.*, 1157.:90–100, 2009.

[83] S. le Cessie and J. C. van Houwelingen. Ridge estimators in logistic regression. *Applied Statistics.*, 41.(1.):191–201., 1992.

[84] Anna Merzagora Alberto Gaggiani Sergio Cerutti Lorenzo Rossi, Anna Maria Bianchi and Francesco Bracchi. Single trial somatosensory evoked potential extraction with arx filtering for a combined spinal cord intraoperative neuromonitoring technique. *BioMedical Engineering OnLine.*, 6., 2007.

[85] Congedo M. L ecuyer a. Lamarche F. & Arnaldi B. Lotte, F. A review of classification algorithms for eeg-based brain-computer interfaces. *Journal of neural engineering.*, 4.(2.):R1–R13., June, 2007.

[86] Kari Nieminen Ilkka Korhonen Anna Maria Bianchi Linda Pattini Jukka Takala Jari Karhu Luca T. Mainardi, Jukka Kupila and Sergio Cerutti. Single sweep analysis of event related auditory potentials for the monitoring of sedation in cardiac surgery patients. *Computer Methods and Programs in Biomedicine.*, 63.(3):219–227., 2000.

[87] Steven J. Luck. An introduction to the event-related potential technique. *The MIT Press.* ISBN 0-262-12277-4.

[88] K. Alho M. Sams, P. Paavilainen and R. Näätänen. Auditory frequency discrimination and event-related potentials. *Electroencephalogr. Clin. Neurophysiol.*, 62.:437–448., 1985.

[89] Mauro Marchetti. *Covert orienting of visuospatial attention in a Brain-Computer Interface for communication.* PhD thesis, Universitá degli Studi di Padova - Dipartimento di Psicologia dello sviluppo e della Socializzazione., 2012.

[90] Piccione F. Silvoni S. Marchetti, M. and K. Priftis. Exogenous and endogenous orienting of visuospatial attention in p300-guided brain-computer interfaces: A pilot study on healthy participants. *Clinical Neurophysiology.*, 123.:773–779., 2012.

[91] Dal Seno B. Matteucci M., Mainardi L. A genetic algorithm for automatic feature extraction, in p300 detection. *Conference: Proceedings of the International Joint Conference on Neural Networks, IJCNN 2008, part of the IEEE World Congress on Computational Intelligence.*, June 1-6, 2008.

[92] David SV Wolpaw JR. McFarland DJ, McCane LM. Spatial filter selection for eeg-based communication. *Electroencephalogr Clin Neurophysiol.*, 103.:386–394., 1997.

[93] Braun C. Arnold M. Witte H. Taub E. Miltner, W. H. Coherence of gamma-band eeg activity as a basis for associative learning. *Nature.*, 397.(6718.):434–436., 1999.

[94] M.M. Moore. Real-world applications for brain-computer interface technology. *IEEE Trans.Neural Syst. Rehabil. Eng.*, 11.:162–165., 2003.

[95] Thais C. Morata. Young people: Their noise and music exposures and the risk of hearing loss. *International Journal of Audiology*, 2007.

[96] Magoun H. W. Moruzzi, G. Brain stem reticular formation and activation of the eeg. *Electroencephalography and Clinical Neurophysiology.*, 1.:455–473., 1949.

[97] Halder S Rosenstiel W Bogdan M Birbaumer N et al. Mugler E, Bensch M. Brain-computer interface control of an internet browser. *Int J Bioelectromagnet.*, 10.(1):56–63., 2008.

[98] Weihing J. Musiek F, Froke R. The auditory p300 at or near threshold. *J Am Acad Audiol.*, 16.:699–708., 2005.

[99] Birbaumer N. Brain-computer interface research: coming of age. *Clin Neurophysiol.*, 117.(3.):479–483., 2006.

[100] Yu H et al. Nomura T, Higuchi K. Slow-wave photic stimulation relieves patient discomfort during esophagogastroduodenoscopy. *Gastroenterol Hepatol.*, 21.(1 Pt 1.):54–58., 2006.

[101] Effects of Electrode Placement. `http://www.focused-technology.com/electrod.htm`.

[102] G. M. McArthur P. R. Hill and D. V. Bishop. Phonological categorization of vowels: A mismatch negativity study. *Electroencephalogr. Clin. Neurophysiol.*, 15.:21952199., 2004.

[103] Geoffrey. Pearce, John M. Hall. A model for pavlovian learning: Variations in the effectiveness of conditioned but not of unconditioned stimuli. *Psychological Review.*, pages 532–552., 1980.

[104] Roth WT Kopell BS. Pfefferbaum A, Ford JM. Age related changes in auditory event-related potentials. *Electroenceph Clin Neurophysiol.*, 49.:266–276., 1980.

[105] Wenegrat BG Roth WT Kopell BS. Pfefferbaum A, Ford JM. Clinical application of the p3 component of event-related potentials in noraml aging. *Electroenceph Clin Neurophysiol.*, 59.:85–103., 1984.

[106] T. W. Picton and S. A. Hillyard. Human auditory evoked potentials. ii: Effects of attention. *Electroencephalogr. Clin. Neurophysiol.*, 36.:191–199., 1974.

[107] J.R. Pierce. *An Introduction to Information Theory.* Dover, New York., 1980.

[108] Criado JR. Polich J. Neuropsychology and neuropharmacology of p3a and p3b. *Int J Psychophysiol.*, 60.:172–185., 2006.

[109] Johnson R. On the neural generators of the p300 component of the event-related potential. *Psychophysiology.*, 30.:90–97., 1993.

[110] A. W. K. Gaillard R. Näätänen and S. Mäntysalo. Early selective-attention effect on evoked potential reinterpreted. *Proc. Aud. Res. Meet. Acoust. Soc. Jpn.*, 43.:31–35., 2013.

[111] T. Rinne R. Näätänen, P. Paavilainen and K. Alho. The mismatch negativity (mmn) in basic research of central auditory processing: A review. *Clin. Neurophysiol.*, 118.:2544–2590., 2007.

[112] Guan C Zhang H Teo C Zeng Q Ang M Laugier C. Rebsamen B, Burdet E. A brain-controlled wheelchair based on p300 and path guidance. *In: Proceedings of IEEE/RAS-EMBS international conference on biomedical robotics and biomechatronics.*, 2006.

[113] Ingo Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.*, volume Fromman-Holzboog. 1973.

[114] Ingo Rechenberg. *Practical Genetic Algorithms.*, volume Wiley. 2004.

[115] J. G. Roederer. *The Physics and Psychophysics of Music.*, volume Springer Verlag. 1980.

[116] V. L. Schwent S. A. Hillyard, R. F. Hink and T. W. Picton. Electrical signs of selective attention in the human brain. *Science.*, 182.:177–180., 1973.

[117] R. Andreoni F. Nijboer E.M. Hammer S.C. Kleih N. Birbaumer A. Kübler. S. Halder, M. Rea. An auditory oddball brain-computer interface for binary choices. *Clinical Neurophysiology.*, pages 516–523., 2010.

[118] M. Bleichner S. Paulmann and S. A. Kotz. Valence, arousal, and task effects in emotional prosody processing. *Psychol.*, 4.:345., 2013.

[119] Cutcomb S. D. Crawford H. J. Pribram K. Sabourin, M. E. Eeg correlates of hypnotic susceptibility and hypnotic trance: spectral analysis and coherence. *International Journal of Psychophysiology.*, 10.(2.).

[120] Donchin E. Sellers E. A p300-based braincomputer interface: initial tests by als patients. *Clin Neurophysiol.*, 117.(3.):538–548., 2006.

[121] B. Dal Seno. *Toward An Integrated P300- And ErrP-Based Brain-Computer Interface.* PhD thesis, Politecnico di Milano - Dipartimento di Elettronica e Informazione, 2009.

[122] Yom-Tov E. & Inbar G. F. Serby, H. An improved p300-based brain-computer interface. *IEEE Trans. Neur. Syst. Rehabil. Eng.*, 13.:89–98., 2005.

[123] D. Siever. The application of audio-visual entrainment for the treatment of seasonal affective disorder. *Biofeedback.*, 32.(3.):32–35., 2004.

[124] Volpato-C. Cavinato M. Marchetti M. Priftis K. Merico A. Tonin P. Koutsikos K. Beverina F. Silvoni, S. and F. Piccione. P300-based brain-computer interface communication: evaluation and follow-up in amyotrophic lateral sclerosis. *Frontiers in Neuroscience.*, 3.(60.), 2009.

[125] Android Studio site:. `https://developer.android.com/sdk/index.html`.

[126] EBNeuro site:. `http://www.ebneuro.biz`.

[127] Info Solution site:. `http://www.infosolution.it`.

[128] IntelliJ IDEA site:. `https://www.jetbrains.com/idea/`.

[129] Marchetti site:. `http://htlab.psy.unipd.it/index.php?page=mauro-marchetti`.

[130] Matlab site:. `ftp://ftp.gnu.org/gnu/octave/`.

[131] Ubuntu Studio site:. `http://ubuntustudio.org/`.

[132] Delargy M. Smith E. Locked-in syndrome. *BMJ.*, 330.(7488.):406–409., 2005.

[133] Hillyard S. Snyder E. Long-latency evoked potentials to irrelevant, deviant stimuli. *Behav Biol.*, 16.:319–331., 1976.

[134] Donchin E. Spencer KM, Dien J. Spatiotemporal analysis of the late erp responses to deviant stimuli. *Psychophysiology.*, 38.(2.):343–358., 2001.

[135] Wickens C Donchin E. Squires K, Petuchowski S. Effects of stimulus sequence on event related potentials: comparison of visual and auditory sequences. *Percept Psychophys.*, 22.:31–40., 1977.

[136] Hillyard S. Squires N, Squires K. Two varieties of long-latency positive waves evoked by unpredictable auditory stimuli in man. *Electroencephalogr Clin Neurophysiol.*, 38.:387–401., 1975.

[137] Stephen V. Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote Sensing of Environment.*, 62.(1.):77–89., 1997.

[138] Edwin B. Stevens, Stanley Smith; Volkmann; John; & Newman. A scale for the measurement of the psychological magnitude pitch. *ournal of the Acoustical Society of America*, 8(3):185–190, 1937.

[139] Sadler R. M. Young G. B. Sundaram, M. and N. Pillay. Electroencephalography in epilepsy: current perspectives. *Canadian Journal of Neurological Sciences.*, 26.(4.):255–262., 2012.

[140] Braren M. Zubin J. John E.R. Sutton, S. Evoked-potential correlates of stimulus uncertainty. *Science.*, 150.:1187–1188., 1965.

[141] S. Sutton. P300-thirteen years later. In H. Begleiter, editor, *Evoked brain potentials and behavior.*, pages 107–125. New York: Plenum Press., 1979.

[142] K. Ueda T. Kikuchi, G. B. Remijn and Y. Nakajima. Mismatch negativity related to the perception of whispered speech. *Proc. Aud. Res. Meet. Acoust. Soc. Jpn.*, 43.:31–35., 2013.

[143] Kambara H. Koike Y. TAhi, S.T. A dictionary-driven p300 speller with a modified interface. *IEEE Trans. Neural Syst. Rehabil. Eng.*, 19.:6–14, 2011.

[144] LaPallo B.K. Boulay C.B. Krusienski D.J. Frye G.E. Hauser C.K. Schwartz N.E. Vaughan T.M. Wolpaw J.R. Townsend, G. A novel p300-based brain-computer interface stimulus presentation paradigm: Moving beyond rows and columns. *Clin. Neurophysiol.*, 121.:1109–1120., 2010.

[145] V. Vapnik. *The nature of statistical learning theory.* Springer Verlag., New York., 1995.

[146] Heberlein I Hagenah J Verleger R. Vieregge P, Wauschkuhn B. Selective attention is impaired in amyotrophic lateral sclerosis a study of event-related eeg potentials. *Brain Res Cogn Brain Res.*, 8.(1.):27–35., 1999.

[147] K. N. Stevens W. A. Rosenblith, the staff of Bolt Baranek, and Newman. *Handbook of Acoustic Noise Control.*, volume WADC Tech. Repl.., pages 52–104. Order from Office of Technical Services, Department of Commerce, Washington, D.C., 11, noise and man. edition, June 1953.

[148] J. Lavikainen K. Reinikainen K. W. Ritter, P. Paavilainen. Event-related potentials to repetition and change of auditory stimuli. *Electroencephalogr. Clin. Neurophysiol.*, 83.:306–321., 1992.

[149] S. A. Shappell W. S. Pritchard and M. E. Brandt. Psycho-physiology of n200/n400: A review and classification scheme. *Adv. Psychophysiol.*, 4.:43–106., 1991.

[150] Cooper R. Aldridge V.J. McCallum W.C. Winter A.L. Walter, W.G. Contingent negative variation: an electric sign of sensorimotor association and expectancy in the human brain. *Nature.*, 203.(4943.):380–384., 1939.

[151] I. E. Wickramasekera I. On attempts to modify hypnotic susceptibility: Some psychophysiological procedures and promising directions. *Annals of the New York Academy of Sciences.*, 296.:143–153., 1977.

[152] After Wiener. *Physics Today.*, volume 4. December, 1953.

[153] Ramaswamy D. Williams, J. and A. Oulhaj. 10 hz flicker improves recognition memory in older people. *BMC Neurosci.*, 7.:21., 2006.

[154] Wolpaw EW. Wolpaw JR. Brain-computer interfaces: something new under the sun. In Wolpaw EW. Wolpaw JR, editor, *Brain-computer interfaces: principles and practice.* Oxford University Press, Oxford., 2012.

[155] Pritchard WS. Psychophysiology of p300. *Psychol Bull.*, 89.:506–540., 1981.

[156] Zhao H. Tang. Z. Zhuang, T. A study of brainwave entrainment based on eeg brain dynamics. *Computer and Information Science.*, 2.(2.).

Appendix A

User manual

v1.0 - 8^{th} July 2015

A neural interface, better known as BCI ("Brain-Computer Interface"), is a direct communication pathway between the brain and an external device. BCIs are often directed at assisting, augmenting, or repairing human cognitive or sensory-motor functions. A BCI is a system able to redefine signals generated by the user's brain in a defined command, bypassing the work of muscles and nerves.

With the ON application, the equipment is able to recognize a brain wave that occurs when a subject focusing his spatial attention on an acoustic target told with a voice synthesizer before each session. It follows that the subject should focus only on the choice he did, for example, counting the number of the target stimulus heard. For a correct operation it is necessary that the user is concentrated and relaxed. Then a training is necessary to get a classification of the binary choice.

A.1 How to install the application

To install the application, download the ON application package in the SD memory card, and then open it. It will launch the Android Play Store; follow the installation procedure and then agree to the permission request. For submitting training records and receiving data is recommended to install Dropbox, an application available for free.

A.2 Hardware component

The hardware set-up is made of many separate components:

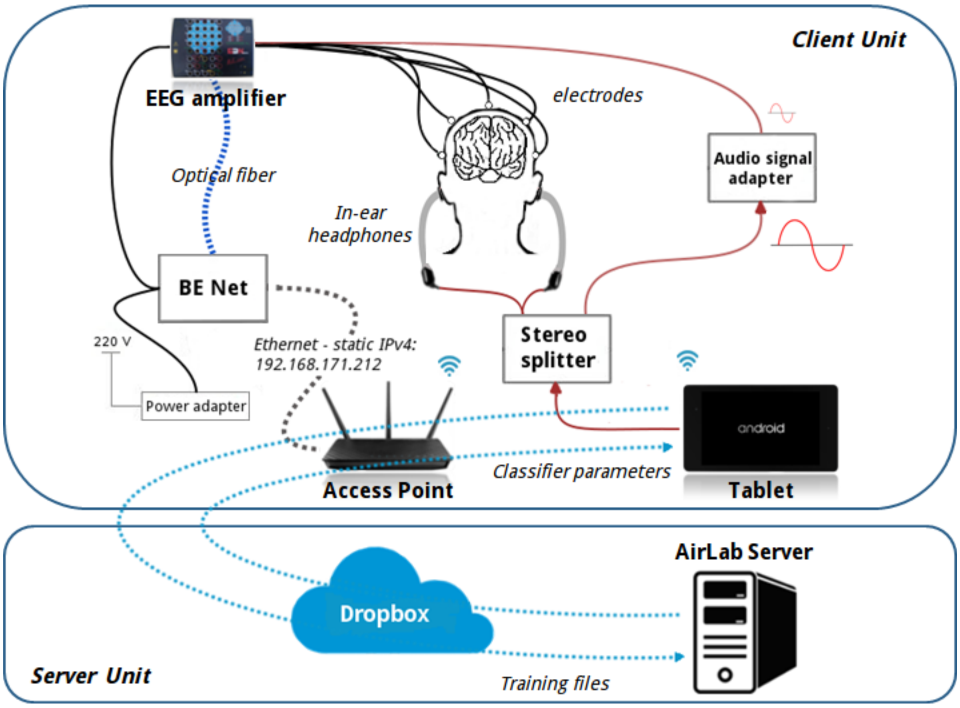


Figure A.1: Auditory BCI set-up scheme.

1. Tablet
2. Access Point
3. BE Net interface
4. BE Light amplifier
5. Audio stereo splitter
6. Clinical headphones
7. Audio signal adapter

The single components interact in the following way (A.1). In the following each single components is analyzed.

A.2.1 Tablet

The application has to run on Android operating system (3.0 up to *Lollipop* 5.1.1) with a working Wi-Fi connection. The tablet has to be connected to a network created from the access point.

A.2.2 Access Point

A generic access point will be sufficient. The network configuration is arbitrary as long as you use a range of IPv4 addresses with net mask 255.255.255.0. It is very important to leave one address free to be able to connect to the network, which has a specific static non-configurable IP (192.168.171.212).

A.2.3 BE Net

Ethernet interface equipped with an optical fiber for EEG data transmission from *EBNeuro*. Works as a power manager for the EEG amplifier. No specific configuration is mandatory, but it has to be plugged to a power source.

A.2.4 BE Light

EEG signal amplifier from *EBNeuro*. It has to be placed at least one meter to the patient, so that the electrodes can be easily connected. It is powered by the BE Net with its cable and an optical fiber.

A.2.5 Stereo splitter

Splits the audio signal generated from the tablet so that it can go both into the BE light through the clinical headphones.

A.2.6 Clinical headphones

In-ear headphones that have been modified in order to not interfere with the EEG signals reading, isolating the magnetic field generated.

A.2.7 Audio signal adapter

Electrical circuit that reduces the audio output voltage dynamics from the tablet into the BE Light. This cable has to be plugged in the 22 port and NEP port (distinguishable from the color of the connectors).

A.3 Patient preparation

Once the application is installed correctly on the tablet, it can be possible to proceed the stimulation. To do so it is extremely important to prepare the patient and positioning the electrodes correctly over its scalp according to the 10-20 system. To do this it is needed:

- conductive paste
- swabs
- grease removal preparatory gel
- a paper tape measure

According to the necessities, decide which electrodes to position and which channels activate in the application.

A.3.1 The 10-20 system

The 10-20 system defines the position of the electrodes according to anthropometric measurements. The nasion and the inion points are taken as guides on the anteroposterior midline while auricular points are used on the coronal-lateral side. The distances between the electrodes are 10% or 20% of the total distance measured from he nasion and the inion points as in Figure A.2.

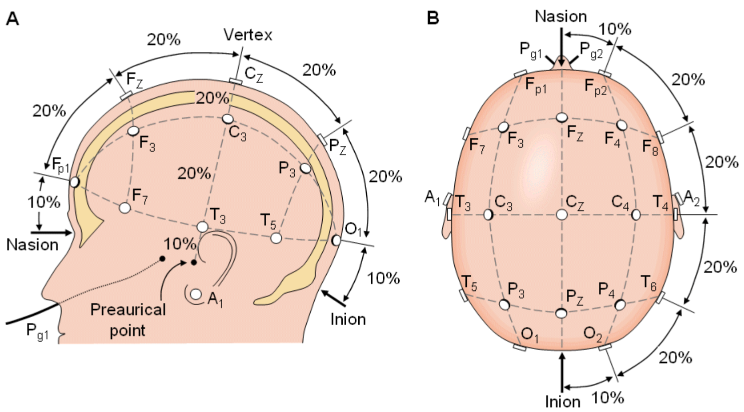


Figure A.2: Electrodes in the 10-20 system.

A.3.2 Patient Preparation Procedure

1. Identify using the paper tape the nasion and inion points as well as all the points in which you will position your electrodes. If necessary mark the precise spot using a marker.
2. Prepare the needed electrodes, cleaning them with the grease removal. Be sure that there is no residual grease on the electrodes.

3. Put some conductive paste on the electrodes, making sure that it covers the whole thing.

4. Place the electrodes on the previously marked points. Be sure to avoid trapping any hair between the skin and the electrode. The electrode has to be positioned in the position of the 10-20 system.

5. The first two electrodes positioned, should be the ISOGN (ground) and the NE (reference) and they should be positioned on the mastoid, behind the ear.

6. Position the remaining electrodes in the correct positions.

7. In the IMPEDANCE CHECK menu verify that the value of the impedances is correct. If the value is visualized in red or orange, it means that the impedance is too high. If this happen, replace the electrodes and trying to find the optimal value. If the problem is generic, it means that all the electrodes have a high impedance so replace the ISOGN and the NE electrodes.

8. If the problem continues, replace all the electrodes starting with the ISOGN and NE. Problems could be due to the electrodes not properly cleaned.

9. Position the clinical headphones on the patient in a comfortable way.

A.4 Menu of the application

A.4.1 ON home screen

After launching the application, the home screen is visualized (Figure A.3). Selecting the BCI menu, is possible to choose between the visual and auditory BCI (Figure A.5 on the left). In this manual only the Auditory BCI is treated. After having selected the auditory BCI, wait a few second to ensure that the connection has been established correctly. One it happens, a green sign with written CONNECTED will occur. The home menu of the acoustic BCI is shown in Figure A.5 on the right.

Inside the auditory BCI menu, you can choose between the following options:

- Training Session
- Impedance Check

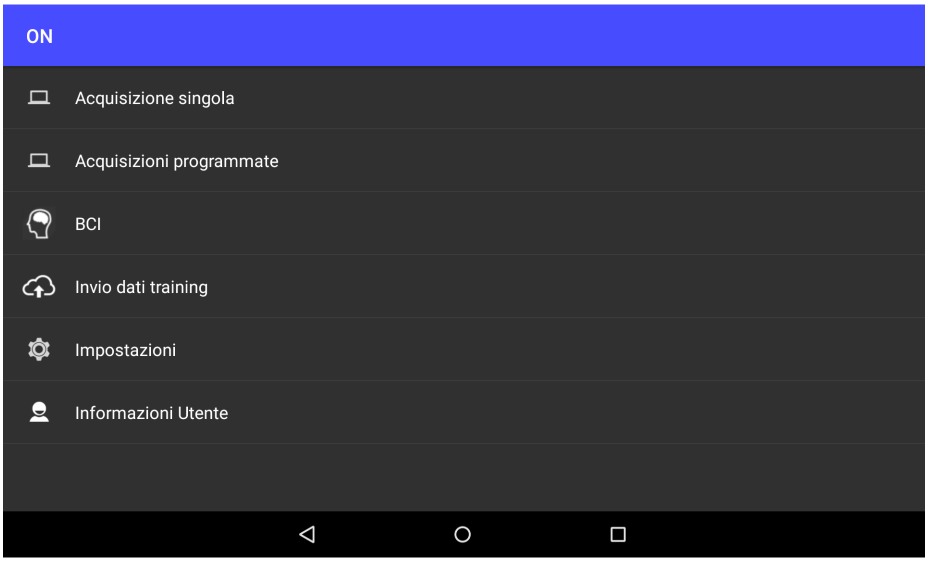


Figure A.3: Home Screen menu.

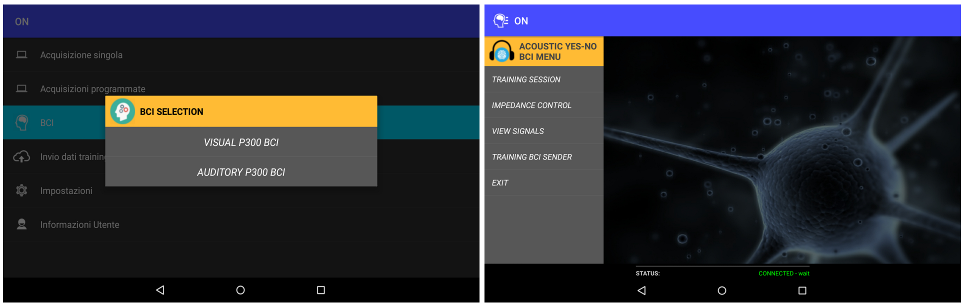


Figure A.4: Acoustic yes-no BCI menu.

- View Signals
- Training BCI Sender

A.4.2 Training session

In this interface it is possible to set the auditory stimuli that will reach the patient.

Main settings

In the first layout there are 11 audio channels, where is possible to activate or deactivate them through their selection. Once a channel is activated and

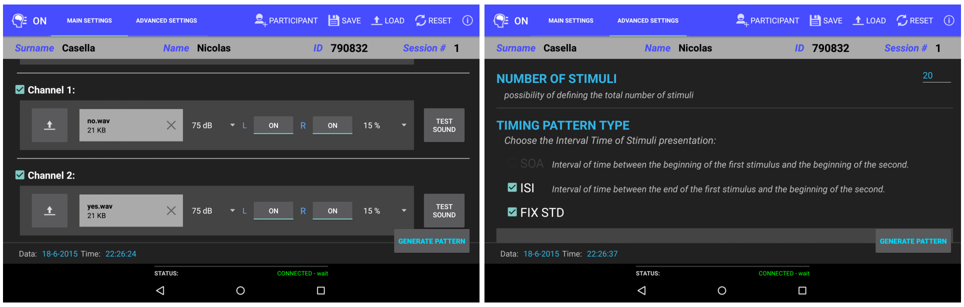


Figure A.5: Main settings and Advanced settings in training session interface.

than a wave audio file is loaded, is possible to define the acoustic audio synthesis for that file defining a Loudness level (in terms of Sound Pressure Level [dB]) with its direction in Mono Left, Mono Right or Stereo, and the percentage of trials for the final stimuli presentation (70 for frequent sound and 15 for deviants). In order to maintain a correct stimuli presentation is possible to prepare more than 1 channel for Frequent sound and more than 2 channels for Deviant sounds, but is necessary to leave selected at maximum 3 channels (1 channel for Frequent sound and 2 channels for the Binary deviants choice) otherwise the application does not generate the sequence in order to prevent errors and alert the user to see the User Guide.

Advanced settings

In this layout is possible to define:

- The total number of the acoustic stimuli presentation to the participant.
- The type of interval between two consecutive stimuli (SOA counts the time interval between the beginning of one stimulus and the beginning of the next one, while ISI counts the time interval between the end of one stimulus and the beginning of the next one).
- The relative timing presentation related to the type of interval (FIXED INTERVAL set the same time interval between all stimuli, INTERVALS set a randomized time intervals between stimuli chosen from five timing defined and RANDOM INTERVALS set a randomized time intervals chosen at random between a minimum and a maximum timings defined).

Patient credentials

Every time a training session starts with a new patient, it is important to insert the patient credentials (Figure A.6) selecting the button "ADD PARTICIPANT" from the top of the menu.

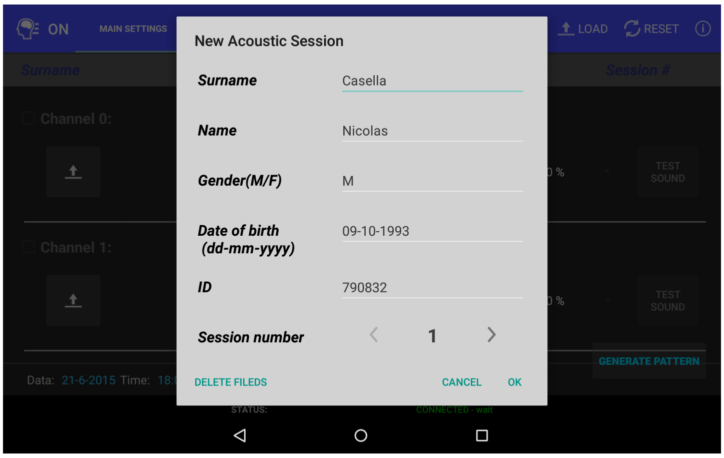


Figure A.6: Menu to insert patient credentials in the training session interface.

Generate pattern

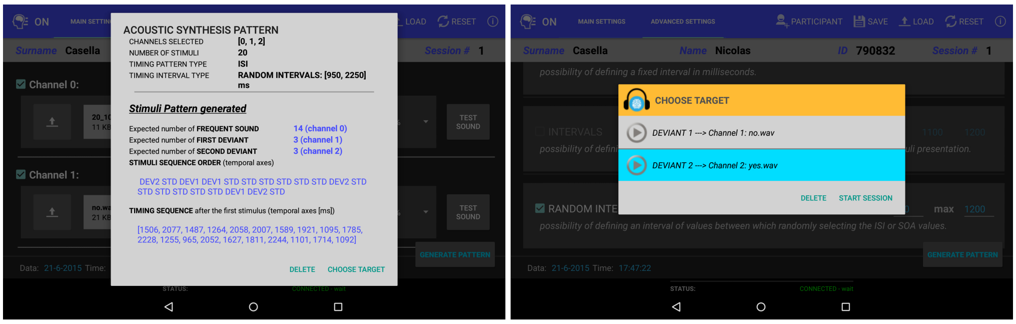


Figure A.7: Acoustic pattern generation and choose target selection.

When the session is created and all parameters are properly set, is possible to generate the sequence to deliver. A simple summary of the sequence and timings generated is shown (Figure A.7). After the selection of the CHOOSE TARGET button, it needs to define the current target that is sent via headphones to the patient to pay attention and then initialize the session selecting the START SESSION button.

A.4.3 Impedance check

In this menu (Figure A.8) it is possible to check the status of the electrodes positioned on the patient. A circle that changes color based on the value of the impedance represents each electrode. Optimal values are $< 5k\Omega$ and are represented by green; If the value is too high the circle will turn orange ($< 5k\Omega \ll 10k\Omega$) or red ($> 10k\Omega$).

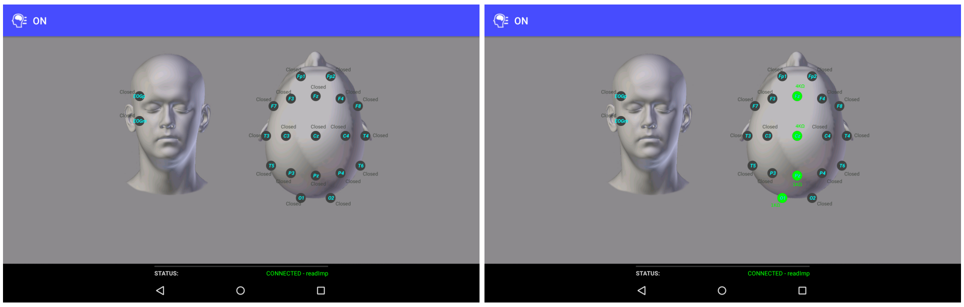


Figure A.8: Impedance check menu.

A.4.4 View signals

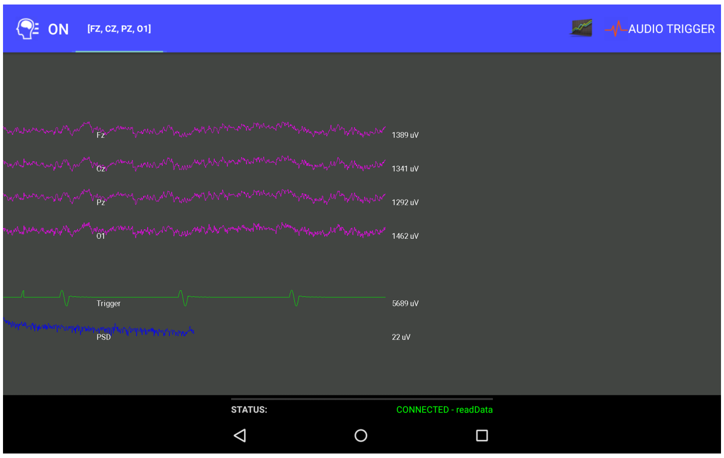


Figure A.9: View signals menu.

The menu (Figure A.9) is divided in a three parts. In the upper part of the screen (in purple) it is possible to visualize the signal trace for each activated channel. Every trace is associated with the name of the corresponding electrode and the peak-peak value in μV. In the central part of the screen (in green) there is the trigger channel. Whenever the patient will listen to a stimulus it will be synchronized with the trigger. To view the audio trigger is possible to play a sound loaded in the training interface and visualize it on the screen clicking on “AUDIO TRIGGER” button. In the bottom part of the screen is possible to visualize the power spectral density (in blue). It is computed using the first electrode positioned on the patient and it allows to visualize the power distribution with respect to the frequencies. This allows to visualize the peaks due to environmental electromagnetic noise. Is the possible to change the channel computing the PSD with in the top right corner button.

A.4.5 Training BCI sender

In the main menu choose “TRAINING BCI SENDER” to reach the screen of selection and send the training data (Figure A.10).

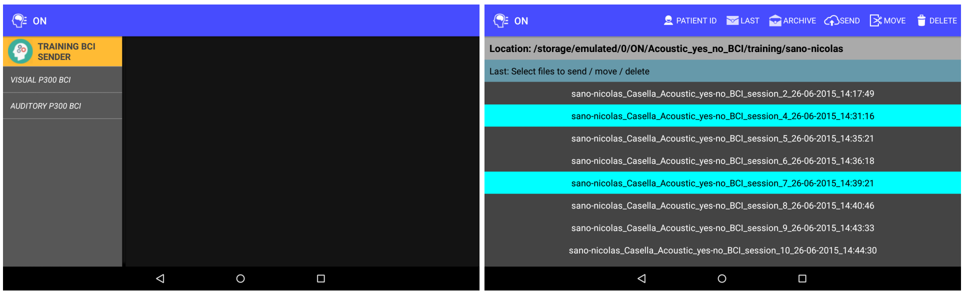


Figure A.10: Training BCI sender menu.

Here is possible to visualize the Patient ID folder selected that collects all the training sessions not unset yet (LAST button) or already archived (ARCHIVE button). By the selection of one or more training sessions is possible to execute one of these operations:

1. *PATIENT ID*: allows to browse the desired ID folders and visualize all the training files inside it.

2. *SEND*: allows to send the training session to the data elaboration server in order to obtain an update of the classifier. Sessions, after sending, will be automatically archived.

3. *MOVE*: change the position of the selected sessions from to the LAST folder and ARCHIVE folder (and vice-versa).

4. *DELETE*: delete the training session from the local memory. If the sessions are already been sent to the server, they are not deleted from the tablet storage.

A.5 An example of use

To avoid the inadvertent contact, the menu only reacts to prolonged selection on the screen. To start a new clean session insert the patient credentials (or confirm the previous ones already entered) clicking in the main menu on ADD PARTICIPANT Button.

Initially the SAVE button in the menu is disabled until the session creation. It is possible also to restore a session of a configuration saved before in device SD card just clicking on LOAD button, browsing the patient ID folder and select the desired configuration. There is a possibility to reset the session simply selecting the RESET button and if there is already a session created for a specific participant, this operation restore all parameters at their initial state (as a new creation session).

For a correct acoustic stimuli presentation for the training session is necessary to:

1. Select a channel.
2. Load an audio file for each selected channel.
3. Define the Loudness level for each selected channel.
4. Select the switches (R for Mono right, L for Mono left, and R + L for Stereo) for each selected channel.
5. Select the percentage of trials for each selected channel.
6. Define the total number of stimuli presentation (the default number is 7).
7. Select the type of the interval presentation between the stimuli SOA or ISI (Fix STD is automatically set by default).
8. Select and modify the desired time interval between the two stimuli for the type selected.
9. Activate the desired electrodes from the IMPEDANCE CHECK button in the main menu.

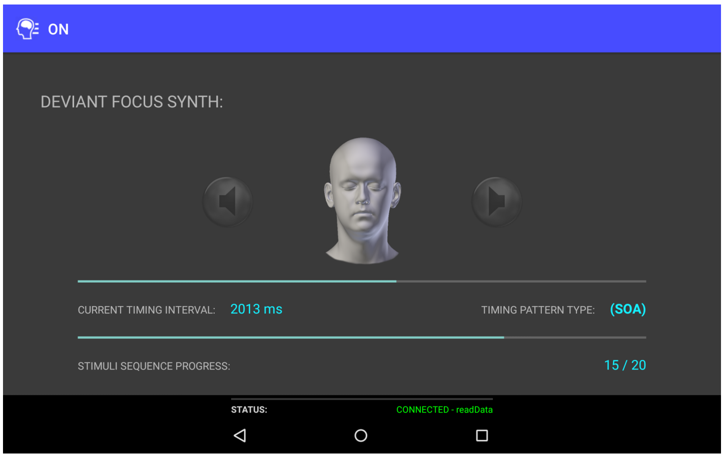


Figure A.11: Example of stimuli delivering.

10. Verify the correct signals visualization of the activated electrodes in the VIEW SIGNALS menu.

11. If everything is correctly visualized, go back to the training interface and select the on GENERATE PATTERN button to see the acoustic synthesis pattern generated by system.

12. Click on the SELECT TARGET button to confirm the training session.

13. After an acoustic message to the participant choose the current sound target for current session (deviant that participant have to focus during the session).

14. Click on the START SESSION button to initialize the session.

During the acoustic stimuli presentation the current stimulus synthesized, the progression of the total stimuli and the current delay progression are visualized (Figure A.11)