FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING

Master's Thesis

Software for Neurorehabilitation

Tomáš Kment

PILSEN, CZECH REPUBLIC

2025

FACULTY OF APPLIED SCIENCES
UNIVERSITY
OF WEST BOHEMIA

DEPARTMENT OF
COMPUTER SCIENCE
AND ENGINEERING

Master's Thesis

Software for Neurorehabilitation

Bc. Tomáš Kment

Thesis advisor
doc. Ing. Roman Mouček, Ph.D.

PILSEN, CZECH REPUBLIC **2025**

Citation in the bibliography/reference list:
KMENT, Tomáš. `Software for Neurorehabilitation`. Pilsen, Czech Republic, 2025. Master's Thesis. University of West Bohemia, Faculty of Applied Sciences, Department of Computer Science and Engineering. Thesis advisor doc. Ing. Roman Mouček, Ph.D.

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2024/2025

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Tomáš KMENT**
Osobní číslo:	**A22N0070P**
Studijní program:	**N0613A140037 Informatika a její specializace**
Specializace:	**Medicínská informatika**
Téma práce:	**Software pro neurorehabilitace**
Zadávající katedra:	**Katedra informatiky a výpočetní techniky**

Zásady pro vypracování

1. Seznamte se s neuroinformatickou laboratoří, vybranými terapeutickými roboty a jejich příslušenstvím.
2. Seznamte se s BCI systémy (a jejich principy) používanými v laboratoři a návaznými neurorehabilitačními postupy.
3. Navrhněte softwarovou aplikaci pro terapeutického robota (bod 1), BCI systém a neurorehabilitační postup (bod 2) se souhlasem vedoucího práce.
4. Navrženou sw aplikace dle bodu 3 implementujte a důkladně otestujte.
5. Zhodnoťte dosažené výsledky.

Rozsah diplomové práce:	**doporuč. 50 s. původního textu**
Rozsah grafických prací:	**dle potřeby**
Forma zpracování diplomové práce:	**tištěná/elektronická**
Jazyk zpracování:	**Angličtina**

Seznam doporučené literatury:

dodá vedoucí diplomové práce

Vedoucí diplomové práce:	**Doc. Ing. Roman Mouček, Ph.D.** Katedra informatiky a výpočetní techniky
Datum zadání diplomové práce:	**9. září 2024**
Termín odevzdání diplomové práce:	**15. května 2025**

L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Ing. Přemysl Brada, MSc., Ph.D.
vedoucí katedry

V Plzni dne 30. září 2024

Declaration

I hereby declare that this Master's Thesis is completely my own work and that I used only the cited sources, literature, and other resources. This thesis has not been used to obtain another or the same academic degree.

I acknowledge that my thesis is subject to the rights and obligations arising from Act No. 121/2000 Coll., the Copyright Act as amended, in particular the fact that the University of West Bohemia has the right to conclude a licence agreement for the use of this thesis as a school work pursuant to Section 60(1) of the Copyright Act.

V Plzni, on 14 May 2025

. . . .

Tomáš Kment

Abstract

This thesis presents the development of a modular neurorehabilitation system that integrates Brain-Computer Interface (BCI) technology with real-time EEG signal processing, a Unity-based virtual front-end, and robotic arm control. The primary objective is to enable real-time detection of movement intention in motor-impaired individuals and to support rehabilitation through interactive and engaging feedback.

The system was built around the **OpenBCI** platform for EEG acquisition and the **FourMotors** software for robotic assistance. A custom dataset was created using a structured rehabilitation scenario, and rigorous signal preprocessing methods—including filtering, artifact removal, and normalization—were applied to ensure data quality. Multiple classification pipelines were evaluated, including a convolutional neural network (CNN) trained on raw EEG signals, a multilayer perceptron (MLP) trained on extracted features such as CSP and bandpower, and a meta-classifier ensemble combining both approaches.

Offline evaluation demonstrated solid performance for subject-specific classification, while real-time implementation showed low-latency responsiveness suitable for interactive applications. However, generalization to unseen subjects remains a key challenge. Despite integration limitations, the final system demonstrates the feasibility of closed-loop neurorehabilitation using BCI and highlights critical areas for further research, such as improving subject-independent accuracy and refining robot feedback mechanisms.

This work contributes a practical and extensible foundation for future neurorehabilitation applications, combining clinical relevance with technical innovation.

Abstrakt

Tato diplomová práce představuje vývoj modulárního systému pro neurorehabilitaci, který kombinuje technologii rozhraní mozek-počítač (BCI) s reálným zpracováním EEG signálů, virtuálním rozhraním vytvořeným v Unity a řízením robotické paže. Hlavním cílem je umožnit detekci záměru pohybu u osob s poruchou hybnosti v reálném čase a podpořit rehabilitaci prostřednictvím interaktivní a motivující zpětné vazby.

Systém je postaven na platformě **OpenBCI** pro snímání EEG a softwaru **FourMotors** pro robotickou asistenci. V rámci strukturovaného rehabilitačního scénáře byl vytvořena vlastní datová sada a na získaná data byly aplikovány pokročilé metody předzpracování – filtrování, odstranění artefaktů a normalizace – za účelem zvýšení kvality signálu. Bylo otestováno několik klasifikačních metod: konvoluční neuronová síť (CNN) pracující s nezpracovanými EEG daty, vícevrstvý perceptron (MLP) využívající extrahované příznaky jako csp a pásmový výkon, a dále metaklasifikátor kombinující oba přístupy.

Offline testování prokázalo dobré výsledky při klasifikaci specifické pro jednotlivé uživatele, zatímco v reálném čase systém vykazoval nízkou latenci vhodnou pro interaktivní využití. Hlavní výzvou však zůstává zobecnění modelu na nové uživatele. Systém i přes určitá omezení v oblasti integrace, demonstruje proveditelnost uzavřené smyčky neurorehabilitace pomocí BCI a poukazuje na klíčové oblasti dalšího výzkumu, zejména zlepšení přesnosti u nových uživatelů a optimalizaci zpětné vazby ze strany robotického zařízení.

Tato práce přináší praktický a rozšiřitelný základ pro budoucí aplikace v oblasti neurorehabilitace, který spojuje klinickou relevanci s technickou inovací.

Keywords

Neurorehabilitation • Brain–Computer Interface (BCI) • EEG Signal Processing • Unity • Real-Time Classification • Motor Intention Detection

Acknowledgement

I would like to acknowledge and express my gratitude to my supervisor, Doc. Ing. Roman Mouček, Ph.D, for his helpfulness, prompt communication and consultation, which helped me significantly during uncertain times.

I want to thank my colleagues, MgA. Radka Krajíčková and Bc. Calum Aitchison, for the 3D models presented in this thesis. Without them, the visuals wouldn't be nearly as pleasant for the eye as they are.

Lastly, I would like to thank all my friends and colleagues who participated in the creation of the dataset.

Tomáš Kment,

Contents

Introduction 1

What if a person could regain control over a paralyzed limb—not through surgery or medication, but simply by thinking? This once-science-fiction concept is rapidly becoming reality, thanks to the advancement of brain–computer interfaces (BCIs). These technologies have the potential to transform neurorehabilitation by creating new channels through which patients can interact with assistive systems using only their brain signals.

Neurological disorders and injuries—such as stroke, Traumatic Brain Injury (TBI), or Spinal Cord Injury (SCI), often result in severe motor impairments, limiting independence and overall quality of life. Traditional rehabilitation methods, while beneficial, often lack precision, adaptability, and motivation-driven feedback. To address these limitations, researchers have increasingly turned to BCIs, particularly those based on Electroencephalography (EEG), which offer a non-invasive, low-cost, and temporally precise window into brain activity.

This thesis explores the design and implementation of a modular BCI-driven neurorehabilitation system that closes the loop between brain activity, real-time interpretation, and interactive feedback. The proposed system integrates (1) real-time EEG acquisition and signal processing using **OpenBCI** hardware, (2) robust classification models capable of distinguishing between resting and movement intention states, (3) a Unity-based virtual reality environment to promote user engagement, and (4) robotic assistance via the **FourMotors** system to provide physical feedback and support.

A structured rehabilitation scenario was created to collect a novel dataset, upon which multiple classification pipelines—including Convolutional Neural Network (CNN), Multilayer Perceptron (MLP), and meta-classifiers—were trained and evaluated both offline and in real-time conditions. The results demonstrate the feasibility and challenges of building a practical, low-latency, and extensible BCI system for motor recovery. By combining neurophysiological insights, modern machine learning techniques, and immersive feedback, this work contributes toward the vision of intuitive, brain-controlled rehabilitation systems capable of adapting to individual users and driving long-term recovery.

Physiological background

2

The human nervous system is a highly complex network responsible for coordinating voluntary and involuntary functions through intricate electrochemical processes. At its core, the system relies on neurons—specialized cells that transmit information via electrical impulses and neurotransmitter release. Understanding the physiological principles underlying neural function is essential for comprehending motor control, neuroplasticity, and rehabilitation, all of which play a crucial role in restoring movement and function after neurological injury.

This chapter provides a foundational overview of the physiological mechanisms governing neural activity, motor execution, and adaptation following injury. It begins with an exploration of neuron structure and function, detailing the processes of synaptic transmission and neural signalling. This is followed by an examination of motor control pathways, highlighting the role of the motor cortex, basal ganglia, cerebellum, and spinal cord in generating and refining movement. The concept of neuroplasticity is then discussed, emphasizing its significance in rehabilitation and recovery. Finally, the chapter concludes with an overview of common neurological disorders that necessitate rehabilitation interventions.

2.1 Fundamentals of Neurons and Neural Signaling

Neurons are the fundamental building blocks of the nervous system, responsible for transmitting and processing information through electrical and chemical signals. A typical neuron consists of three primary components: the dendrites, which receive input signals; the soma (cell body), which integrates incoming signals; and the axon, which transmits signals to other neurons via synapses [BCP20].

Neuronal communication occurs through action potentials, which are rapid electrical impulses generated when the neuron's membrane potential reaches a certain threshold. This process is driven by voltage-gated ion channels that regulate the flow of sodium (Na+) and potassium (K+) ions across the membrane [Hil01]. The resting membrane potential of a neuron, typically around -70 mV, is maintained by the sodium-potassium pump (Na+/K+ ATPase), which actively transports Na+ out of the cell and K+ into the cell. When a neuron is sufficiently stimulated, a rapid influx of Na+ ions leads to depolarization, followed by repolarization as K+ ions exit the cell. This cyclical process enables the propagation of electrical signals along the axon.

At the synapse, the electrical signal is converted into a chemical signal via neurotransmitters, which cross the synaptic cleft and bind to receptors on the postsynaptic neuron, facilitating signal transmission [Pur+18]. Different types of neurotransmitters play crucial roles in regulating neural activity. For instance, glutamate acts as the primary excitatory neurotransmitter in the central nervous system, whereas GABA (gamma-aminobutyric acid) serves as the main inhibitory neurotransmitter, balancing excitatory signals to prevent excessive neuronal firing.

Other neurotransmitters, such as dopamine, serotonin, and acetylcholine, contribute to various functions, including movement regulation, mood modulation, and cognitive processes. Dysregulation in neurotransmitter function is linked to numerous neurological disorders. For example, a deficiency in dopamine is a hallmark of Parkinson's disease, leading to motor impairments, while imbalances in serotonin levels are associated with mood disorders such as depression and anxiety.

Beyond synaptic transmission, neurons exhibit plasticity, the ability to strengthen or weaken connections based on experience and activity. This is a fundamental aspect of learning and memory and plays a crucial role in neurorehabilitation. Mechanisms of plasticity include Long-Term Potentiation (LTP), which strengthens synaptic connections through repeated stimulation, and Long-Term Depression (LTD), which weakens them when activity decreases [BC93]. Understanding these principles is essential for designing effective rehabilitation strategies for individuals recovering from neurological injuries.

Furthermore, neurons can be classified based on their function: sensory neurons detect external stimuli and relay information to the brain, motor neurons transmit signals from the brain and spinal cord to muscles, and interneurons facilitate communication between sensory and motor neurons. These diverse neuron types work together to process and execute complex behaviours, underscoring their importance in neurorehabilitation and Brain-Computer Interface (BCI) applications (more about that topic in Section 4).

2.2 Brain Regions Involved in Movement and Recovery

The brain is a highly complex organ composed of various interconnected regions, each playing a crucial role in cognitive and motor functions. Understanding the structure and function of these brain regions is essential for comprehending movement generation, coordination, and recovery following neurological injury. Different parts of the brain work in unison to facilitate voluntary movement, balance, sensory perception, and motor learning. This section provides an overview of the primary brain regions involved in movement control and recovery mechanisms. A visual representation of these brain regions can be found in Figure 2.1, which illustrates their relative positions and functional significance. Some of the neurological disorders and injuries mentioned in this section will be discussed in more detail further in Section 2.5.

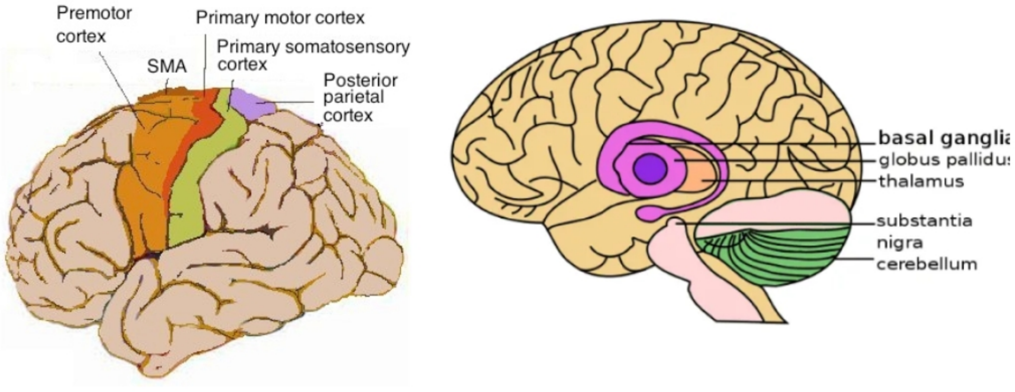


Figure 2.1: Brain regions related to movement [YD16]

2.2.1 Primary Motor Cortex

Located in the precentral gyrus of the frontal lobe, the primary motor cortex is responsible for generating neural impulses that control voluntary muscle movements. It serves as the final output region for motor commands before they are transmitted to the spinal cord and muscles. The motor cortex is somatotopically organized in what is known as the motor homunculus, meaning specific areas of M1 correspond to particular body parts, with larger representations for regions requiring fine motor control, such as the hands and face.

In cases of injury, such as stroke, the plasticity of M1 allows other brain regions to compensate for lost function. Rehabilitation techniques like Constraint-Induced Movement Therapy (CIMT)) and brain-computer interfaces (BCIs) can facilitate the reorganization of M1 to enhance recovery.

2.2.2 Supplementary Motor Area

The Supplementary Motor Area (SMA) is located anterior to the primary motor cortex and plays a key role in planning complex movements and coordinating sequences of motor actions. The SMA is particularly important for internally guided movements, such as those performed without external cues.

In neurorehabilitation, training methods that involve repetitive motor planning tasks and imagery-based exercises can help patients regain movement abilities when the SMA is affected by neurological disorders.

2.2.3 Premotor Cortex

The Premotor Cortex, located just anterior to M1, is involved in the selection and execution of movements based on external sensory cues. It works closely with the primary motor cortex and sensory regions to adjust motor plans according to environmental feedback. The premotor cortex is also divided into dorsal and ventral sections, with distinct roles in motor preparation and sensory integration.

Damage to the premotor cortex can result in apraxia, a condition where individuals struggle to perform learned movements despite having the necessary motor strength. Rehabilitation strategies often involve sensory-guided movement training to enhance functional recovery.

2.2.4 Basal Ganglia

The Basal Ganglia is a group of subcortical nuclei involved in movement initiation, inhibition, and fine-tuning, ensuring accuracy and coordination through a process called error correction [Doy00]. It plays a crucial role in motor learning, habit formation, and the execution of automatic movements. The basal ganglia receive input from various cortical areas and send processed motor signals to the thalamus before reaching M1.

Disorders of the basal ganglia, such as Parkinson's disease and Huntington's disease, are characterized by impaired movement regulation. Treatments like Deep Brain Stimulation (DBS) and pharmacological interventions targeting dopamine pathways help mitigate symptoms and restore function.

2.2.5 Cerebellum

Located at the back of the brain, the Cerebellum is essential for motor coordination, balance, and error correction during movement. It receives sensory feedback from the body and adjusts motor commands to ensure smooth and precise actions. The cerebellum is also crucial for motor learning, helping individuals adapt to new movement patterns through repetition and feedback.

Damage to the cerebellum can result in ataxia, a condition marked by uncoordinated movements, poor balance, and difficulty in performing fine motor tasks. Rehabilitation strategies, such as vestibular training and coordination exercises, can improve function in individuals with cerebellar dysfunction.

2.2.6 Somatosensory Cortex

The Somatosensory Cortex, located in the postcentral gyrus of the parietal lobe, processes sensory information related to touch, proprioception, and body position. This region plays a critical role in providing the feedback necessary for precise motor control and movement adjustments.

Patients with somatosensory deficits often experience difficulties with motor control due to impaired feedback processing. Sensory retraining exercises and tactile stimulation therapy are commonly used in rehabilitation to improve sensory-motor integration.

2.3 Neurophysiology of Motor Control

It was mentioned in Section 2.2 that the control of voluntary movement is primarily governed by the motor cortex and that it is also responsible for generating neural impulses that direct muscle contractions, while the premotor cortex and supplementary motor area (SMA) contribute to motor planning and coordination [Kan+13].

In the article *The Neural Basis of Motor-Skill Learning*, Willingham suggested that four processes support motor-skill learning and that we can observe two modes of motor control. The four processes of motor control are as follows [Wil99] (with visual example in Figure 2.2):

(a) **Strategic process** - The actor selects the environmental goal of the movement. This process can support motor-skill learning through the selection of successively more effective goals.

(b) **Perceptual-motor integration** - The actor selects spatial targets for movements that will achieve the environmental goal. The environmental goal is selected in allocentric space (i.e., a coordinate system in which objects are located relative to one another), but the target for movement is selected in egocentric space (i.e., a coordinate system anchored on a part of the body). Allocentric space depends on vision, and egocentric space depends on proprioception (information about the position of the body that comes from receptors in the muscles, tendons, joints, and skin), so learning becomes necessary when the relationship between them is changed.

(c) **Sequencing process** - The third motor-control process sequences spatial targets for movement. Learning supported by this sequencing process occurs when the actor must make the same sequence of movements repeatedly.

(d) **Dynamic process** - Translates the sequence of egocentric spatial targets into a pattern of muscle activation. This process could support skill learning when the relationship between egocentric space and muscle movements is poorly represented (e.g., fine movements made with the non-preferred hand).

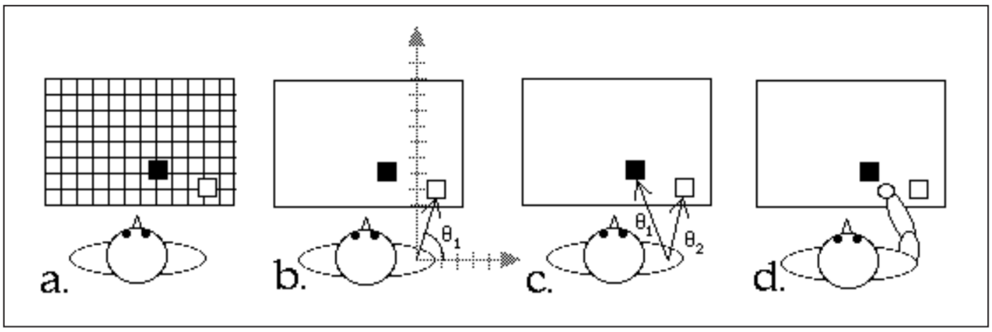


Figure 2.2: Four processes of motor control. In the strategic process (a), the actor decides to move a drinking glass (filled square) to a new location (empty square). The spatial locations are described in allocentric space (i.e., relative to the table). In perceptual-motor integration (b), the spatial locations are translated into egocentric space, in this case, relative to the location of the shoulder. In the sequencing process (c), the two spatial locations are sequenced, to ensure that the current location of the glass is reached first, and then the goal location of the glass. Finally, in the dynamic process (d), the spatial targets are translated into a pattern of muscle activation to move the hand to the targets. [Wil99]

Regarding the two mentioned modes, Willingham proposed that the four processes can operate either in *unconscious mode* or in *conscious mode*. In the *unconscious mode*, the actor is conscious only of setting the environmental goal. The other processes operate outside of consciousness. In the *conscious mode*, the strategic process not only selects the environmental goal, but also selects and sequences the spatial targets for movement. Under typical circumstances, the actor employs the unconscious mode and is conscious only of wanting the glass moved from one location to the other. However, the actor can also use the conscious mode and consciously consider the exact location in which the glass is to be grasped and the sequence of movements necessary to move the glass. When the conscious mode is used, the output of the strategic process replaces the output of the sequencing and perceptual-motor integration processes [Wil99].

This will be important later in the Motor Imagery (MI) rehabilitation in connection with BCI and EEG signal retrieval.

2.4 Neuroplasticity and Rehabilitation

Neuroplasticity refers to the brain's ability to reorganize and form new neural connections in response to learning, experience, or injury. This phenomenon underlies both normal cognitive development and the brain's ability to recover from damage. Neuroplasticity is fundamental to neurorehabilitation, as it enables the restoration of function following neurological damage [KG11]. It occurs at various levels, including synaptic, structural, and functional changes within the brain. These changes allow the nervous system to adapt to new challenges, learn new skills, and compensate for lost function.

Neuroplasticity can take place under two main conditions and is divided into - **Experience-Dependent Plasticity** and **Injury-Induced Plasticity**. The first type of plasticity is driven by learning and practice. Activities like playing an instrument, acquiring a new language, or engaging in motor training exercises can induce neuroplastic changes in relevant brain regions. This claim is also supported by **Hebbian learning** - often summarized as "neurons that fire together, wire together" [Heb49]. The second condition is triggered after brain damage, where the brain undergoes a natural reorganization process to compensate for lost function. This form of plasticity is the basis of most rehabilitation interventions, helping individuals recover abilities through structured therapy.

Through several mechanisms, neuroplasticity can manifest, including [Nud13]:

- **Synaptic Plasticity** - Modifications in the strength of synapses through processes such as LTP, which strengthens synaptic connections through repeated stimulation, and LTD, which weakens them when activity decreases [BC93]. These processes are crucial for memory formation and motor learning.

- **Dendritic Branching and Axonal Sprouting** - Following injury, neurons can grow new dendrites and axons to establish new synaptic connections. This process enhances communication between surviving neurons and compensates for lost pathways.

- **Cortical Remapping** - When brain regions are damaged, adjacent or functionally related areas may take over lost functions. This reorganization is particularly evident in stroke recovery, where the undamaged hemisphere may assume motor control for affected limbs.

- **Neurogenesis** - In certain brain regions, such as the hippocampus, new neurons can be generated even in adulthood. While neurogenesis is limited in most parts of the brain, its role in recovery is still being explored in neurorehabilitation research.

Regarding rehabilitation, neuroplasticity is leveraged to restore lost functions and improve motor control. Several evidence-based interventions promote plasticity, such as CIMT (which forces the use of an impaired limb by restricting the movement of the unaffected limb, thereby encouraging cortical reorganization and improving motor function), BCI (described in more detail in Section 4), Virtual Reality and Mirror Therapy or Repetitive Task Training. Studies using functional MRI (fMRI) and EEG have demonstrated shifts in motor activity representation during rehabilitation [Joh+02]. Bilateral motor training, intensive physiotherapy, and electrical stimulation have also been shown to promote beneficial neuroplastic changes, ultimately improving movement outcomes.

2.5 Common Neurological Disorders Requiring Rehabilitation

Neurorehabilitation is particularly crucial for conditions that impair motor function, as it aims to restore movement, cognition, and independence. Since the causes of these disorders differ in most cases, this section will be further divided into subsections, providing descriptions and treatments for these disorders in terms of rehabilitation.

2.5.1 Stroke

Stroke is one of the leading causes of disability worldwide, resulting from either ischemic (blockage of blood flow) or hemorrhagic (rupture of a blood vessel) events in the brain [LBK11]. The resulting damage often leads to hemiparesis (weakness on one side of the body), loss of coordination, and difficulties with speech and cognition.

Rehabilitation strategies for stroke focus on task-oriented motor training, CIMT, and mirror therapy, all of which leverage neuroplasticity to reorganize surviving neural circuits [Dob05]. Studies using functional MRI have demonstrated cortical reorganization following intensive rehabilitation, particularly in the motor cortex

[JR02]. Moreover, the use of brain-computer interfaces (BCIs) and robotic-assisted rehabilitation has shown promising results in improving motor recovery [Soe15].

2.5.2 Traumatic Brain Injury (TBI)

Traumatic Brain Injury (TBI) results from a sudden impact to the head, causing widespread neuronal damage and deficits in motor, cognitive, and emotional functions. The severity of TBI varies, ranging from mild concussions to severe brain trauma leading to long-term disability [Maa+17]. Recovery depends on the extent of damage and the brain's ability to reorganize and compensate for lost functions.

Rehabilitation approaches for TBI include cognitive-motor therapy, task-specific training, and multisensory integration exercises to improve motor and cognitive abilities. Neurostimulation techniques such as Transcranial Magnetic Stimulation (TMS) and transcranial Direct Current Stimulation (tDCS) are being investigated as potential adjunct therapies to enhance neural recovery [Dun16].

2.5.3 Neurodegenerative Diseases

Neurodegenerative diseases are progressive disorders that gradually impair motor control, cognition, and autonomic functions. Unlike stroke and TBI, which involve acute damage, neurodegenerative diseases lead to continuous neuronal loss over time. The most well-known are:

- **Parkinson's Disease (PD)** - Characterized by dopamine deficiency in the basal ganglia, leading to bradykinesia, rigidity, and tremors. Deep Brain Stimulation (DBS) and intensive gait training have been shown to improve motor function in PD patients [KL15].

- **Multiple Sclerosis (MS)** - An autoimmune disease that damages myelin sheaths, affecting nerve signal transmission. Rehabilitation strategies focus on Functional Electrical Stimulation (FES), aerobic exercise, and cognitive therapy to maintain function and delay progression.

- **Amyotrophic Lateral Sclerosis (ALS)** - A neurodegenerative condition affecting motor neurons, leading to muscle atrophy and respiratory failure. Rehabilitation efforts aim to preserve mobility and quality of life through assistive technologies, speech therapy, and adaptive exercise programs [Har17].

2.5.4 Spinal Cord Injury (SCI)

Spinal Cord Injury (SCI) results in varying degrees of paralysis depending on the location and severity of the injury. Complete SCI leads to loss of function below the injury site, while incomplete SCI may allow some degree of movement and sensation.

Rehabilitation strategies for SCI emphasize locomotor training, robot-assisted therapy, and neuromodulation techniques such as Epidural Electrical Stimulation (EES), which has shown potential for restoring motor function in individuals with severe SCI [Ang18]. Additionally, brain-computer interfaces (BCIs) are being explored to enable motor control through neural signal decoding.

EEG 3

Electroencephalography (EEG) is a non-invasive[1] neurophysiological technique used to measure electrical activity in the brain. It provides valuable insights into brain function, cognitive processes, and neural disorders by capturing the oscillatory patterns generated by neuronal activity [NS05]. EEG has been widely used in research and clinical applications, including neurorehabilitation, brain-computer interfaces (BCIs), and the diagnosis of neurological conditions such as epilepsy and sleep disorders [MM12].

This chapter explores the fundamental principles of EEG, including its historical development, underlying physiological mechanisms, and technical aspects of signal acquisition. It also examines the different types of EEG signals, their relevance to brain function, and the methods used for data processing and analysis.

3.1 Principles of EEG Measurement

EEG records electrical activity generated by the synchronous firing of neurons in the cerebral cortex. These signals are captured by electrodes placed on the scalp, which detect voltage fluctuations due to ion flow in neural tissue. The recorded signals reflect the collective activity of thousands to millions of neurons, primarily from the pyramidal cells located in the cerebral cortex [Luc14].

The basic principle behind EEG is that neurons communicate through electrochemical processes, generating electrical potentials that can be measured externally. The primary contributors to EEG signals are postsynaptic potentials rather than action potentials, as the summation of these potentials produces detectable voltage changes at the scalp [Buz12].

Based on these principles, EEG offers several advantages over other neuroimaging techniques, mainly:

- **High Temporal Resolution** - EEG can detect neural activity on a millisecond timescale, making it one of the fastest methods available for studying

[1] Note the existence of invasive EEG method, but for the sake of this thesis, it will not be covered.

brain function [And04].

- **Non-Invasiveness** - Unlike intracranial recording techniques, EEG does not require surgical implantation of electrodes, making it widely accessible for clinical and research applications.
- **Cost-Effectiveness** - EEG is relatively inexpensive compared to methods like functional MRI (fMRI) or Magnetoencephalography (MEG), making it a practical tool for large-scale studies [Lop10].

However, EEG also suffers from certain limitations, chiefly:

- **Limited Spatial Resolution** - Because EEG measures electrical activity from the scalp, it lacks the ability to localize deep-brain activity precisely [MM12].
- **Susceptibility to Artifacts** - EEG signals are easily contaminated by artifacts from eye movements, muscle activity, and electrical noise, requiring rigorous preprocessing techniques [DSM07].
- **Volume Conduction Effects** - Electrical activity from different brain regions can mix as it propagates through the skull and scalp, reducing the ability to isolate specific neural sources [Sri+96].

3.2 Electrode Placement and Signal Acquisition

It was outlined in Section 3.1 that the signal is acquired through electrodes. In conventional scalp EEG, the recording is obtained by placing electrodes on the scalp with a conductive gel or paste. Many systems typically use electrodes, each of which is attached to an individual wire. Some systems use caps or nets into which electrodes are embedded; this is particularly common when high-density arrays of electrodes are needed. The locations and names of specific electrodes are defined by the *International 10-20 system*, which will now be described in greater detail because of its use in the practical part of this thesis.

3.2.1 10-20 System

The *International 10–20 system* is a globally accepted approach for positioning scalp electrodes during EEG examinations, sleep studies, and research experiments. It was developed to ensure standardized testing procedures, making it easier to compare, reproduce, and analyze results in both clinical and research settings. The system establishes a correlation between electrode placement and the underlying brain regions, particularly the cerebral cortex. The name "10–20" comes from the spacing between electrodes, which is determined as either 10% or 20% of the total distance across the skull, measured front-to-back or side-to-side [Kle+99]. A key reference measurement runs from the nasion (the point between the forehead and nose) to the inion (the prominent bone at the back of the skull). Other common landmarking methods involve measuring from one ear to the other, often passing over the top of the head, using anatomical reference points like the tragus, auricle, and mastoid. The visual description of the system is portrayed in Figure 3.1.

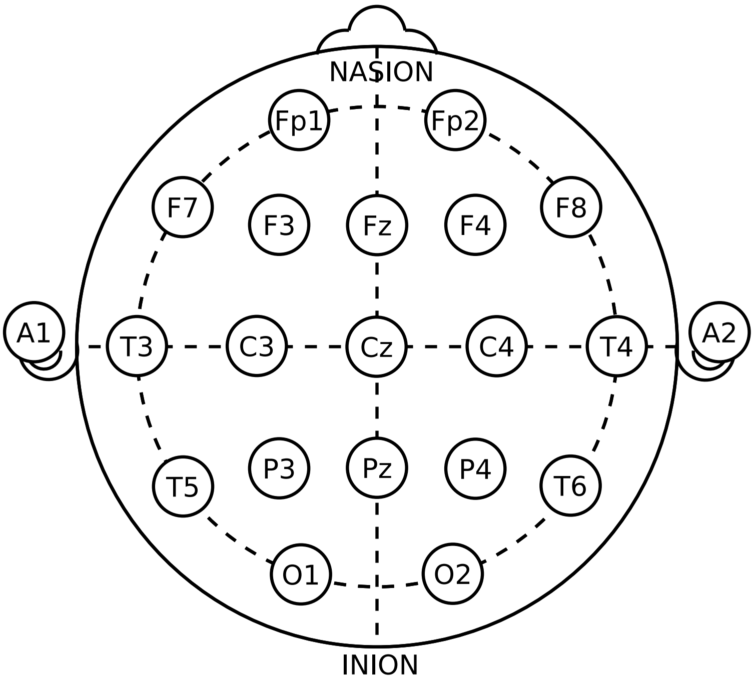


Figure 3.1: Electrode names and locations of *International 10-20 system* [Roj+18]

Each electrode placement is labelled with a letter corresponding to the brain region it monitors: pre-frontal (**Fp**), frontal (**F**), temporal (**T**), parietal (**P**), occipital (**O**), and central (**C**) [Kle+99; NL04]. Although there is no distinct "central lobe," the electrodes marked with "C" can pick up EEG signals associated with frontal, temporal, or parietal-occipital activity, depending on individual variations. These central electrodes are always used in polysomnography to assess sleep stages. Additionally, there are "Z" (zero) sites, which indicate electrodes positioned along the midline sagittal plane of the skull, such as **FpZ**, **Fz**, **Cz**, and **Oz**. These midline electrodes primarily serve as reference points rather than recording lateralized brain activity, as they sit above the corpus callosum and do not distinctly represent either hemisphere. In polysomnography, diagnostic EEG studies, and epilepsy evaluations, "Z" electrodes often function as ground or reference points, particularly in cases of suspected seizure activity or clinical brain death. The number of electrodes required, as well as their precise placement, depends on the specific clinical or research purpose.

Electrodes are numbered to indicate their placement on the head: even numbers (**2,4,6,8**) designate sites on the right hemisphere, while odd numbers (**1,3,5,**7) refer to placements on the left [Kle+99; NL04]. This numbering convention is consistent across EEG (brain activity), EOG (eye movement), and ECG (heart activity) electrode placements. Meanwhile, EMG (electromyogram) electrodes, which are typically placed on the chin, are often labelled as "right," "left," and "reference" or "common" since only three are usually used, and they can be referenced differentially with EEG and EOG signals. The "A" designation (sometimes referred to as "M" for the mastoid process) corresponds to the bony area located just behind the outer ear, which is more pronounced in some individuals than others. In a standard polysomnography setup, commonly used electrodes include **F3**, **F4**, **Fz**, **Cz**, **C3**, **C4**, **O1**, **O2**, **A1**, and **A2** (or M1 and M2 for the mastoid references) [Kle+99]. **Cz** and **Fz** serve as common reference points for EEG and EOG electrodes, while A1 and A2 provide contralateral referencing for EEG signals.

3.2.2 Acquisition of the Signal

Each electrode is linked to one input of a differential amplifier, with each amplifier corresponding to a pair of electrodes. A common reference electrode is connected to the other input of each amplifier. These amplifiers enhance the voltage difference between the active electrode and the reference, typically by a factor of 1,000 to 100,000 (equivalent to a power gain of 60–100 dB). Most modern EEG systems are digital, meaning the amplified signal is first processed through an anti-aliasing filter before being converted into a digital format using an analog-to-digital converter. In clinical scalp EEG, digitization generally occurs at sampling rates between 256 and 512 Hz, while certain research applications may utilize rates as high as 20 kHz [NL04].

During EEG recording, various activation techniques can be applied to elicit normal or abnormal brain activity that might not appear under resting conditions. These techniques include hyperventilation, photic stimulation (using a strobe light), eye closure, mental exertion, sleep, and sleep deprivation.

The digital EEG data is stored electronically and can be filtered for visualization. Typically, a high-pass filter is set between 0.5 and 1 Hz to remove slow-moving artefacts such as electrogalvanic signals and movement-related noise. A low-pass filter, ranging between 35 and 70 Hz, is used to eliminate high-frequency artefacts like muscle activity. Additionally, a notch filter is commonly applied to remove interference from electrical power sources, which operate at 60 Hz in the United States and 50 Hz in many other countries [NL04].

A typical adult human EEG signal, measuring between 10 μV and 100 μV in amplitude at the scalp, represents the voltage difference between two electrodes [Aur+04]. Its display for interpretation, known as a **montage**, can be arranged in various formats:

- **Sequential montage** - In this setup, each channel (or waveform) represents the voltage difference between two neighbouring electrodes, creating a sequence of connected comparisons across the electrode array.
- **Referential montage** - In a referential montage, each channel represents the voltage difference between a specific electrode and a designated reference electrode. Unlike the recording electrodes, the reference does not have a fixed standard position but is placed separately. Midline locations, such as Cz, Oz, or Pz, are commonly used to prevent signal bias between hemispheres when serving as an online reference. Additionally, offline references are also widely utilized.
- **Average reference montage** - The outputs of all of the amplifiers are summed and averaged, and this averaged signal is used as the common reference for each channel.
- **Laplacian montage** - Each channel represents the difference between an electrode and a weighted average of the surrounding electrodes [NP91].

3.3 Frequency Bands and Their Functional Significance

EEG activity is generally categorized into two components: **rhythmic activity** and **transients**. Rhythmic activity is further classified into frequency bands, with specific ranges assigned standardized names (e.g., activity between 4–7 Hz is termed "Theta"). While these classifications are somewhat arbitrary, they originated from observed patterns where specific frequency ranges were associated with distinct scalp distributions or biological significance. Most of the cerebral signal observed in the scalp EEG falls in the range of 1–20 Hz (activity below or above this range is likely to be artefactual, under standard clinical recording techniques). Waveforms are subdivided into bandwidths known as Alpha, Beta, Theta, and Delta[2] to signify the majority of the EEG used in clinical practice [Tat14].

Delta waves refer to frequencies up to **4 Hz**, characterized by their slow oscillations and high amplitude. In adults, they typically appear during slow-wave sleep,

[2] It is worth noting the presence of Gamma and Mu frequency bands; however, since they are not commonly used in standard clinical EEG recordings, they will not be discussed in detail.

while in infants, they are a normal feature of brain activity. Pathologically, delta waves can be observed focally in cases of subcortical lesions or more diffusely in conditions such as metabolic encephalopathy, hydrocephalus, or deep midline lesions. In adults, they are most prominent in the frontal region (e.g., FIRDA—Frontal Intermittent Rhythmic Delta), whereas in children, they are more commonly seen in the occipital region (e.g., OIRDA—Occipital Intermittent Rhythmic Delta).

Theta waves range from **4 Hz to** 7 **Hz** and are typically present in young children. In older children and adults, they may appear during drowsiness, arousal, or even during meditation. While theta activity is normal in certain contexts, excessive theta for a person's age is considered abnormal. It may indicate focal disturbances in cases of subcortical lesions or appear more diffusely in conditions such as metabolic encephalopathy, deep midline disorders, or hydrocephalus. Conversely, theta waves have also been linked to states of relaxation, meditation, and creativity [CP06].

Alpha waves fall within the **8–12 Hz** frequency range and typically emerge when the eyes are closed and the individual is in a relaxed state, diminishing with eye opening or mental activity. In young children, the posterior basic rhythm is slower than 8 Hz, technically placing it in the theta range. Beyond the posterior rhythm, other forms of alpha activity exist, such as the mu rhythm, which appears in the contralateral sensory and motor cortical areas when the hands and arms are at rest, and the "third rhythm," which manifests in the temporal or frontal lobes [FRV01]. While alpha activity is generally normal, it can be pathological in certain cases, such as alpha coma, where diffuse alpha waves persist in an unresponsive comatose state

Beta waves range **from 13 Hz to approximately 30 Hz** and typically appear symmetrically on both hemispheres, with the highest prominence in the frontal region. This frequency band is strongly associated with motor activity and tends to decrease during active movement [PL99]. Low-amplitude beta waves with multiple fluctuating frequencies are often linked to states of active thought, concentration, or anxiety. In contrast, rhythmic beta waves with a dominant frequency pattern can indicate certain neurological conditions, such as Dup15q syndrome, or be influenced by medications, particularly benzodiazepines. Beta activity may be absent or diminished in regions affected by cortical damage. It is the predominant rhythm in individuals who are awake, alert, anxious, or have their eyes open.

A visual summary of the waveforms is portrayed in Figure 3.2[3]

[3] The practice of using only whole numbers in the definitions comes from practical considerations in the days when only whole cycles could be counted on paper records. This leads to gaps in the range definitions, as seen in the figure. The theoretical definitions have always been more carefully defined to include all frequencies. Unfortunately, there is no agreement in standard reference works on what these ranges should be.

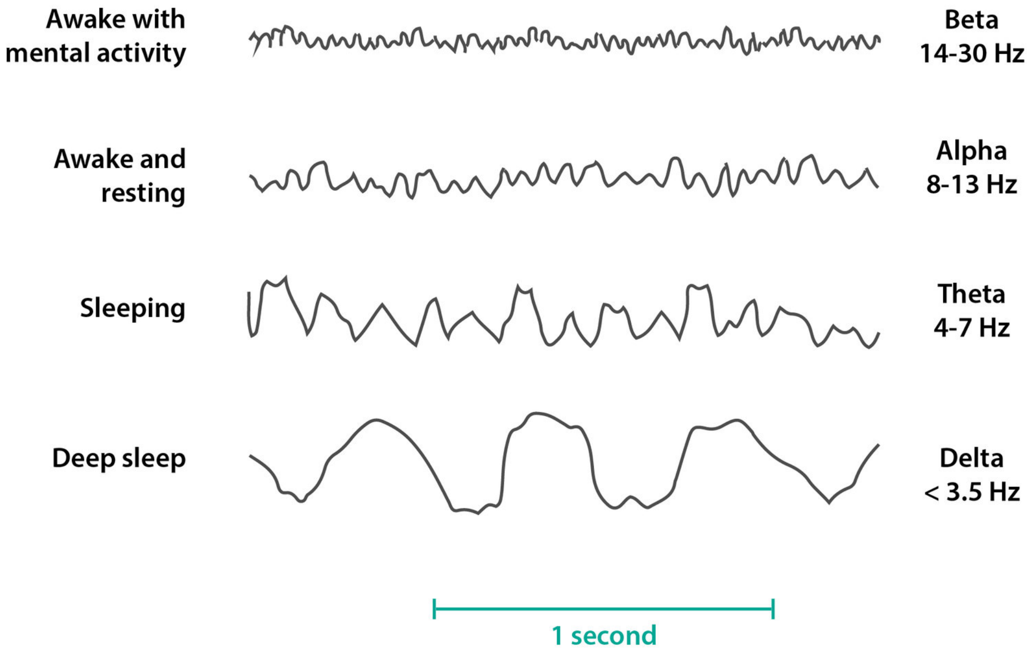


Figure 3.2: Visual summary of mentioned EEG waves [Siv17]

3.4 Data Processing and Artifact Removal

EEG is a highly valuable tool for analyzing brain activity; however, the recorded signal is often affected by artefacts, which can complicate data interpretation. Artefacts refer to any signals that do not originate from brain activity, and their removal remains a challenge despite the existence of various filtering algorithms. These unwanted signals can stem from technical issues, such as faulty electrodes, electrical interference, or high electrode impedance, as well as physiological factors like eye blinks, muscle activity, or cardiac rhythms, which are particularly difficult to eliminate. Artefacts can distort EEG readings, potentially mimicking cognitive activity and leading to misinterpretations in clinical diagnoses, including conditions like Alzheimer's disease or sleep disorders. Therefore, ensuring effective artefact removal is crucial for maintaining the accuracy and reliability of EEG data in practical applications [JBT19]. The processing pipeline can be abridged as follows:

1. **Preprocessing** - Several preprocessing techniques are applied to improve signal quality. Filtering is the most well-known, representing the use of high-pass filters (> 0.5 Hz → removing slow drifts and baseline wander), low-pass filters (< 50 Hz → eliminating high-frequency noise, including muscle artifacts), or the notch filter (50/60 Hz → removing power line interference). Another technique is baseline correction, which compensates for slow drifts by subtracting the mean or performing detrending techniques. Epoching and

signal segmentation are also worth noting, as they are able to divide the EEG signal based on event-related time windows for further analysis.

2. **Feature Extraction** - Relevant features, such as Power Spectral Density (PSD), wavelet coefficients, entropy measures, and event-related potentials ERP, are extracted from the cleaned signal. These features provide critical information about brain activity, including frequency distributions, temporal variations, and connectivity patterns.
3. **Analysis and Classification** - Processed signals are analyzed using statistical, machine learning, or deep learning techniques. Statistical methods, such as Fourier and time-frequency analysis, help characterize EEG patterns. Machine learning approaches, including Support Vector Machines (SVMs), random forests, and deep learning models like convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are used for classification tasks, such as detecting cognitive states, sleep stages, or neurological disorders.

Regarding artefact removal techniques, various methods exist for removing artefacts while preserving brain activity. Back in the day, there was manual inspection and rejection. While effective, it was time-consuming and rather subjective. In '90s regression methods such as linear regression or adaptive filtering were used for the raw signal. These have been then replaced with **Blind Source Separation (BSS)** algorithms, including Principal Component Analysis (PCA)[4] and Independent Component Analysis (ICA)[5] [Jun+00]. Another group of suppression methods are wavelet-based methods which decompose the EEG signal into different frequency bands, allowing selective removal of artefacts. Lastly, recent advancements integrate machine learning and deep learning for automated artefact detection and removal. This involves classification-based artefact detection for supervised learning and clustering techniques to separate clean EEG for unsupervised learning [MPT21].

[4] PCA reduces dimensionality by retaining the principal components of the signal, which can help eliminate noise components.

[5] ICA decomposes signals into independent components, allowing researchers to identify and remove artefact sources.

Brain-Computer Interface

4

Brain-Computer Interfaces (BCIs) represent a groundbreaking field in neuroscience and neuroengineering, enabling direct communication between the brain and external devices. BCIs hold immense promise for individuals with motor impairments by providing new avenues for communication, mobility, and control over assistive devices [Wol+02]. They have applications in neurorehabilitation, assistive technology, and cognitive enhancement, aiming to restore lost functions and improve quality of life.

This chapter explores the fundamental principles of BCIs, their categorization, classification, signal acquisition methods, and processing techniques. Furthermore, it examines their applications in neurorehabilitation and related fields.

4.1 Fundamentals of BCI

A Brain-Computer Interface is a system that interprets brain activity and translates it into commands for external devices. These systems rely on neural signals that reflect cognitive or motor intentions, which are then processed to drive an output device such as a computer cursor, robotic arm, or speech synthesizer [BC07]. BCIs are primarily developed to study, enhance, support, or restore human cognitive and sensory-motor functions [Kru+16]. They are typically designed as direct human-machine interfaces that bypass physical movement, such as hand gestures. Additionally, BCIs raise intriguing possibilities regarding the integration of brain and machine, potentially blurring the boundaries between the two. Due to the cortical plasticity of the brain (Section 2.4), signals from implanted prostheses can, after adaptation, be handled by the brain like natural sensors or effector channels [Lev+00]. Following years of animal experimentation, the first neuroprosthetic devices were implanted in humans in the mid-1990s. The term *Neuroprosthetics*[1] is

[1] An area of neuroscience concerned with neural prostheses, that is, using artificial devices to replace the function of impaired nervous systems and brain-related problems, or of sensory or other organs (bladder, diaphragm, etc.)

sometimes used interchangeably with BCI, as both fields share the common goal of restoring functions such as vision, hearing, movement, communication, and cognition [Kru+16]. They also employ similar experimental approaches and surgical techniques in pursuit of these objectives.

The general architecture of a BCI is depicted in Figure 4.1 and consists of five key components:

- **Signal Acquisition** - This involves capturing neural activity using methods like EEG, Electrocorticography (ECoG), Magnetoencephalography (MEG), or Functional Near-Infrared Spectroscopy (fNIRS). Each technique, with its trade-offs, will be described more in Section 4.3.
- **Signal Processing** - The raw neural data undergo preprocessing, artefact removal, and feature extraction to identify patterns linked to user intentions.[2]
- **Translation Algorithms** - These computational models classify and interpret brain signals into meaningful commands using techniques like machine learning.
- **Output Devices** - Processed signals control external applications, such as prosthetic limbs, virtual reality interfaces, or communication aids.
- **User Feedback and Adaptation** - Many BCIs utilize adaptive algorithms that refine performance based on user interaction, ensuring improved accuracy and usability over time.

4.2 BCI Categorization

BCIs can be classified into three fields, based on their invasiveness. This classification is based on the target sense/area and location, and access to the brain.

Invasive BCIs involve surgically implanting electrodes beneath the scalp to directly access brain signals, offering the advantage of enhanced accuracy. However, this approach comes with risks, including potential surgical complications, the formation of scar tissue that may interfere with signal transmission, and the possibility of the body rejecting the implanted electrodes [AAM15]. Because they lie in the grey matter, invasive devices produce the highest quality signals of BCI devices. Invasive BCI research targets several areas. One of them targets repairing damaged sight and providing new functionality for people with paralysis. In vision science, direct brain implants have been used to treat non-congenital (acquired) blindness. Another area

[2] The signal processing is dependent on the chosen acquisition method. For this thesis, EEG will be used, which was already described in Section 3.4.

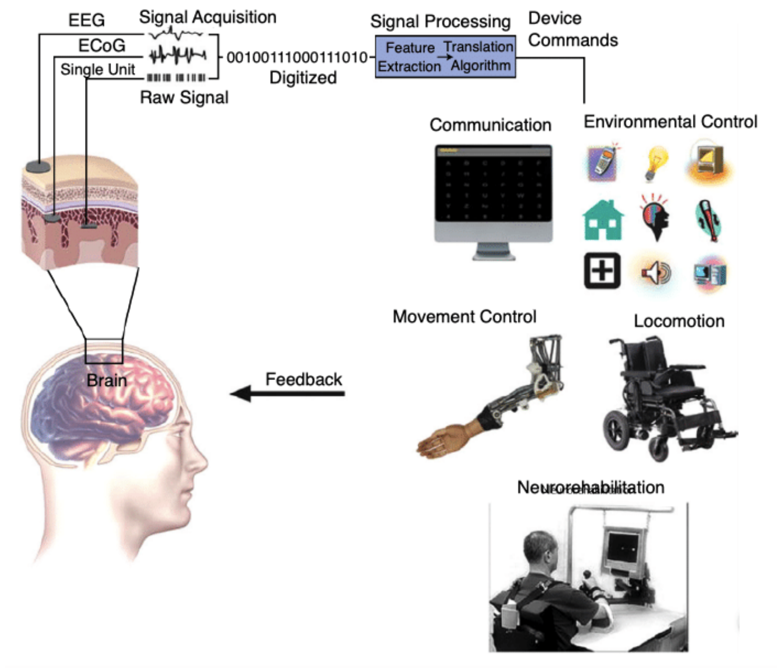


Figure 4.1: Components of BCI system and its communication (source: [KBA21])

is focusing on motor neuroprosthetics, aiming to restore movement in individuals with paralysis or provide devices to assist them, such as interfaces with computers or robot arms. Lastly, a recent breakthrough focuses on communication. In May 2021, a team from Stanford University demonstrated a proof-of-concept system that allowed a quadriplegic participant to generate English sentences at a rate of approximately 86 characters per minute or 18 words per minute. By imagining hand movements to write letters, the participant's brain activity was recorded from the motor cortex, and the system decoded the signals using Hidden Markov models and Recurrent Neural Network (RNN) for handwriting recognition [Wil+21].

Partially invasive BCI devices are implanted within the skull but positioned outside the brain, rather than directly in the grey matter. These devices offer higher-resolution signals compared to non-invasive BCIs, as they avoid signal distortion caused by the skull, while also posing a lower risk of scar tissue formation in the brain compared to fully invasive BCIs. Additionally, preclinical studies have demonstrated the potential of intracortical BCIs in utilizing signals from the perilesional cortex in stroke patients [Gul+15]. It is utilized in endovascular treatments through the use of a stentrode[3]. The stentrode transmits neural activity to a battery-free telemetry unit implanted in the chest, which wirelessly communicates with an ex-

[3] Stentrode is a monolithic stent electrode array designed to be delivered via an intravenous catheter under image-guidance to the superior sagittal sinus, in the region which lies adjacent to the motor cortex [Opi21].

ternal telemetry system for power and data transfer. While endovascular BCIs eliminate the need for a craniotomy, they carry risks such as clot formation and venous thrombosis.

Non-invasive BCI represents most of the human experiments. The majority of published BCI research involves non-invasive EEG-based BCIs. As mentioned in Section 3.1, although EEG-based BCIs are non-invasive, easy to wear, and do not require surgery, they have limitations such as low spatial resolution and reduced ability to detect higher-frequency signals due to signal distortion caused by the skull. Additionally, EEG-based systems often require setup time and calibration before each session, whereas some other BCI types function without prior training. One of the well-known BCI strategies that typically requires training to acquire acceptable control is *Motor Imagery*. Motor imagery refers to the mental simulation of body movements, which activates the sensorimotor cortex and alters sensorimotor oscillations in the EEG. BCIs can detect these changes to interpret user intent. However, training for motor imagery-based BCIs typically requires multiple hours over several days, and despite extended practice, users often struggle to fully master the control scheme.

4.3 Acquisition Techniques, Signal Processing and Translation

Accurate signal acquisition is crucial for BCI performance. Different neuroimaging modalities capture distinct neural activity features. Acquisition of the EEG signal is already described in great detail in Section 3.2.2 and therefore will be skipped in this section.

Electrocorticography (ECoG) measures brain electrical activity from beneath the skull in a way similar to non-invasive electroencephalography, using electrodes embedded in a thin plastic pad placed above the cortex, beneath the dura mater [SD04] and therefore belongs to partially invasive BCIs. ECoG provides several advantages over scalp-recorded EEG, including higher spatial resolution, an improved signal-to-noise ratio, a broader frequency range, and reduced training requirements. At the same time, it poses fewer technical challenges, carries lower clinical risks, and may offer greater long-term stability compared to intracortical single-neuron recordings [NG12]. These characteristics, along with evidence of high-level control with minimal training, suggest strong potential for real-world applications, particularly for individuals with motor disabilities.

Magnetoencephalography (MEG) is a functional neuroimaging technique that maps brain activity by detecting the magnetic fields generated by natural electrical currents in the brain, using highly sensitive magnetometers. Although EEG and MEG both capture brain activity from the same underlying neurophysiological processes, they have key differences [CC83]. Magnetic fields experience less distortion from the skull and scalp than electric fields, giving MEG superior spatial resolution. While EEG detects both tangential and radial components of current sources in a spherical volume conductor, MEG is sensitive only to tangential components. Additionally, EEG primarily records extracellular volume currents generated by postsynaptic potentials, whereas MEG primarily detects intracellular currents associated with these synaptic potentials, as volume currents tend to cancel out in a spherical conductor [BSB86]. Unfortunately, MEG requires specialized, expensive equipment, making it a less viable option in comparison to EEG.

Functional magnetic resonance imaging fMRI measures brain activity by detecting changes in blood flow, based on the principle that neuronal activation and cerebral blood flow are closely linked. When a specific brain region is active, blood flow to that area increases accordingly [Log+01]. The most common form of fMRI utilizes Blood-Oxygen-Level Dependent (BOLD) contrast. This hemodynamic response reflects the energy consumption of brain cells, providing insights into functional brain activity in humans and other animals. fMRI and MEG have both been used as non-invasive BCIs.

Extracting meaningful patterns from data involves identifying biomarkers.

An **Event-Related Potential (ERP)** is a brain response directly linked to a specific sensory, cognitive, or motor event. More precisely, it is a consistent electrophysiological reaction to a given stimulus, measurable through EEG or MEG[4]. Compared to behavioural methods, ERPs offer a continuous measure of brain activity between a stimulus and a response, allowing researchers to pinpoint which processing stages are influenced by experimental manipulations. A key advantage of ERPs is their ability to track stimulus processing even in the absence of observable behavioural changes. However, due to their small signal amplitude, accurately measuring ERPs typically requires a large number of trials [Luc05].

Motor imagery was already mentioned in Section 4.2. By definition, motor imagery can be defined as a dynamic state during which an individual mentally simulates a physical action. This type of phenomenal experience implies that the subject feels themselves performing the action [Dec96]. Motor imagery activates motor pathways. Muscular activity often increases with rest, during motor imagery. Measurements of cardiac and respiratory activity during motor imagery and during actual motor performance revealed a covariation of heart rate and pulmonary ventilation with the degree of imagined effort [Dec+93; WM92]. Algorithms such as Common Spatial Pattern (CSP) or Filter Bank Common Spatial Pattern (FBCSP) have been deployed for feature extraction of motor imagery [LC23].

[4] Then we are talking about an Event-Related Field (ERF) which is equivalent to ERP.

Visual Evoked Potential (VEP)[5] is an electrical potential recorded after a subject is presented with visual stimuli. The most well-known types of VEPs include SSVEPs and P300 potentials. **Steady-State Visually Evoked Potential (SSVEP)** are typically elicited using alternating checkerboard patterns or flashing images, with the stimulus's phase reversal frequency distinguishable through EEG. The SSVEP signal is strong due to the topographic organization of the primary visual cortex, where a large cortical area processes input from the central or foveal visual field. However, SSVEP-based BCIs present challenges. Since they rely on flashing stimuli to detect user intent, users must maintain their gaze on specific flashing or oscillating symbols to interact with the system. Over extended periods, this can lead to discomfort and irritation, potentially reducing usability. **P300 potential** is a positive peak in the EEG that appears approximately 300 milliseconds after the presentation of a target stimulus—a stimulus the user is anticipating or searching for—within an oddball paradigm. The amplitude of the P300 decreases as the distinction between target and non-target stimuli becomes less pronounced. This potential is believed to be associated with higher-level attentional processes or an orienting response.

Translation algorithms are based on the acquisition technique. For EEG, these computational models used for classification and brain interpretation were briefly mentioned in Section 3.4. One worth noting for its simplicity and efficiency is **Linear Discriminant Analysis (LDA)**. LDA seeks to find a linear combination of features that best separates two or more classes by maximizing the between-class variance while minimizing the within-class variance [DHS01]. Multiclass LDA can be implemented; however, in the field of neurorehabilitation, its binary version is more commonly utilized.[6].

4.4 Applications of BCIs in Neurorehabilitation

BCIs have revolutionized neurorehabilitation by offering individuals with motor impairments alternative pathways for recovery and interaction with their environment. By harnessing brain activity to control external devices, BCIs can significantly enhance mobility, communication, and independence in patients with neurological conditions.

Motor recovery BCIs are designed to support motor neuroprosthetics, either by restoring movement in individuals with paralysis or by providing assistive de-

[5] Note that evoked potential is different from Event-Related Potential (ERP), although the terms are sometimes used synonymously.

[6] Mainly in motor imagery left-right movement [LC23].

vices. Research in recent years has demonstrated the utility of EEG-based BCI systems in aiding motor recovery and neurorehabilitation in patients who have had a stroke [Sil+11; Lea+14; Tun+13]. In this approach, a BCI monitors motor activity as the patient either imagines or attempts movements under the guidance of a therapist. This method offers two key benefits: (1) if the BCI detects that the patient is not correctly imagining the movement (non-compliance), it can provide feedback to both the patient and therapist, and (2) rewarding feedback, such as functional stimulation or the movement of a virtual avatar, is contingent on the patient accurately performing motor imagery. Imaging studies combined with EEG-based BCI systems hold promise for investigating neuroplasticity during motor recovery post-stroke [Mra+17].

Another area concerns **communication abilities**. For patients suffering from Locked-in Syndrome (LIS), where voluntary muscle movement is nearly impossible, BCIs provide a vital means of communication. Birbaumer et al. ([Bir+99]) pioneered early research demonstrating that ALS patients could use Slow Cortical Potentials (SCPs) to communicate via a BCI system. More recent advancements have improved the accuracy and usability of such interfaces, enabling LIS patients to type words and sentences using event-related potentials (ERPs) or steady-state visually evoked potentials SSVEPs [Mak+11]. The P300 speller, for example, allows patients to select letters on a screen based on their brain responses to flashing stimuli, greatly enhancing their ability to communicate [SD06].

Assistive Technologies BCIs are increasingly being integrated into assistive devices such as wheelchairs, prosthetics, and smart home systems. Hochberg et al. [Hoc+12] conducted a landmark study with the **BrainGate** system, in which tetraplegic individuals controlled robotic arms using intracortical BCI implants. This study demonstrated the feasibility of direct brain control over complex assistive technologies, paving the way for further innovation. In addition, non-invasive BCIs are being used to operate wheelchairs, allowing individuals with severe mobility impairments to navigate their environments by controlling directional commands with brain activity [IMM10]. Similarly, smart home applications have been developed where users can adjust lighting, temperature, and electronic devices through BCI interfaces, further enhancing independence for individuals with disabilities.

Current State Of Art 5

Before presenting a solution to the problem addressed in this thesis and implementing it, it is essential to first examine the current state of the art in the field. Over the past decade, researchers have explored various techniques and rehabilitation scenarios aimed at improving motor outcomes in patients with stroke, spinal cord injuries, and other motor impairments. This chapter is focused on the neurorehabilitation research area (mostly motor recovery) and its current state.

For starters, Cantillo-Negrete et al. ([Can+23]) provide a practical guide for implementing Brain-Computer Interface (BCI)-based interventions for stroke neurorehabilitation, specifically targeting upper extremity motor recovery. The guide addresses challenges in BCI clinical applications, ensuring usability regardless of infrastructure or study design limitations. The review synthesises insights from administering hundreds of BCI rehabilitation sessions, offering a structured approach to executing BCI-based stroke interventions effectively. The article acts as a structured guide to help researchers and clinicians effectively implement BCI therapies, covering technical requirements, session design, and intervention parameters. By offering practical insights from real-world BCI applications, the guide bridges the gap between theoretical research and clinical implementation, making BCI-based rehabilitation more accessible and effective. The article concludes with standardization as a key challenge in (BCI)-based interventions for neurorehabilitation.

Cioffi et al. ([Cio+24]) provide a scoping review examining EEG-based sensorimotor neurofeedback as a neurorehabilitation tool for individuals with neurological motor impairments, including children and adults. A total of 4,380 studies were initially identified, with 133 meeting the criteria for final analysis. Notably, only three studies focused on children, underscoring a significant research gap. Additionally, 77% of the studies concentrated on adult stroke patients, while fewer explored other movement disorders. The outcomes of the studies varied significantly and encompassed BCI classification accuracy, Mu rhythm modulation (enhancements in neurophysiological parameters), and clinical/motor outcome improvements. However, few studies explicitly established a direct connection between functional motor improvements and EEG-based neurofeedback, leaving it unclear whether the observed changes were directly attributable to the neurofeedback intervention. In the end, the study highlighted the need for improved study design, methodological consistency, and direct clinical validation of neurofeedback effects to ensure its future application in evidence-based neurorehabilitation.

Yu Tung Lo et al. ([Lo+24]) conducted an Individual Patient Data (IPD) meta-analysis to evaluate the effectiveness of neural interface-based neurorehabilitation for poststroke upper limb recovery. The analysis focused on Brain-Computer Interface (BCI) and Brain-Machine Interface (BMI), which control Functional Electrical Stimulation (FES) and powered exoskeletons to facilitate motor rehabilitation. The study also examined clinical parameters influencing recovery outcomes. A conclusion was drawn that neural interface-based motor rehabilitation significantly reduces poststroke impairment, though the improvements are modest. Motor attempt-based training and longer rehabilitation durations (>4 weeks) yield better results. While younger patients and those with less severe impairment tend to improve more, age and severity do not completely preclude meaningful recovery.

Many neurorehabilitation sessions use different types of games to provide more enjoyable and engaging therapies for the patients. In recent years, a sudden rise of Virtual Reality (VR) in neurorehabilitation came in the form of research articles and experiments. Specht et al. ([Spe+23]) investigated the effectiveness of cognitive training using Head-Mounted Display (HMD) VR in neurorehabilitation. It was a pilot Randomized Controlled Trial (RTC) aimed at evaluating the impact of VR-based cognitive training on cognitive function and user engagement compared to conventional training methods. The research focused on patients with neurological conditions that affect cognitive functions, assessing whether VR-based training provides better cognitive improvements and adherence rates. The study suggested that VR-based cognitive training is an effective and engaging alternative to traditional cognitive rehabilitation methods. It has the potential to improve cognitive functions in patients with neurological impairments while also increasing adherence to therapy. However, further large-scale studies were recommended.

A very similar study was performed by Prats-Bisbe et al. ([Pra+24]) regarding VR as a support tool for people with cognitive impairments. The primary goal was to assess whether VR can serve as a practical and effective rehabilitation tool for patients experiencing cognitive deficits due to conditions such as stroke or Traumatic Brain Injury (TBI). The results were similar to the ones presented by Specht et al., and that is that VR-based neurorehabilitation is a promising and engaging approach for individuals with acquired brain injuries. While usability and feasibility were largely positive, challenges such as user adaptation and mild discomfort need to be addressed. The findings support continued exploration and refinement of VR interventions to improve cognitive rehabilitation outcomes. This claim also supports Blázquez-González et al. ([Blá+24]) with the study on *Efficacy of Virtual Re-*

ality on Health Literacy in Patients with Stroke. Dividing the patients into two groups (the first had conventional rehabilitation only, and the second added a 20-minute VR session), VR-based neurorehabilitation showed the potential to improve health literacy, however, the study did not find statistically significant evidence to confirm its effectiveness.

O'Neil et al. ([ONe+18]) approached the issue from a more practical point of view. The study explored the real-world application of virtual reality in neurorehabilitation by analyzing three different European clinics that implemented VR for various patient groups and settings. It highlighted the motivations, implementation challenges, and observed outcomes of using VR in Parkinson's disease rehabilitation, pediatric neurorehabilitation, and home-based telerehabilitation. Practical insights were deducted from the study, mainly while VR improves patient engagement and accessibility, its clinical effectiveness is still under evaluation, and costs, technical challenges, and implementation variability remain barriers to widespread adoption. Particular potential was found in telerehabilitation and gamified therapy models.

A brand new study from Sánchez-Gil et al. ([Sán+25]) introduced and validated PACTUS, a gamified electronic device for stroke rehabilitation. The device is designed to improve cognitive and motor function in the upper limbs of post-stroke patients, addressing barriers to rehabilitation such as economic, emotional, and social constraints. The goal of PACTUS is to increase patient motivation and provide precise rehabilitation progress tracking for both therapists and patients. The study highlighted its broad potential application across different neurological conditions, but further research and refinements are needed before full clinical deployment.

Proposed BCI system 6

The task of this thesis is to create a closed BCI loop for motor imagery rehabilitation of the upper extremity with the help of a rehabilitation robot situated in the neurolaboratory at the University of West Bohemia. The system will form a closed-loop pipeline that integrates real-time BCI processing, robotic-assisted rehabilitation, and interactive feedback through a Unity-based front-end. The abstract system architecture is portrayed in Figure 6.1.

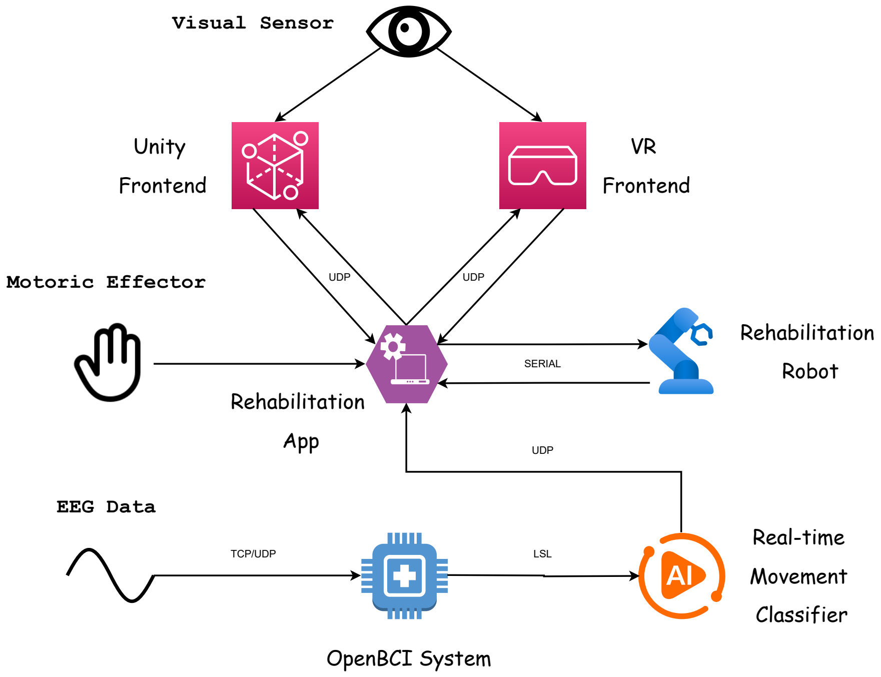


Figure 6.1: Diagram of proposed BCI system architecture

At the core of the loop will be the **OpenBCI** EEG system, which will continuously acquire neural signals from motor-related cortical areas. These signals will be streamed in real time via the Lab Streaming Layer (LSL) to a custom-built Python classification pipeline that distinguishes between resting states and movement attempts. The classifier's predictions will be transmitted via User Datagram Protocol (UDP) to the rehabilitation control application, which interfaces with the robotic rehabilitation device using a serial port connection. These predictions will signal the concentration of the patient. Based on it, the robotic arm will help the patient rehabilitate by providing more mechanical support, making it easier to move. In parallel, the rehabilitation application will communicate bidirectionally with a Unity-based graphical front-end that will visualise the patient's performance, providing immediate neurofeedback. A VR-based variant of the front-end will also be developed as a part of **KIV/OP**[1] to extend this concept into an immersive environment.

6.1 Expectations

A few expectations preceding this proposed system need to be mentioned. The first anticipation is regarding the rehabilitation software that guides the robotic arm. In order to successfully create a closed loop, the application needs to be able to communicate with the **Unity** front-end the position of the arm and potentially other settings such as the current state of the machine, rehabilitation curve properties, etc.. In another vein, the rehabilitation software needs to be able to receive movement predictions and, based on the value, adjust the movement of the servomotors. The values can be presented either as binary values or represented as a scale.

The second expectation is about the binary classifier. Since creating the classifier is not directly part of the thesis assignment, it is proposed to use other available classifiers. Several works at the Faculty of Applied Sciences at UWB cover this matter. Kodera created and tested multiple neural network structures with different augmentation techniques to analyze the performance and accuracy of various classifiers in his master's thesis [Kod23]. In his work, Kodera used an open-source dataset consisting of 11 trials from 29 healthy subjects aged 19 to 24 [Kod+23] [2]. The data correspond to 9 electrodes in the officially recognized Brain Vision data format. To evaluate the classification performance of each model, Kodera employed **10-fold cross-validation** and tested the classifiers across three different data representation domains: time series, frequency, and time-frequency. Analysis of the results evaluated the Convolutional Neural Network (CNN) without any augmentations as the best model with an accuracy of **76.00 ± 0.80%** [Kod23]. It is worth noting that Kodera **did not test the trained models in a real-time scenario**, and that the results he presents are supported only by a **cross-validation** performed on the mentioned dataset.

[1] This is more intensively described in Section 7.3.

[2] It needs to be noted that the measuring setup and scenario differ from this thesis.

Duc Thien Pham followed up by creating a hybrid version of his deep learning model called *CNN-Transformer-LSTM* [PM24]. Pham used the same MI dataset as Kodera, but created a more sophisticated and complex structure. The model yielded similar results to Kodera's, increasing its accuracy to **78.51 ± 1.21%**. Both studies concluded that augmenting EEG data provided no significant benefits. In summary, it is proposed to use one of these models and apply it to the proposed pipeline.

System Implementation

7

7.1 Hardware and Software Setup

Before diving into the actual implementation of the system parts, examinations of the third-party software setups are needed in order to better understand the entire flow of the system. The section will begin with a description and the hardware/software setup of the data acquisition application, followed by a description and setup of the rehabilitation software used in this thesis.

7.1.1 OpenBCI

OpenBCI is an open-source platform designed for biosensing applications, particularly EEG. The platform includes hardware components like the Cyton board, which features eight channels for EEG signal acquisition and utilises the Texas Instruments ADS1299 analog-to-digital converter, known for its high-resolution biopotential measurements [Ope25]. The Cyton board supports data transmission via Bluetooth or Wi-Fi, through Wi-Fi shield (TCP/UDP), facilitating real-time data streaming. Its Graphical User Interface (GUI) provides widgets such as a time series channel-wise filtered data, FFT plots, a bandpass widget, spectogram and a lot more.

7.1.1.1 Hardware setup

OpenBCI provides hardware kits that contain the necessary equipment for measuring EEG activity through their software application. The Gelfree BCI Electrode Cap Kit was used in this thesis and includes the following:

- Stretchy cap with adjustable chinstrap
- Labelled Ag-AgCl electrodes/cables
- Measuring utensils for saline conductive solution

- Saline retaining Hydro-link conductive inserts
- Cotton swabs for scalp preparation
- Towel for mess-free clean-up
- Cyton Biosensing Board (8-channels) with Wi-fi shield

The Cyton board is powered by a four-A4 battery power source. The 8-channel system was utilised, with 5 channels designated for data collection (precisely **Fp1**, **Fp2**, **Cz**, **C3**, **C4**) and two electrodes serving as ground electrode (**GND**) and reference electrode (**REF**). The electrode setup is displayed in Figure 7.1.

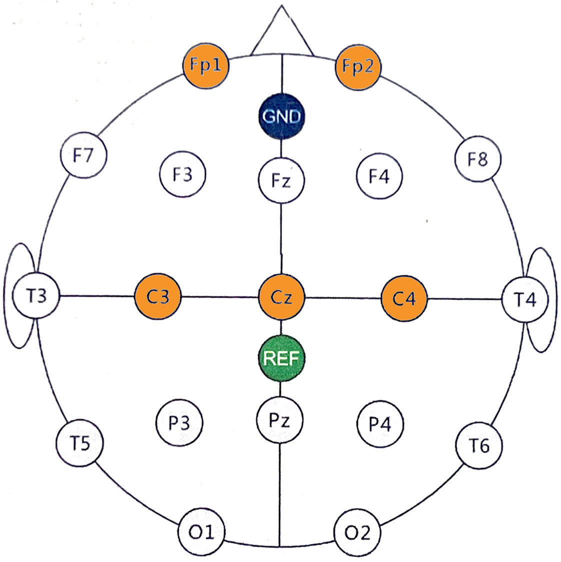


Figure 7.1: **OpenBCI** Gelfree electrode cap setup

Before attaching the electrodes to the designated cap slots, the cotton swab and Hydro-link inserts need to be immersed in a saline conductive substance for 5-10 minutes in order to gain the conductive properties necessary for the measurement. After that, each slot highlighted in Diagram 7.1 needs to be moistened with the cotton swab. It is important to note that each electrode cable needs to be attached to its assigned slot. The chosen mapping is shown in Table 7.1.

Table 7.1: Mapping of EEG electrodes to cable numbers

Electrode	Fp1	Fp2	Cz	C3	C4	REF	GND
Cable No.	1	2	3	4	5	7	8

The number of the electrode cable is noted around the plastic grip of the Hydro-link insert area. The connection of the conductive Hydro-link and skin needs to be secured [1].

7.1.1.2 Software setup

The connection to the hardware in this thesis was executed through the mentioned Cyton board in Wi-Fi mode. To successfully connect to the board and receive the data, the PC needs to be connected to a Wi-fi created by the Wi-fi shield on the board [2]. As mentioned in Section 7.1.1.1, an 8-channel system with a sample rate of **1000Hz** was chosen through a static IP of the board.

For the time series widget, the last three channels can be filtered out for improved readability. **OpenBCI** applies basic filtering (Notch filter with bandpass 5-50Hz) before rendering the signals from the channels, but for better motor signal recognition, using filter settings described in section 7.6.2 is advised.

The networking widget serves for streaming the data out of the **OpenBCI** application, in this thesis, to the real-time binary classifier (movement/resting). The widget provides up to three different streams at any moment, but for the purposes of this thesis, one is satisfactory. The type of data sent needs to be **filtered** so that the prediction script receives correct data. **OpenBCI** provides four communication protocols, from which LSL was selected. The reason behind this choice is that LSL was designed for the unified collection of measurement time series in research experiments that handles both the networking, time-synchronization, (near-) real-time access [KM19].

OpenBCI also provides recordings of the conducted experiment sessions. The data from these sessions is locally saved as Comma-Separated Values (CSV) or text files. The character and the format of these files are further described in section 7.4.

7.1.2 FourMotors software

The **FourMotors** rehabilitation software is an external application designed to control and monitor robot-assisted therapy sessions. Although not officially documented, its functionality can be inferred through source code analysis. It serves as a middleware layer between the robotic hardware and external control or visualization systems, such as the **Unity** front-end and EEG classification modules shown in Diagram 6.1.

At its core, **FourMotors** is a Node.js application that uses `Electron` to render a full-screen graphical interface, which launches a local server and opens a browser window to serve the rehabilitation interface. This window visualizes the current

[1] This is achieved by turning the plastic grip clockwise in the electrode slot until it produces a clicking sound.

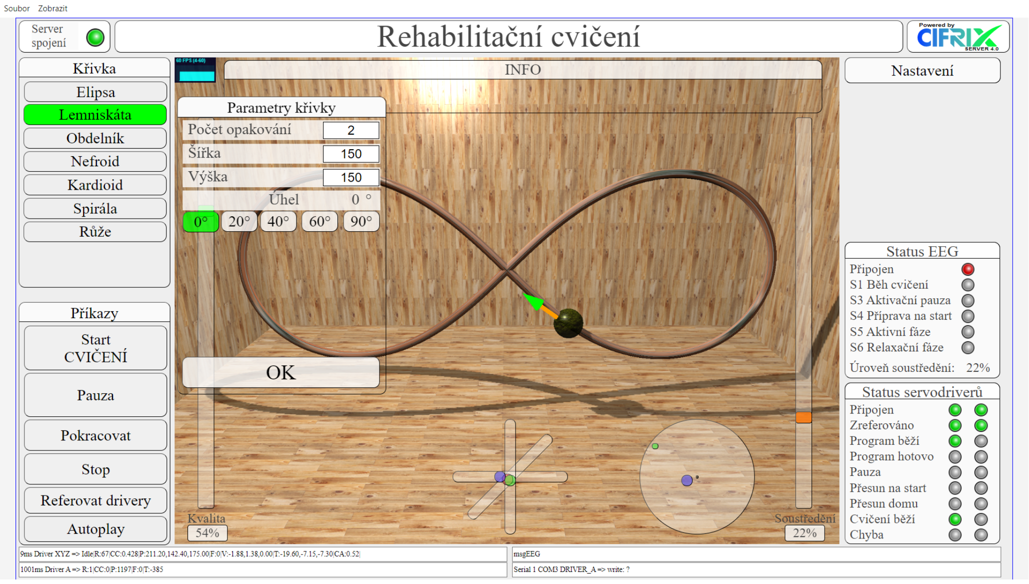
[2] The Wi-fi does not have internet access and its sole purpose is data transfer.

robot status, curve parameters, and supports operator input. Communication within the system is distributed across two primary channels:

- **Serial Communication with the Robot** - Through serial commands, it controls different robot drivers (e.g., DRIVER_XYZ, DRIVER_A) and reads back detailed telemetry such as position, velocity, force, and execution state. Commands like program start, pause, stop, reset, and home positioning are sent directly to the robot.

- **UDP Communication with External Clients** - The software runs a UDP server listening on port 4000. It handles JSON-based requests for dynamic robot parameters (e.g., curve parameters, current position data), enabling bidirectional communication with clients such as the **Unity** front-end or external control systems (detailed description is given in section 7.2).

The rehabilitation flow consists of movements in which the robotic arm assists in tracing seven predefined curves—ellipse, lemniscate, rectangle, nephroid, cardioid, spiral, and rose—each configurable with settings such as number of repetitions, width, height, and angle. The curve information and modifications cannot be configured externally (via UDP or other protocol), but can be requested by the front-end part of the system. A visual showcase of the rehabilitation application is displayed in Figure 7.2



Parametry křivky

Počet opakování 2

Šířka 150

Výška 150

Úhel 0 °

Kvalita 54%

Soustředění 22%

Status EEG: Připojen, S1 Běh cvičení, S3 Aktivační pauza, S4 Příprava na start, S5 Aktivní fáze, S6 Relaxační fáze

Úroveň soustředění: 22%

Status servodriverů: Připojen, Zreferováno, Program běží, Program hotovo, Pauza, Přesun na start, Přesun domu, Cvičení běží, Chyba

```
9ms Driver XYZ => Idle|R:67|CC:0.428|P:211.20,142.40,175.00|F:0|V:-1.88,1.38,0.00|T:-19.60,-7.15,-7.30|CA:0.52|
1001ms Driver A => R:1|CC:0|P:1197|F:0|T:-385
msgEEG
Serial 1 COM3 DRIVER_A => write: ?
```

Figure 7.2: **FourMotors** UI representation

7.2 Unity Front-end

The **Unity** Front-end serves as the main visual point of the rehabilitation. It must be, therefore, crafted and adjusted for the patients who are recovering from conditions mentioned in Section 2.5. This section describes the visual and practical design of the rehabilitation, followed by code implementation of the visual functionality and communication with the robotic arm device.

7.2.1 Environment

The environment plays a crucial role in rehabilitation scenarios. Patients spend countless hours rehabilitating, and the need for a pleasant, calm, and soothing environment is evident. With support from colleagues at the Sutnar Faculty of Design and Art, who contributed the models, a dojo inspired by traditional Japanese architecture was created. The main design was to create a closed space for two reasons: to save computational resources that would be otherwise needed for rendering and to create a cosy feeling for the patient. The room provides multiple dim lights with different warm colors to support the feeling of safety. For optimisation purposes (which are crucial for the development of a VR application with limited resources - Section 7.3), all the lights are static and baked [3]. The colour palette contains mostly earthly, faint colours, further enforcing the tranquillity of the environment. The complete room is displayed in Figure 7.3. The environment consists of approximately 260,000 vertices.

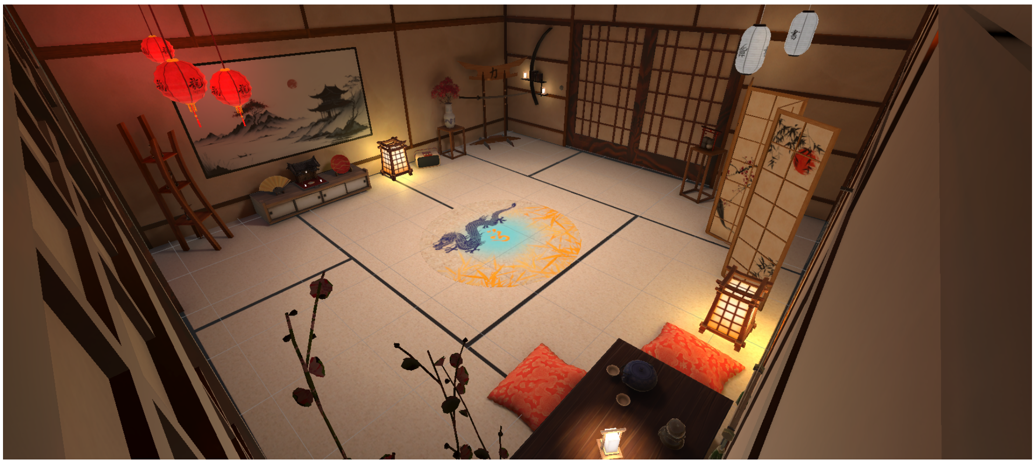


Figure 7.3: Rehabilitation environment in **Unity**

[3] Baked lighting refers to a technique where lighting information, such as shadows and light bounce, is precomputed and stored in textures or lightmaps. This reduces the real-time rendering workload and improves performance.

7.2.2 Scenario

Various scenarios were taken into account concerning the chosen environment and the rehabilitation flow provided by the **FourMotors** application. Since the rehabilitation curves are predefined, the primary object needs to stay the same, but the visual representation may differ. One idea was, for example, to rehabilitate with the robotic arm visualized as a katana and create either just movements as a form of martial art meditation or to prepare some sort of breakable objectives on the curve's path so that the patient needs to cut through them. This idea was later discarded because of the visual complexity.

As the final scenario, painting on a canvas was chosen. Painting is for many people a sort of relaxation, and it supports the basics of rehabilitation regulations. It also does not require knowing the curve details in advance, and it is therefore modular, allowing future features to be added without the need to change currently working code [4]. It also introduces features that can enhance the user experience or make the otherwise repeating exercises fresher and new (colour picking, brush type, brush thickness, etc.). Both **Unity** and VR front-ends support the same functionality regarding rehabilitation as **FourMotors** software. A more detailed description in this regard can be found either in Section 7.4 or in Appendix A. A snippet of a part of the scenario run is showcased in Figure 7.4.

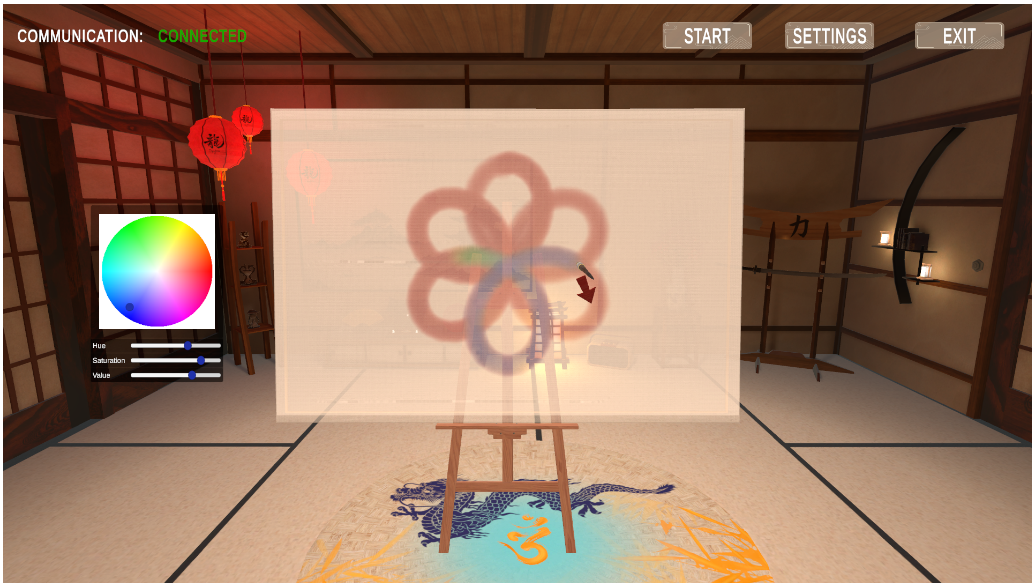


Figure 7.4: A showcase of the **Unity** front-end

[4] This applies to both the **Unity** front-end and to the rehabilitation software.

7.2.3 Implementation

The implementation of the **Unity** front-end is structured in a modular and scalable manner [5]. The application is developed using the **Unity** engine (C#), which allows for more refined game and even VR development. The entire implementation diagram is portrayed in Figure 7.5. Both PC and VR versions were developed in **Unity** editor version **2022.3.21f**.

7.2.3.1 Architecture Overview

The front-end architecture is component-based and revolves around the central class `RehabilitationManager`, which coordinates the flow of rehabilitation sessions. It handles the scenario logic, manages user settings (`RehabilitationSettings`), collects movement data (`RehabData`), and interfaces with the external robot control system via UDP (`UdpConnection`).

The visual rehabilitation metaphor, as already mentioned, is painting on a virtual canvas. A paintable surface (`PaintableCanvas`) responds to input from a "virtual brush", which can be dynamically customized via different `BrushTip` settings (e.g., size, color, texture). These brushes are implemented using a base class `BaseBrush`, allowing the system to support extensibility for new brush types. The canvas is represented by a texture. Whenever the tip of the Brush touches the canvas texture (this is detected by a Raycast), it calculates the UV coordinates of the texture and applies them as follows:

1. The method converts UV coordinates into pixel coordinates to identify the center point for the brush stroke on the canvas.

2. It determines the scaled dimensions of the brush based on a configurable scale factor.

3. It iterates over the scaled brush area, checking boundaries to avoid drawing outside the canvas.

4. For each valid pixel, it samples the brush texture using UV coordinates and retrieves the corresponding canvas pixel.

5. A blending operation is applied between the brush and canvas pixels, modulated by brush color and pressure.

6. The modified pixel is written back, and all updates are applied to the texture at once.

[5] These properties can be seen in Section 7.3 where most of the implementation is the same and the only adjustments are device specific.

7.2.3.2 Scenario Control and Data Flow

The scenario is initiated through the `RehabilitationManager`. The script serves as a state machine and controls the scenario run (more information in Section 7.4). During the session, the **Unity** application receives positional data from the **FourMotors** software via UDP. The communication occurs on port **4000**. To retrieve data from the server, `{"get": "data"}` is sent; to obtain curve parameters, `{"get": "curve"}` is sent. The data is received as a JSON with the following structure:

- `report`: program state
- `curveCycle`: number of completed movement cycles (floating-point, includes fractions)
- `pos`: position of the hand
- `vector`: movement direction vector
- `rotation`: hand rotation
- `thrustForce`: applied hand force
- `cosAngleVector`: directional angle (cosine)
- `cnt`: counter of sent messages

And for the curve data:

- `repeat`: number of repetitions
- `width`, `height`: width and height of the curve
- `angle`: tilt angle of the curve (in radians)
- `paramN`, `paramD`: parameters for the rose curve (type 7)
- `type`: curve type (1 – ellipse, 2 – lemniscate, 3 – rectangle, 4 – nephroid, 5 – cardioid, 6 – spiral, 7 – rose)

This data is processed and visualized in real-time, with a `DirectionArrow3D` object indicating target directions or assistance cues for the user.

7.2.3.3 Customization and Interaction

User feedback and engagement are enhanced through a custom-built UI system. A `UI_ColourPicker`, built from `UI_ColourWheel` and `UI_ColourPickerComponent`, allows patients to select brush colors in an intuitive way. This interaction adds gamified elements that help sustain motivation and engagement during repetitive tasks.

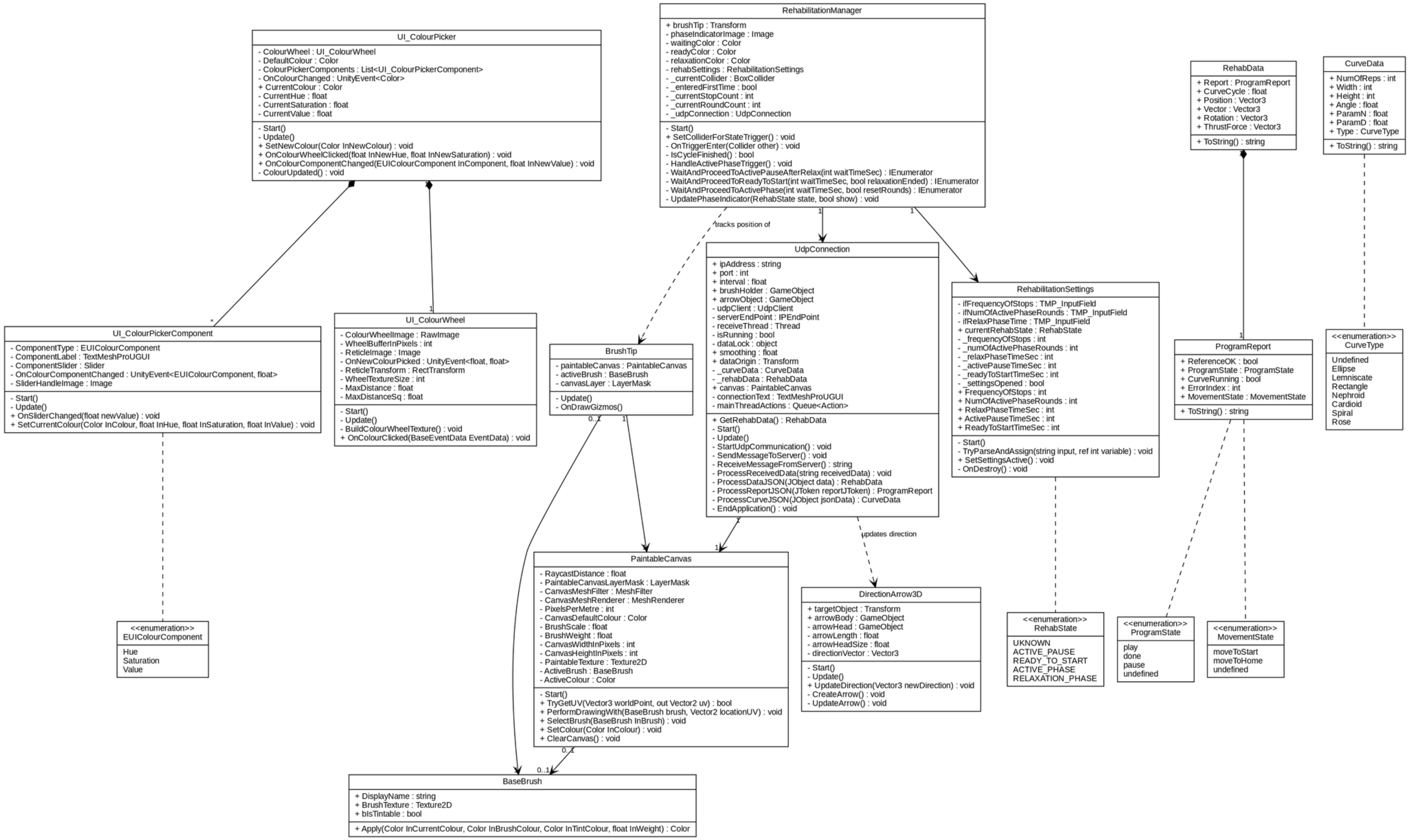

UI_ColourPicker
- ColourWheel : UI_ColourWheel
- DefaultColour : Color
- ColourPickerComponents : List<UI_ColourPickerComponent>
- OnColourChanged : UnityEvent<Color>
+ CurrentColour : Color
- CurrentHue : float
- CurrentSaturation : float
- CurrentValue : float
- Start()
- Update()
+ SetNewColour(Color InNewColour) : void
+ OnColourWheelClicked(float InNewHue, float InNewSaturation) : void
+ OnColourComponentChanged(EUIColourComponent InComponent, float InNewValue) : void
- ColourUpdated() : void
RehabilitationManager
+ brushTip : Transform
- phaseIndicatorImage : Image
- waitingColor : Color
- readyColor : Color
- relaxationColor : Color
- rehabSettings : RehabilitationSettings
- _currentCollider : BoxCollider
- _enteredFirstTime : bool
- _currentStopCount : int
- _currentRoundCount : int
- _udpConnection : UdpConnection
- Start()
+ SetColliderForStateTrigger() : void
- OnTriggerEnter(Collider other) : void
- IsCycleFinished() : bool
- HandleActivePhaseTrigger() : void
- WaitAndProceedToActivePauseAfterRelax(int waitTimeSec) : IEnumerator
- WaitAndProceedToReadyToStart(int waitTimeSec, bool relaxationEnded) : IEnumerator
- WaitAndProceedToActivePhase(int waitTimeSec, bool resetRounds) : IEnumerator
- UpdatePhaseIndicator(RehabState state, bool show) : void
RehabData
+ Report : ProgramReport
+ CurveCycle : float
+ Position : Vector3
+ Vector : Vector3
+ Rotation : Vector3
+ ThrustForce : Vector3
+ ToString() : string
CurveData
+ NumOfReps : int
+ Width : int
+ Height : int
+ Angle : float
+ ParamN : float
+ ParamD : float
+ Type : CurveType
+ ToString() : string
tracks position of
1
1
1
1
*
1
UdpConnection
+ ipAddress : string
+ port : int
+ interval : float
+ brushHolder : GameObject
+ arrowObject : GameObject
- udpClient : UdpClient
- serverEndPoint : IPEndPoint
- receiveThread : Thread
- isRunning : bool
- dataLock : object
+ smoothing : float
+ dataOrigin : Transform
- _curveData : CurveData
- _rehabData : RehabData
+ canvas : PaintableCanvas
- connectionText : TextMeshProUGUI
- mainThreadActions : Queue<Action>
+ GetRehabData() : RehabData
- Start()
- Update()
- StartUdpCommunication() : void
- SendMessageToServer() : void
- ReceiveMessageFromServer() : string
- ProcessReceivedData(string receivedData) : void
- ProcessDataJSON(JObject data) : RehabData
- ProcessReportJSON(JToken reportJToken) : ProgramReport
- ProcessCurveJSON(JObject jsonData) : CurveData
- EndApplication() : void
RehabilitationSettings
- ifFrequencyOfStops : TMP_InputField
- ifNumOfActivePhaseRounds : TMP_InputField
- ifRelaxPhaseTime : TMP_InputField
+ currentRehabState : RehabState
- _frequencyOfStops : int
- _numOfActivePhaseRounds : int
- _relaxPhaseTimeSec : int
- _activePauseTimeSec : int
- _readyToStartTimeSec : int
- _settingsOpened : bool
+ FrequencyOfStops : int
+ NumOfActivePhaseRounds : int
+ RelaxPhaseTimeSec : int
+ ActivePauseTimeSec : int
+ ReadyToStartTimeSec : int
- Start()
- TryParseAndAssign(string input, ref int variable) : void
+ SetSettingsActive() : void
- OnDestroy() : void
1
ProgramReport
+ ReferenceOK : bool
+ ProgramState : ProgramState
+ CurveRunning : bool
+ ErrorIndex : int
+ MovementState : MovementState
+ ToString() : string
<<enumeration>>
CurveType
Undefined
Ellipse
Lemniscate
Rectangle
Nephroid
Cardioid
Spiral
Rose
UI_ColourPickerComponent
- ComponentType : EUIColourComponent
- ComponentLabel : TextMeshProUGUI
- ComponentSlider : Slider
- OnColourComponentChanged : UnityEvent<EUIColourComponent, float>
- SliderHandleImage : Image
- Start()
- Update()
+ OnSliderChanged(float newValue) : void
+ SetCurrentColour(Color InColour, float InHue, float InSaturation, float InValue) : void
UI_ColourWheel
- ColourWheelImage : RawImage
- WheelBufferInPixels : int
- ReticleImage : Image
- OnNewColourPicked : UnityEvent<float, float>
- ReticleTransform : RectTransform
- WheelTextureSize : int
- MaxDistance : float
- MaxDistanceSq : float
- Start()
- Update()
- BuildColourWheelTexture() : void
+ OnColourClicked(BaseEventData EventData) : void
BrushTip
- paintableCanvas : PaintableCanvas
- activeBrush : BaseBrush
- canvasLayer : LayerMask
- Update()
- OnDrawGizmos()
0..1
1
updates direction
1
1
PaintableCanvas
- RaycastDistance : float
- PaintableCanvasLayerMask : LayerMask
- CanvasMeshFilter : MeshFilter
- CanvasMeshRenderer : MeshRenderer
- PixelsPerMetre : int
- CanvasDefaultColour : Color
- BrushScale : float
- BrushWeight : float
- CanvasWidthInPixels : int
- CanvasHeightInPixels : int
- PaintableTexture : Texture2D
- ActiveBrush : BaseBrush
- ActiveColour : Color
- Start()
+ TryGetUV(Vector3 worldPoint, out Vector2 uv) : bool
+ PerformDrawingWith(BaseBrush brush, Vector2 locationUV) : void
+ SelectBrush(BaseBrush InBrush) : void
+ SetColour(Color InColour) : void
+ ClearCanvas() : void
DirectionArrow3D
+ targetObject : Transform
+ arrowBody : GameObject
- arrowHead : GameObject
- arrowLength : float
- arrowHeadSize : float
- directionVector : Vector3
- Start()
- Update()
+ UpdateDirection(Vector3 newDirection) : void
- CreateArrow() : void
- UpdateArrow() : void
<<enumeration>>
EUIColourComponent
Hue
Saturation
Value
<<enumeration>>
RehabState
UKNOWN
ACTIVE_PAUSE
READY_TO_START
ACTIVE_PHASE
RELAXATION_PHASE
<<enumeration>>
ProgramState
play
done
pause
undefined
<<enumeration>>
MovementState
moveToStart
moveToHome
undefined
0..1
0..1
BaseBrush
+ DisplayName : string
+ BrushTexture : Texture2D
+ bIsTintable : bool
+ Apply(Color InCurrentColour, Color InBrushColour, Color InTintColour, float InWeight) : Color

Figure 7.5: UML diagram of **Unity** front-end

7.3 VR Front-end

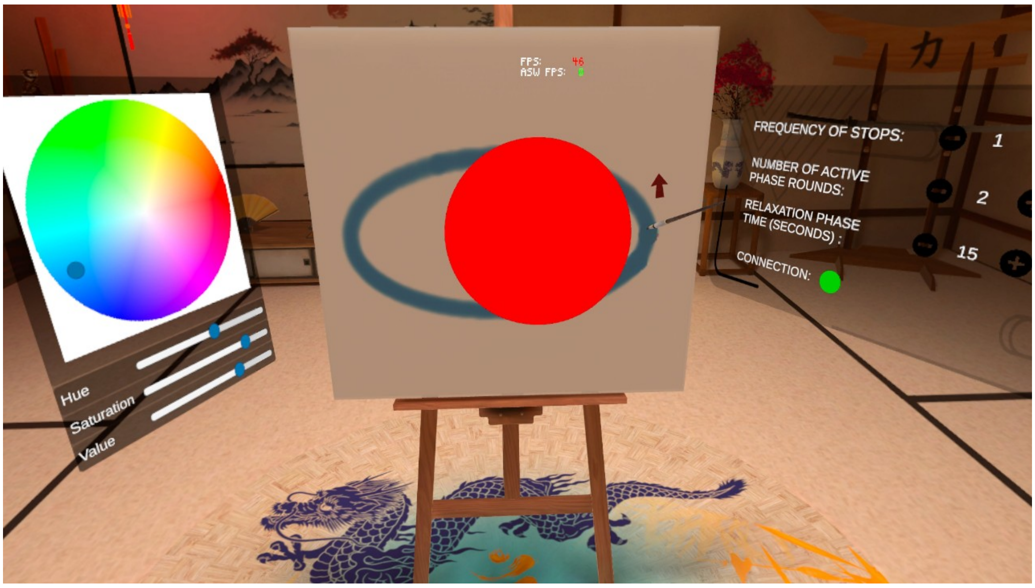
Since the VR version of the **Unity** front-end is a parallel representation of the same application, the logic and implementation are almost identical to the one that was described in great detail in Section 7.2. Therefore, this section will serve as a complement to that section and will only elaborate on features that were either added or modified for a working immersive VR experience. It was designed for *Oculus Quest* headsets using Unity's `meta` packages. The main difference stems from a different input system. The whole UI and its interaction system needed to be remade and adjusted since the interaction now became diegetic [6]. The patient interacts with the diegetic interfaces through his/her index finger. This interaction is made possible through the kinematic `RigidBody` and `Sphere collider` component on the tip of the fingers. Whenever they collide with another object that has `Collider` component set as a trigger, they will either fire an event `OnTrigger` or more complex behaviour through the `TriggerAction` script. This way, the same functionality, such as changing the slider or clicking the button, is preserved while the input trigger action changes.

The essential part of the rehabilitation scenario mentioned in Section 7.4 is movement signalisation. The patient needs to be aware of these visual cues and act accordingly. This is not a problem with a non-diegetic 2D UI similar to the one in Section 7.2. In VR, the patient has the ability to move and look around with nothing constraining his/her movement. As a consequence of that, the `VisionFollower` script resides on these important objects and ensures that they stay in the Field Of View (FOV) of the patient. An example is showcased in Figure 7.6.

Virtual Reality is a fully immersive experience, and because of that, the patient is not thoroughly aware of reality. The VR Head-Mounted Display (HMD) allows for switching between the Virtual Reality (VR) and Augmented Reality (AR) based on user preference, or even creates a blend of the two mentioned spaces. The HMD is equipped with cameras that track both hand movements and the surrounding environment in real space, allowing the captured view to be projected onto the display. This transition (passthrough) was implemented for comfortable and secure access. It works by enabling a special component called `OVRPassthroughLayer`, which renders the camera feed either behind all virtual content (underlay mode) or within a specific 3D shape like a sphere or box (selective overlay mode[7]). The feed itself is grayscale and 2D (no depth), but a custom shader was made to control how and where it's displayed - this creates an animated fade effect transition between VR and passthrough. The VR complementary code workflow is presented in Figure 7.7.

[6] A diegetic interface is when a game's interface elements exist In-Universe; the Player Character sees them, rather than just the player.

[7] Selective overlay is used in this thesis.



FREQUENCY OF STOPS: 1

NUMBER OF ACTIVE PHASE ROUNDS: 2

RELAXATION PHASE TIME (SECONDS) : 15

CONNECTION: ●

Figure 7.6: A screenshot of the VR scenario trial

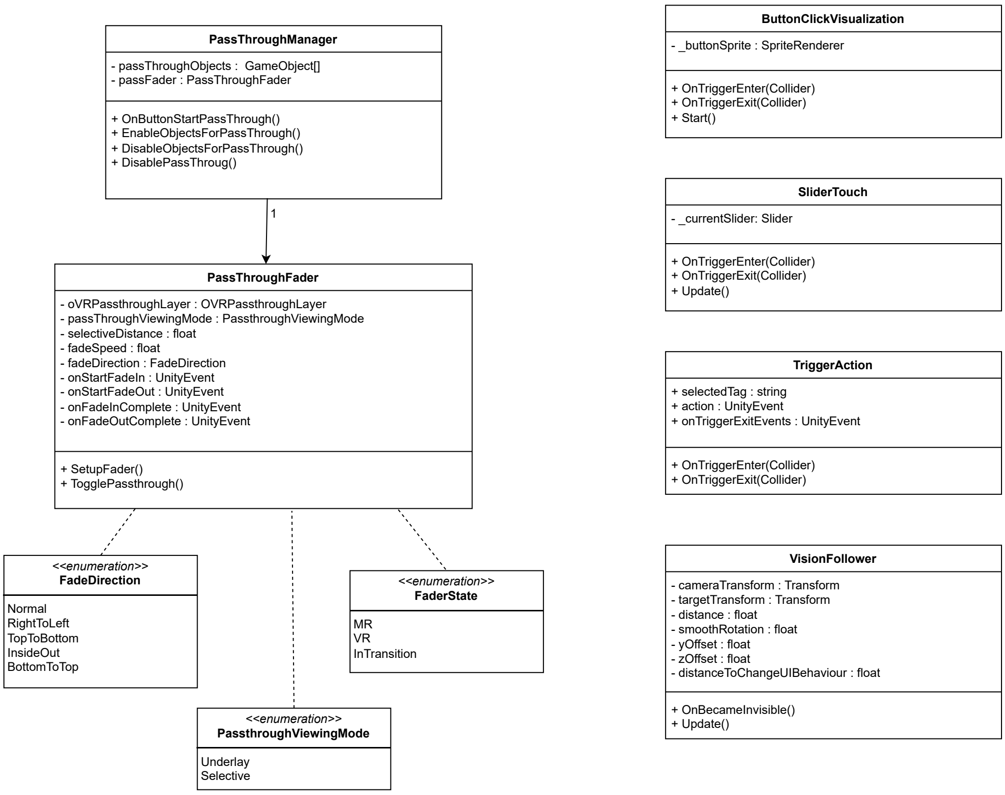

PassThroughManager
- passThroughObjects : GameObject[]
- passFader : PassThroughFader
+ OnButtonStartPassThrough()
+ EnableObjectsForPassThrough()
+ DisableObjectsForPassThrough()
+ DisablePassThroug()
1
PassThroughFader
- oVRPassthroughLayer : OVRPassthroughLayer
- passThroughViewingMode : PassthroughViewingMode
- selectiveDistance : float
- fadeSpeed : float
- fadeDirection : FadeDirection
- onStartFadeIn : UnityEvent
- onStartFadeOut : UnityEvent
- onFadeInComplete : UnityEvent
- onFadeOutComplete : UnityEvent
+ SetupFader()
+ TogglePassthrough()
<<enumeration>>
FadeDirection
Normal
RightToLeft
TopToBottom
InsideOut
BottomToTop
<<enumeration>>
FaderState
MR
VR
InTransition
<<enumeration>>
PassthroughViewingMode
Underlay
Selective
ButtonClickVisualization
- _buttonSprite : SpriteRenderer
+ OnTriggerEnter(Collider)
+ OnTriggerExit(Collider)
+ Start()
SliderTouch
- _currentSlider: Slider
+ OnTriggerEnter(Collider)
+ OnTriggerExit(Collider)
+ Update()
TriggerAction
+ selectedTag : string
+ action : UnityEvent
+ onTriggerExitEvents : UnityEvent
+ OnTriggerEnter(Collider)
+ OnTriggerExit(Collider)
VisionFollower
- cameraTransform : Transform
- targetTransform : Transform
- distance : float
- smoothRotation : float
- yOffset : float
- zOffset : float
- distanceToChangeUIBehaviour : float
+ OnBecameInvisible()
+ Update()

Figure 7.7: UML diagram of the VR complement

7.4 Rehabilitation Scenario and Dataset Creation

It must be noted at this point that the expectations regarding the classifier explained in Section 6.1 were not met. Even though both Kodera's and Pham's implementations included a scaler, the **OpenBCI** application's data differed significantly from the MI dataset used in both researchers' works. During real-time prediction, both models consistently classified all inputs as the movement class with near-certain probabilities (e.g., 0.98 for movement, 0.02 for rest), irrespective of the actual signal. Moreover, the retraining of the models on this thesis dataset yielded only around 40% accuracy. Due to the unsatisfactory performance, it was decided to construct a brand new dataset, analyse its EEG properties, and develop a classification model specifically tailored to the setup's characteristics of this thesis, mentioned in Section 7.1. This section will cover the rehabilitation scenario used in the dataset creation and real-time classification testing. Moreover, the data acquisition and dataset creation will be presented since they play an important role in the following sections.

7.4.1 FourMotors Scenario

The **FourMotors** software provides the following operational modes:

1. Normal mode without stopping
2. Stoppage for EEG learning
3. Stoppage for EEG measurements
4. Stoppage and relaxation for EEG learning
5. Stoppage and relaxation for EEG measurements

The fifth mode was chosen for dataset creation and will now be described in more detail. At any given time, the scenario is set to one of the states displayed in Table 7.2 [8].

The flow of the scenario is represented by the states, which are described in more detail below. and the configurable parameters that are associated with the scenario:

- **The number of rounds of the active phase** - An Integer, determines how many curve rounds are performed before the *Relaxation phase*.

[8] S1 signals only the state of the scenario and does not represent a marker value like the other signals.

Table 7.2: Rehabilitation States, Identifiers, and Durations

Rehab State	Identifier	Duration (sec)
Exercise running	S1	-
Active pause	S3	5
Ready to start	S4	3
Active phase	S5	-
Relaxation phase	S6	param

- **The frequency of stops** - An Integer, describes how many *Active pauses* are in one round (cycle) of the curve.

- **Relaxation phase time** - An Integer, specifies the duration of the *Relaxation phase* in seconds.

7.4.1.1 Active Phase

The patient always starts in *Active phase*. In this phase, the user tracks the curve and performs the motion. The rehabilitation arm is unlocked and free to move. The subject should concentrate on the movement in this phase. The two possible following phases are either *Active pause* or *Relaxation phase* based on the two mentioned parameters.

7.4.1.2 Active Pause

This phase presents a 5-second window, where the patient should relax and rest. The rehabilitation arm is locked, and a huge red diode is shown to a subject, representing the non-movement state. In this phase, the user should be actively relaxing, waiting for the next signalisation of movement. At the end of the duration, the diode begins to blink, signalling the end of the phase to the user. The *Ready to start* phase always proceeds after this phase.

7.4.1.3 Ready To Start

This phase should be the most prominent one regarding the ERD/ERS of the EEG signal from the motor-related cortical areas. A green blinking diode is shown to the patient, signaling the concentration on the movement. The subject wants to move the robotic arm, but the servomotors are locked. For the whole duration, the user should either provide a physical pressure on the arm or use their MI for movement concentration. The *Active phase* always follows after this phase.

7.4.1.4 Relaxation Phase

After the given number of rounds of the active phase set by the parameter, the *Relaxation phase* begins. For the selected number of seconds, a soothing video of a river stream in the woods with audio feedback overlays the rehabilitation application. The arm movement is disabled, and the patient should enter a relaxation state. The *Active pause* always proceeds after this phase.

7.4.2 Data Acquisition

The experiments were carried out in the neurolab at the Faculty of Applied Sciences at UWB over the time span of several days. In total, 8 healthy subjects volunteered, consisting of 2 women and 6 men aged 21-28 (with a mean of 24). The hardware and software setup was executed the same as it is described in Section 7.1. Each subject has been familiarised with the rehabilitation scenario detailed in Subsection 7.4.1. The **OpenBCI** application and **FourMotors** software ran in parallel - **OpenBCI** ran in the background, gathering the EEG data while the rehabilitation application conducted rehabilitation, timestamping the current state of the scenario.

The EEG recordings were saved locally and automatically after each rehabilitation trial in a CSV and plain text format. Each recording follows the standard **OpenBCI** raw EXG data format, with a sampling rate of **1000 Hz** and includes **8 EEG channels** (EXG Channel 0–7). Each row corresponds to a single sample and contains the following columns:

- **Sample Index**: A repeating integer value (0-256)
- **EXG Channels 0–7**: Raw EEG signals from the 8 electrodes
- **Accel Channels 0–2**: 3-axis accelerometer data
- **Digital Channels (D11–D18)**: States of GPIO digital input pins
- **Analog Channels 0–2**: Optional analog input readings
- **Timestamps**: Both raw and formatted time indicators
- **Marker Channel**: Custom markers used to label task events or user actions

> The EXG Channel numbering is important because it directly reflects the mapping of the electrodes in Table 7.1. The mapping in the recording is shifted, starting from zero.

The markers signalling the different states of the rehabilitation were also automatically saved by the **FourMotors** application at the end of each trial. It contains a **timestamped sequence of event markers**, each indicating the start (Sx) or end

(Rx) of a specific experimental phase. The format is a plain text log, where each line consists of:

- **Timestamp** in the format `DD.MM.YY-HH:MM:SS.sss`
- **Event label**, such as:
  - `Start`: Marks the beginning of the recording session
  - `S1–S6`: Start of different experimental phases (e.g., baseline, motor imagery, rest)
  - `R3–R6`: Corresponding end of the phases

7.4.3 Final Dataset

The final dataset consists of 24 trials and is published on *Zenodo*. All trials were performed by the right hand and lasted approximately six minutes with the following settings:

- Number of curve rounds - 10
- The frequency of stops - 1
- The number of rounds of the active phase - 2
- Relaxation phase time - 15s
- Curve type - **e**llipse, **l**emniscate, **r**ectangle, **n**ephroid [9]

The dataset is divided into subfolders, each subfolder representing a trial. The naming format of the folders adheres specific structure:

01m09042025e - <ID><Sex><Date><CurveType>

The folder contains two text files. The file `data` represents the **OpenBCI** recording, while the file `labels` presents the marker file from **FourMotors** software. The association of the data and labels was done manually by inspecting the beginning and ending timestamps.

[9] The rectangle and nephroid appear only once in the dataset because their realisation was suboptimal for the scenario purposes in comparison to the ellipse and lemniscate.

7.5 Data Analysis and Quality Assessment

In order to get a general idea of the data properties (both qualitative and quantitative) before the preprocessing stage, the data analysis was made through the `debug_data` script. The quality of each trial was assessed by seven metrics - channel correlation and amplitude range [Big+15; Ric+20], kurtosis and skewness [Xia+20], Power Spectral Density (PSD) harmonics [Red+24], band RMS ratio, and flatline detection. The analysed channels were **Cz**, **C3**, and **C4** since they are responsible for the signal from motor-related areas.

7.5.1 Channel Correlation

The channel correlation evaluates the similarity between signals recorded from the three mentioned channels. It uses the Pearson correlation coefficient. The implementation (Code 7.1) is done through the *Numpy* library.

Source code 7.1: Channel correlation analysis implementation

```
    corr = np.corrcoef(eeg.T)
    max_corr = np.max([corr[0, 1], corr[0, 2], corr[1, 2]])
    if max_corr > CORRELATION_THRESHOLD:
        issues.append(....)
```

If the maximum correlation exceeds a threshold of **0.9**, it suggests that the signals may be too similar, indicating redundancy or volume conduction effects. This could mean that the electrodes are not capturing independent brain activity, possibly due to poor placement or contact, and therefore, the signal may not be reliable.

7.5.2 Amplitude Range

The amplitude range refers to the peak-to-peak voltage observed in each EEG channel. A healthy EEG signal typically does not exceed **100 microvolts** (this is referred to as `AMPLITUDE_THRESHOLD` in Code 7.2). If any channel has a signal amplitude beyond this threshold, it likely contains motion artefacts, eye blinks, or poor electrode connections.

Source code 7.2: Amplitude range analysis implementation

```
ptp = np.ptp(signal)  # peak-to-peak range
if ptp > AMPLITUDE_THRESHOLD:
    issues.append(...)
```

7.5.3 Kurtosis

To detect statistical abnormalities in the shape of the signal distribution, the script calculates kurtosis, which measures the "tailedness" or extremity of a signal's fluctuations. High kurtosis values (**above 6** in Code 7.3) indicate the presence of sharp spikes or outliers, often caused by electrical interference or transient artefacts like electrode pops.

Source code 7.3: Kurtosis analysis implementation

```
k = stats.kurtosis(signal)
if k > KURTOSIS_THRESHOLD:
    issues.append(...)
```

A normal EEG should have a more Gaussian (bell-shaped) distribution, and excessive kurtosis undermines the signal's quality.

7.5.4 Skewness

In parallel with kurtosis, the script checks for skewness, a measure of the asymmetry of the EEG signal distribution. A symmetric EEG signal will have a skewness close to zero, indicating balanced fluctuations above and below the baseline. If skewness **exceeds** ± **1.5** (see Code 7.4), the signal is considered distorted, which can arise from long-lasting artefacts or baseline drift. Skewed distributions suggest that the data is biased and may not accurately represent ongoing brain activity.

Source code 7.4: Skewness analysis implementation

```
s = stats.skew(signal)
if abs(s) > SKEW_THRESHOLD:
    issues.append(...)
```

Both functions for kurtosis and skewness were implemented through `scipy.stats` library.

7.5.5 Power Spectral Density Harmonics

Another important metric is the detection of harmonics in the Power Spectral Density (PSD). The script uses Welch's method to compute the PSD of each channel and identifies the frequency at which each channel has the highest power. If the differences between these peak frequencies are consistently spaced (e.g., **around 10 Hz apart**), the Code 7.5 flags this as periodic noise. Such harmonics often arise from electrical interference, like 50 or 60 Hz power lines, or from repetitive, non-neural artefacts such as muscle tremors or mechanical vibrations. The analysis uses the MNE library function [10].

Source code 7.5: PSD analysis implementation

```
psd, f = psd_array_welch(eeg.T, sfreq=sfreq, fmin=1, fmax=60,
    n_fft=2048)
peak_freqs = f[np.argmax(psd, axis=1)]
harmonics = np.diff(peak_freqs)
```

7.5.6 Band RMS ratio

Another spectral quality check compares high-frequency power (30–50 Hz) with alpha band power (8–12 Hz) using the Root Mean Square (RMS) of the PSD in each band. Normally, the alpha band dominates resting-state EEG, especially in relaxed conditions. When high-frequency RMS exceeds twice the alpha RMS, the script (Code 7.6) flags this as suspicious. This typically indicates muscle artefacts (e.g., jaw clenching, neck tension) or environmental electrical noise, which disproportionately affect higher frequencies and corrupt the interpretability of the signal.

Source code 7.6: RMS ratio analysis implementation

```
rms_alpha = band_rms(signal, 8, 12)
rms_high = band_rms(signal, 30, 50)
if rms_high > 2 * rms_alpha:
    issues.append(...)
```

7.5.7 Flatline Detection

Additionally, the script includes a basic flatline or dead electrode check by computing the standard deviation of each EEG channel. If the variance is extremely low (below a set threshold - check Code 7.7), the script warns that the signal may be "flat," meaning the electrode is likely not recording any meaningful data. This could be due to physical disconnection, dried electrode gel, or hardware failure.

Source code 7.7: Flatline analysis implementation

```
std = np.std(eeg[:, i])
if std < 1e-3:
    print("Dead␣or␣flat␣signal")
```

[10] MNE is an open-source package for exploring, visualizing, and analyzing human neurophysiological data

7.5.8 Dataset Quality

All the mentioned metrics were applied to the entire dataset. Each trial was given an artefact score, based on the number of metrics it failed. Trial was marked as *Good* if there was no issue, *Suspicious* if the data had less than 4 issues, and *Bad* if more than 4 issues were found. With these settings, 7 trials were marked as *Bad*, **16** as *Suspicious* and only **1** as *Good*. The `debug_data` script logs each issue to a terminal and a file for further analysis. It also saves plots of amplitude histograms, PSDs, and time domain 2-second windows of random data samples with filter modifications to provide a better understanding of the character of the data. For illustration, the data from the only trial marked as *Good* were compared with data from a trial that was marked as *Bad* (the worst - 10 issues). For starters, the amplitude distribution comparison is displayed in Figure 7.8.

The first histogram (*Good*) displays a natural, bell-shaped amplitude spread for all three EEG channels, with most values concentrated around $0\,\mu$V and a moderate dispersion between approximately $-40\,\mu$V and $+40\,\mu$V. In contrast, the second histogram (*Bad*) reveals a severe quality issue. Although the amplitude axis stretches dramatically from $-5000\,\mu$V to $+5000\,\mu$V, nearly all data points fall within an unnaturally narrow band centered at $0\,\mu$V. The towering count values exceeding 250,000 per bin strongly indicate that the signals are nearly flat, lacking dynamic variation. Channel-wise amplitude distribution was also plotted with three different filter settings:

1. Raw data

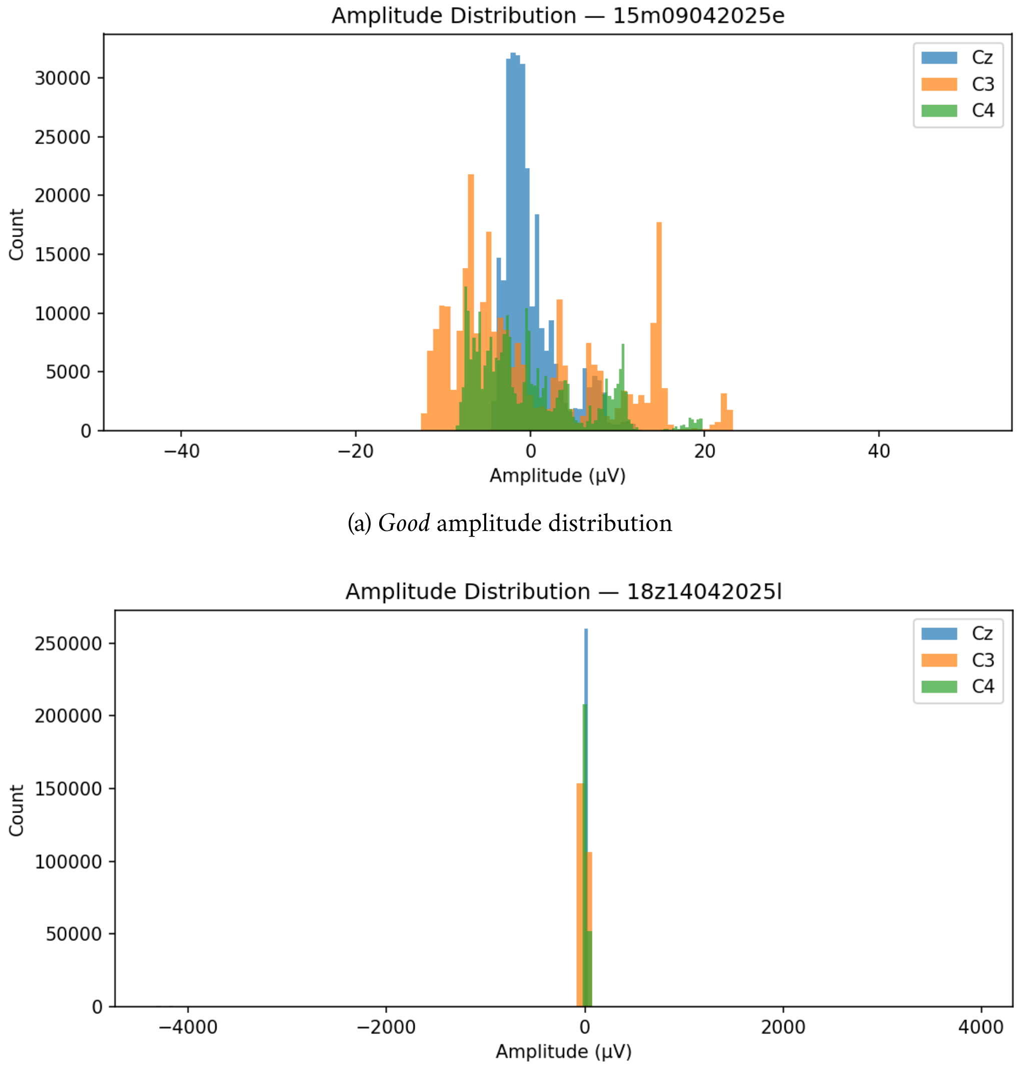
2. Filtering with Notch filter (50Hz) and High-pass Filter (with cutoff 8Hz)

3. Notch with HPF and Band-pass Filter (8- 22Hz)

Since all the trials were performed, as mentioned in Section 7.4.3, using the right hand, the main area of interest lies in C3 electrode comparison, shown in Figure 7.9.

The raw signal in the *Good* trial exhibits a broad amplitude spread, primarily ranging from approximately $-15\,\mu$V to $+20\,\mu$V, which is consistent with physiologically active EEG data. After applying the filtering, the distribution becomes narrower but retains a meaningful shape. On the other hand, all versions of the *Bad* C3 signal are narrowly concentrated around $0\,\mu$V, with an extreme spike in count at that central value. This further proves that the signal is nearly flat. This statement is backed up by the PSD comparison of the C3 channel in Figure 7.10.

In the *Good* trial, the PSD curve shows a typical spectral profile with a clear decrease in power as frequency increases, and a visible peak around 10Hz, which likely corresponds to neural activity in the alpha band, while the *Bad* trial shows an



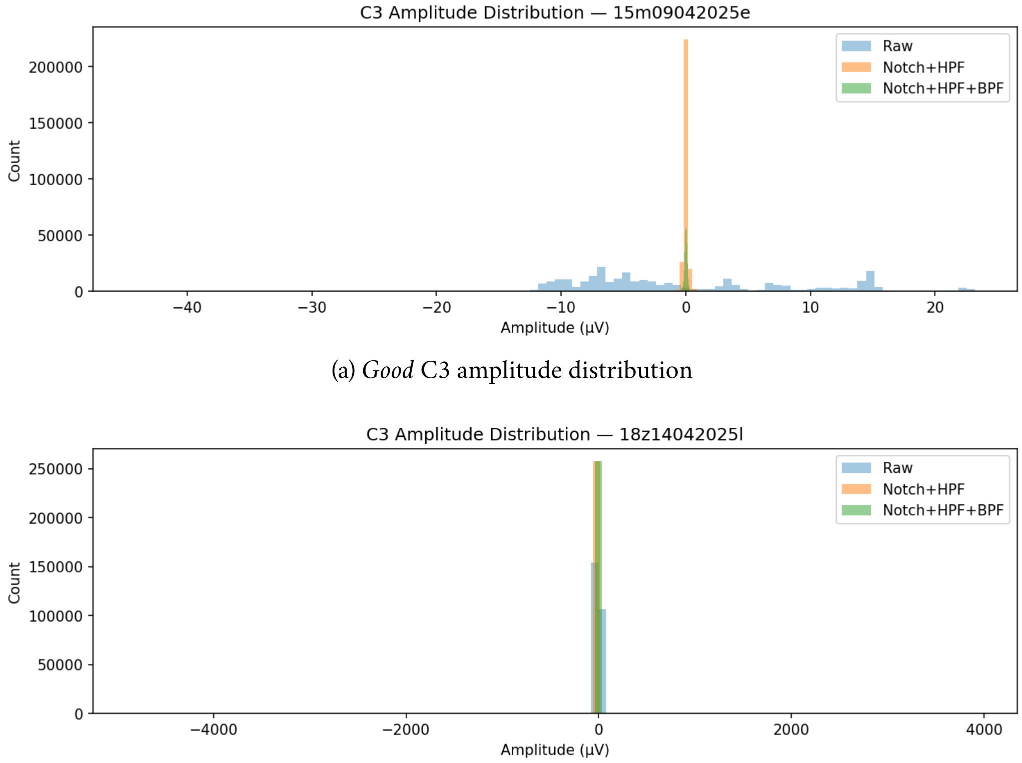
(a) *Good* amplitude distribution

(b) *Bad* amplitude distribution

Figure 7.8: Amplitude distribution comparison

abnormally high power level across the spectrum, with no discernible physiological peaks.

The time-domain comparison plots in Figure 7.11 highlight a distinct difference in signal characteristics. In the *Good* trial, the application of notch and high-pass filters successfully suppresses low-frequency drift and power line interference. In the *Bad* trial, there is an absence of any discernible neural activity or noise structure, and filtering has minimal impact on the waveform, showing low-variance signals across all three channels in both raw and filtered traces. The repeating signal drops suggest signal corruption or hardware malfunction.



(a) *Good* C3 amplitude distribution

(b) *Bad* C3 amplitude distribution

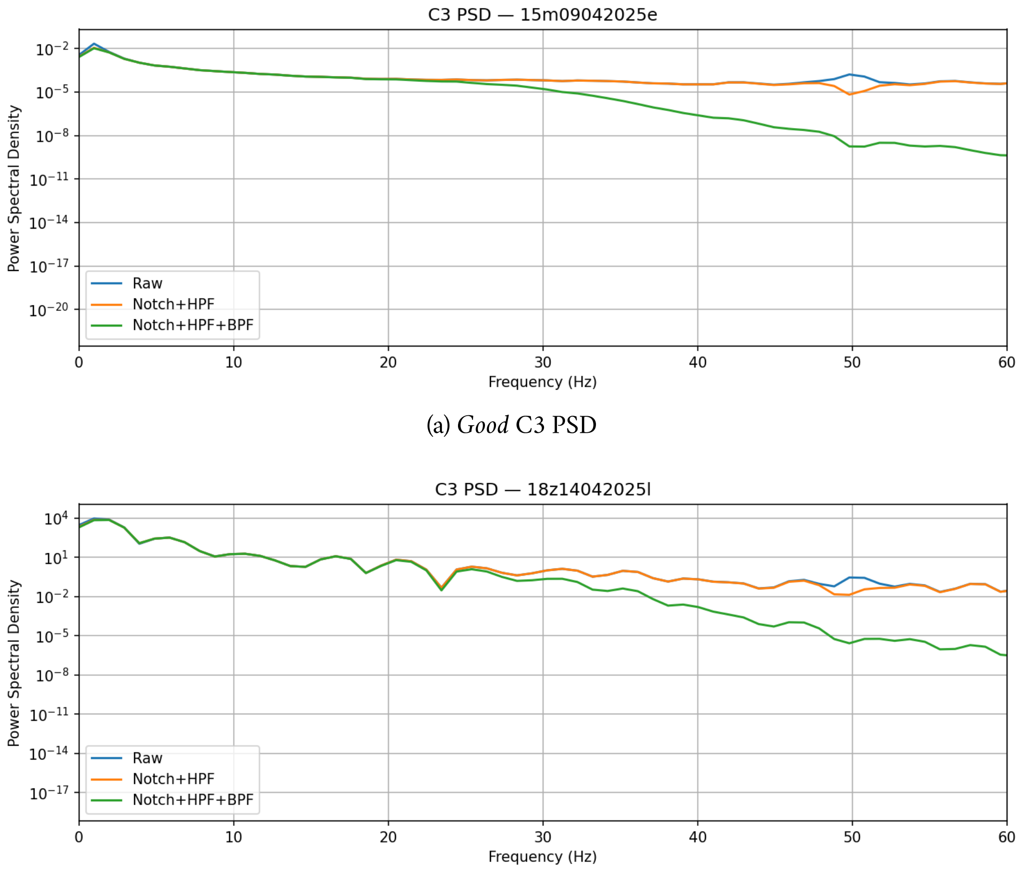
Figure 7.9: C3 filtered amplitude distribution comparison

7.6 Classifier Design and EEG Preprocessing

This section outlines the structure, preprocessing, and implementation of the classification project that was implemented in Python. The project uses the following libraries:

- `NumPy` (Numerical Python) provides efficient tools for working with multi-dimensional arrays and performing scientific computations. It is one of the most essential and widely used libraries in the Python ecosystem for scientific computing. Many other libraries rely on NumPy arrays as their core data structure. In the context of this project, NumPy is primarily used for storing and manipulating preprocessed data.

- `MNE` - library specialized in the analysis of neurophysiological data. It offers a wide range of tools and objects for loading, processing, visualizing, and analyzing such data, including EEG signals. In this project, it is used primarily in one of the implementations for loading and preprocessing the recorded EEG data.



(a) *Good* C3 PSD

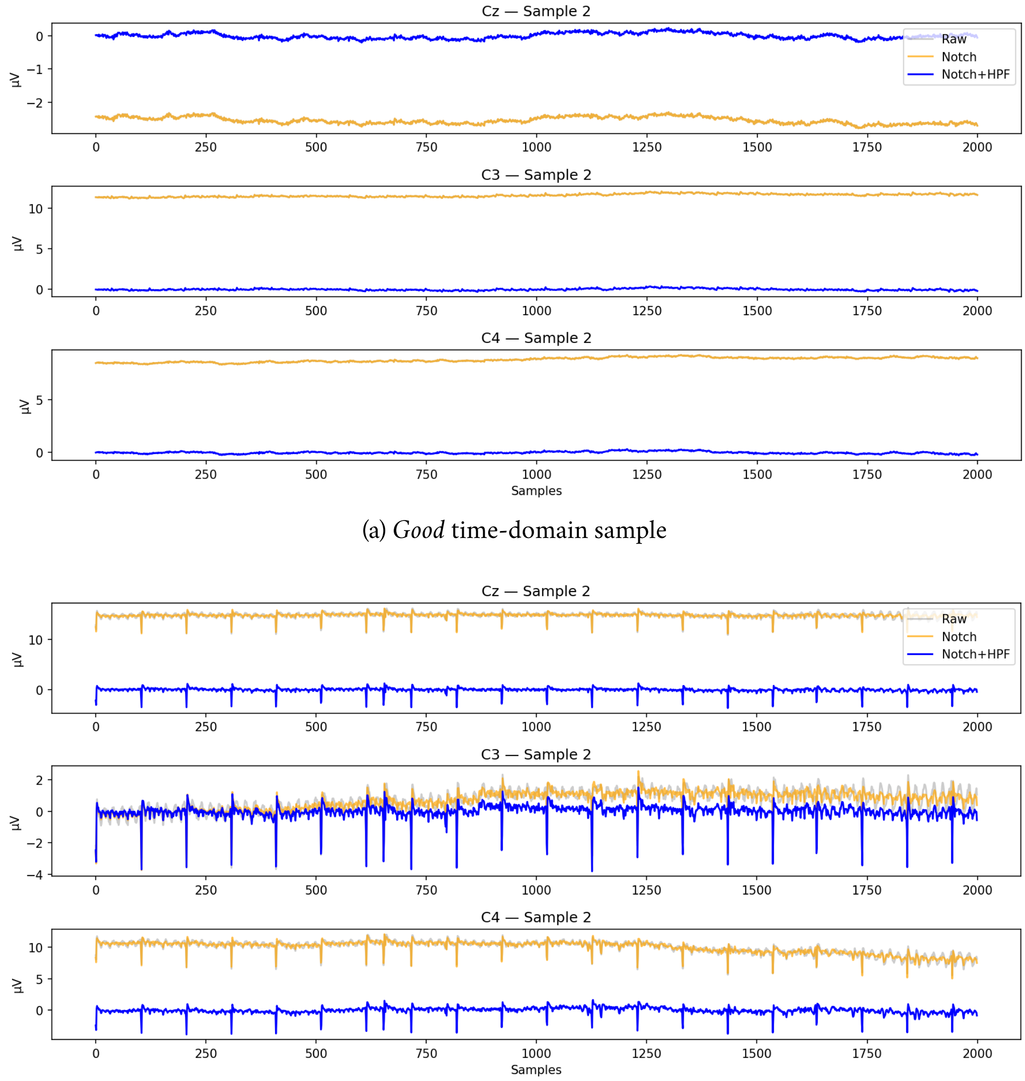
(b) *Bad* C3 PSD

Figure 7.10: C3 filtered PSD comparison

- SciPy - a scientific computing library that builds on top of NumPy and provides additional functionality for advanced mathematical operations. It includes modules for signal processing, optimization, integration, statistics, and more. In this project, SciPy is used mainly for filtering EEG signals and performing statistical analyses during preprocessing.

- scikit-learn - a machine learning library in Python that offers a wide variety of tools for data modeling, classification, regression, clustering, and model evaluation. In this project, scikit-learn is used for training and evaluating machine learning models on extracted EEG features.

- Matplotlib is a plotting library that provides functionality for creating visualizations and rendering various types of data plots. In this project, Matplotlib is used to visualize EEG signals, power spectral densities, classification results, and data distributions.



(a) *Good* time-domain sample

(b) *Bad* time-domain sample

Figure 7.11: Filtered time-domain comparison

- `Keras/TensorFlow` – Keras is a high-level library for deep learning and neural network development. It offers an intuitive API that makes it easy to build and train various neural network architectures. TensorFlow, on the other hand, is a lower-level library that provides detailed control over model operations, along with support for hardware acceleration (e.g., GPU) to optimize computational performance. In practice, Keras uses TensorFlow as its backend, meaning that Keras internally relies on TensorFlow's capabilities. The combination of these two libraries enables rapid prototyping through Keras's user-friendly interface, while still allowing access to TensorFlow's advanced features for customization and performance optimization.

7.6.1 Architecture

The project implementation is divided into 5 modules/packages - `Classification`, `Preprocessing`, `Utils`, `Testing`, `Main`. The communication and the structure of the modules are visualised in Figure 7.12. Each module is presented in a dedicated subsection, with the preprocessing and classification parts further elaborated in the subsequent sections.

7.6.1.1 Clasification

The classification module consists of three classes:

- `CNN` - The class represents a convolutional neural network implemented through `TenserFlow/Keras`. A detailed description can be found in Section 7.6.3.
- `MLP` - This class represents a Multilayer perceptron, which works especially well with extracted features from the data. It is discussed in greater detail in Section 7.6.4.
- `CNN_Features` - The initial script, which was later replaced by the previously mentioned. The main idea was to combine the power of CNN for extracted features.

7.6.1.2 Preprocessing

The preprocessing was divided by the classification method it is used by. Since the development was sequential, and new methods of preprocessing were discovered at different times, the package has the following structure:

- `debug_data` - This script was already mentioned in the Section 7.5. It consists of functions that provide an analytical view of the dataset.
- `prep_core` - It embodies the core preprocessing functionality used by both following scripts.
- `prep_mne` - An extension of `prep_core` with added functionality needed for correct MNE feature extraction.
- `prep_raw` - Also and extension of `prep_core`. This time is used for raw time series data format with minimal preprocessing.
- `prep_old` - An old initial script, no longer used but kept for documentation purposes.

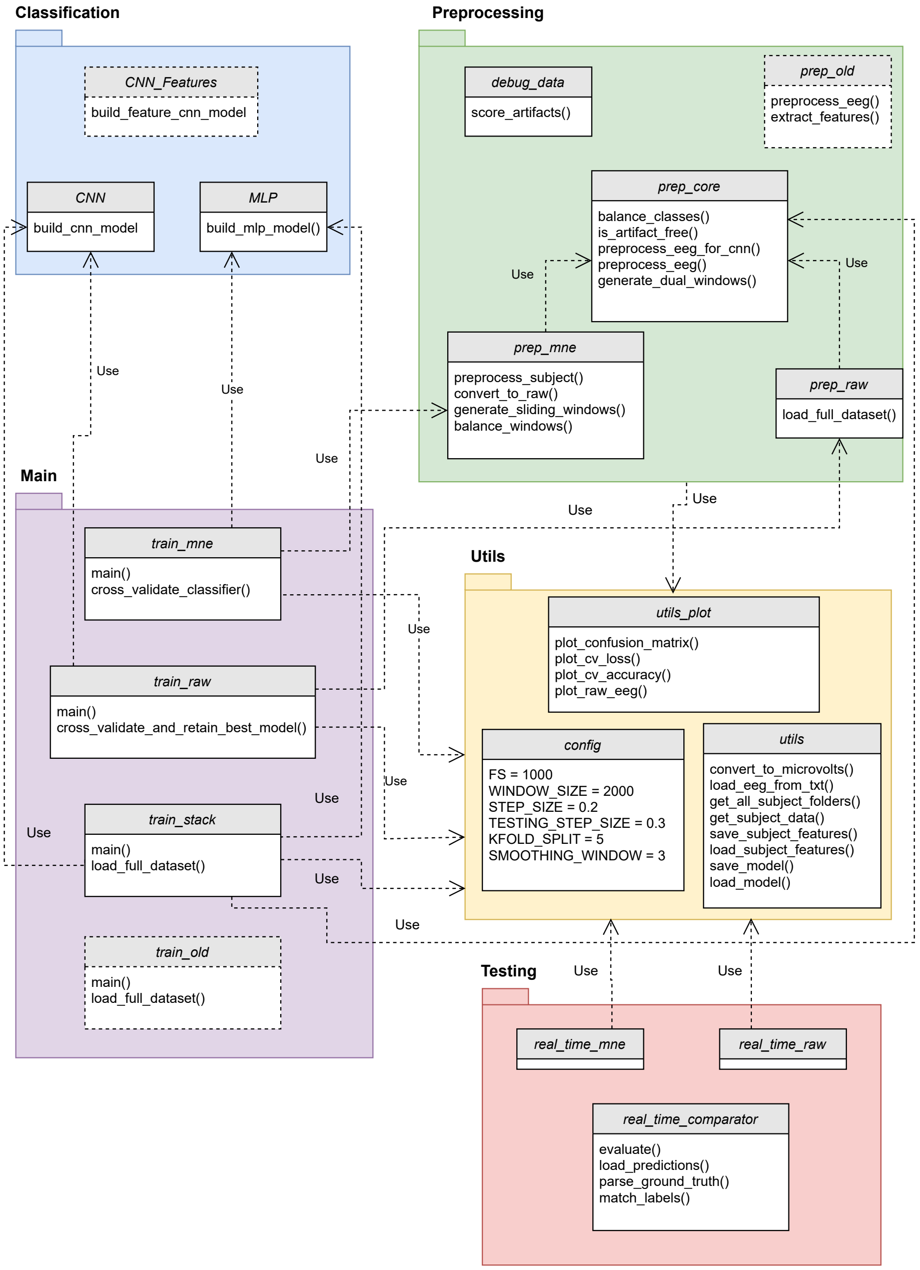


Figure 7.12: UML diagram of classifier architecture. It should be noted that, for clarity, only the most essential methods, variables, and relationships are shown.

7.6.1.3 Utils

The Utils module is constructed from scripts that provide general utility or variables, which are used across the whole project. It consists of the following scripts:

- config - The configuration script contains all the static configurable variables used in preprocessing, classification, file management, and mapping.
- utils - All logic, including file management, import/export of input data, models, scalers, plots, classification reports, preprocessed saved data, or other utility functions, resides in this script.
- utils_plot - Provides functions for plotting confusion matrices and classification trends, input data, and channel-wise comparisons.

7.6.1.4 Testing

This package was created for real-time classification predictions of the two implemented approaches. It is made of three scripts:

- real_time_mne - The script loads the created MLP model with scaler and CSP filters (more information in Section 7.6.2). After that, it predicts incoming EEG data through the LSL protocol and progressively saves the predictions for further offline analysis (properly described in Section 8.2).
- real_time_raw - Loads the CNN model a predicts movement through the LSL protocol, similarly to the preceding script (also described in greater detail in Section 8.2).
- real_time_comparator - Synchronizes the predictions from real_time_mne or real_time_raw with the ground truth signaling from rehabilitation software, providing analysis of the real-time trial predictions.

7.6.1.5 Main

In the implemented system, several experimental training pipelines were explored and iteratively refined to achieve robust classification of EEG signals related to motor intention. The primary distinction between these pipelines lies in the type of input used (raw signals vs. extracted features) and the classification strategy (single model vs. stacked ensemble). The outlined key architecture variants developed throughout the project are as follows:

- train_mne - The script focuses exclusively on high-level features extracted using the MNE library. This variant trains a standalone MLP classifier with early stopping on standardized CSP+feature vectors. It supports both **group-aware** and **stratified cross-validation**, providing plots of accuracy and loss trends across folds. This version benefits most from the heavy preprocessing featured in Section 7.6.2.

- `train_raw` - Represents a baseline approach, where a standalone Convolutional Neural Network (CNN) is trained directly on raw EEG windows. The script loads the complete dataset, applies cross-validation using either **GroupKFold** or **StratifiedKFold**, and evaluates the model performance on held-out folds (closely inspected in Section 8.1). The best-performing model is then retrained on the full dataset to maximize generalization and stored for later use.

- `train_stack` - Introduces a hybrid ensemble training architecture. This pipeline processes EEG data in parallel along two paths: (1) a raw data branch for MLP processing, and (2) a feature-based branch for MLP processing. Raw windows are minimally preprocessed and fed to a MLP, while a separate feature extraction procedure (described in Section 7.6.2) transforms the same time segments into feature vectors used by a Multilayer Perceptron (MLP). The prediction probabilities from both models are then used as inputs for a final meta-classifier, which is a Random Forest (RF). This stacked ensemble approach benefits from both low-level temporal signal features and high-level engineered features. **Inner cross-validation** is used during the meta-classifier training to avoid overfitting and information leakage.

- `train_old` - An archaic version of a training architecture, consisting of heavy feature preprocessing, which was then used in the training of MLP. It served as a prototyping sandbox and verification point for later, more structured pipelines.

7.6.2 Preprocessing

The EEG data are prepared through a pipeline of filtering, normalization, artifact removal, and segmentation before being fed into machine learning models. This preprocessing ensures that both the raw data branch (for MLP input) and the feature-based branch (for MLP input) work with cleaner and more informative data. Common operations applied to the EEG include frequency filtering to isolate important bands, re-referencing and scaling to standardize the signals, and the use of event markers to label data segments for supervised learning. In particular, markers from the experiment protocol (phases S1–S6) are used to assign class labels to time intervals (See Table 7.2). Because neural state transitions (e.g. movement onset/offset) are gradual, the labeling strategy incorporates the possibility pf soft labels (decayed labels) for certain phases (notably S5 and S6) – this means that instead of a hard 0/1 class label, transitional periods are labeled with intermediate values (e.g. 0.5) to reflect the partial engagement of the target activity during those windows (this fact is further debated in Section 9.1). For increased clarity, the mapping of the labels is shown in Table 7.3.

Table 7.3: Label mapping with and without soft labels

Phase Marker	Label (Soft)	Label (Hard)
S3 (Pause)	0.0	0
S4 (Movement Attempt)	1.0	1
S5 (Active Movement)	1.0 (first 4s), then 0.5	1
S6 (Relaxation)	0.5 (first 4s), then 0.0	0

Another important information is regarding the EEG data format saved in the plain-text files of the **OpenBCI** recordings. **OpenBCI** GUI filters and streams all the channel data in microvolts, but that is not the case for the recordings:

> **OpenBCI does not store EEG data directly in microvolts**. Instead, it stores **raw integer values from the ADC** (Analog-to-Digital Converter). For the Cyton board, which is based on the ADS1299 chip, the ADC has a resolution of **24 bits**. With a Programmable Gain Amplifier (PGA) gain of ×24, the voltage resolution is approximately 0.02235 μV per bit.

This important yet hidden detail was revealed by a deep analysis of the input data through the `debug_data` script (Section 7.5). On account of this newly discovered reality, a conversion function (Code 7.8) was created to transform the input data into the correct units.

Source code 7.8: Microvolts conversion

```python
def convert_to_microvolts(eeg_raw, gain=24, adc_resolution=2**23 - 1, v_ref=4.5):
    """
    Convert raw EEG signal to microvolts based on OpenBCI parameters.
    Parameters:
    - eeg_raw: np.ndarray, raw EEG signal
    - gain: int, amplifier gain (default 24)
    - adc_resolution: int, max ADC value (default 2^23 - 1)
    - v_ref: float, reference voltage in volts (default 4.5V)
    Returns:
    - eeg in microvolts as np.ndarray
    """
    scale_uV = (v_ref / adc_resolution) * 1e6 / gain
    return eeg_raw * scale_uV
```

The following subsections detail the preprocessing workflows for the raw-signal branch and the feature-extraction branch, respectively, highlighting what was done and the motivation behind each step.

7.6.2.1 Raw Preprocessing

For the raw EEG branch, the goal was to apply minimal necessary preprocessing so that the convolutional neural network can learn directly from largely unaltered signals. The raw EEG (recorded from three channels: Cz, C3, C4) is processed with a lightweight pipeline that cleans the data while preserving its temporal structure. The steps involved in raw preprocessing are described in the subsequent paragraph.

High-Pass Filtering $\geq$ 0.5 Hz. A gentle high-pass filter with a cutoff around 0.5 Hz is applied to remove very slow drifts and DC offsets in the signal. Such drifts can be caused by perspiration, slow electrode baseline shifts, or movement of the electrode leads. Removing frequencies below 0.5 Hz stabilizes the baseline of the EEG signal, preventing large swings that could interfere with the MLP's training (since the network might otherwise learn the DC level or slow trends rather than task-related patterns).

Band-Pass Filtering (8–30 Hz). Band-Pass Filtering (8–30Hz): After high-pass filtering, a band-pass filter is applied to retain the EEG frequency components most relevant to motor activity, while attenuating irrelevant noise. The passband of 8–30Hz covers the **mu (α)** and **beta** rhythms, which are known to be associated with motor processes. By restricting the input to this range, low-frequency noise is removed (e.g. slow cortical potentials, electrode drift) and high-frequency noise (muscle activity, external electrical interference) that fall outside the motor-related band. This focuses the MLP on the key frequency content (mu/beta power changes) that differentiates movement vs. rest mental states.

Re-referencing (Common Average Reference). The filtered EEG signals are re-referenced using the Common Average Reference (CAR) technique. In CAR, the average of all electrode channels' signals is computed at each time point and subtracted from each channel. This step mitigates noise that is common across the electrodes and reduces the influence of the original reference electrode. By referencing the global average, each channel's signal is centred relative to the overall brain activity, which often improves signal quality and comparability across channels.

Artifact Rejection (Peak-to-Peak Threshold). Even after filtering, some EEG segments may contain transient artefacts (for example, due to jaw clenching or motion) that produce abnormally large amplitude fluctuations. To guard against feeding corrupted data to the MLP, the pipeline checks each sliding window of EEG for excessive amplitude. If the peak-to-peak voltage of any channel in a window exceeds a set threshold (e.g. 100μV, a value above typical EEG amplitudes for healthy adults), that window is likely contaminated and is discarded.

Segmenting into Windows. The continuous EEG stream is chopped into consecutive sliding windows to create fixed-length samples for classification. In the implementation of this thesis, a window length of 2 seconds is used with a 0.2s step (i.e. 200ms shift between windows). This yields overlapping windows that capture temporal context and increase the number of training samples. Each window inherits a label based on the task markers: windows overlapping with a movement period are labelled as class 1 (movement), and those in rest periods as class 0 (no movement).

Class Balancing. After windowing and labelling, the set of raw EEG windows is examined for class imbalance. In many sessions, there can be an unequal number of movement vs. rest windows (e.g., rest periods often last longer, yielding more windows of class 0). To prevent the MLP from biasing towards the majority class, the training set is balanced. This was done by oversampling the minority class windows (replicating or jittering them as needed) until both classes are represented roughly equally.

In summary, the raw preprocessing pipeline yields a set of 2-second EEG windows (in microvolt units) that are band-limited to 8–30 Hz, re-referenced, and free of gross artifacts, with each window assigned a possibly soft label indicating the presence or absence of movement. These minimally processed windows are then used as input to the MLP for end-to-end learning. Each preprocessed trial is saved as a `.npy` file for later usage in repeating training runs.

7.6.2.2 Feature Extraction Preprocessing

The feature-based branch of the pipeline requires a more extensive preprocessing procedure, since it relies on engineered features extracted from the EEG. In this branch, each EEG window is transformed into a vector of quantitative features that characterize the signal's content in time, frequency, and spatial domains. To ensure that these features are meaningful, the raw data undergo rigorous cleaning and standardization prior to feature computation. The preprocessing for feature extraction includes the following steps: some of them are slightly modified versions of the previously mentioned preprocessing pipeline.

Filtering and Artifact Removal. Similar to the raw pipeline, the EEG is first filtered to remove unwanted frequencies and noise. A notch filter at 50 Hz is applied to eliminate mains power interference. Then, a band-pass filter (approximately 8–22 Hz) is used to isolate the mu and low-beta frequency components of interest. The slightly narrower band (compared to the raw branch) focuses on the core motor-related frequencies. This was done through the `MNE` library functions for experiment purposes. After filtering, the EEG is re-referenced using the common average reference, just as in the raw pipeline, to improve signal clarity. An optional step of clipping extreme values is incorporated.

Sliding Window Segmentation. The cleaned continuous EEG is segmented into overlapping windows as the raw branch. Before accepting a window for feature extraction, however, a stricter artefact rejection check is performed. For each candidate window, multiple criteria are evaluated:

- **Amplitude criterion:** No channel's peak-to-peak amplitude should exceed a predefined threshold (e.g., 150 μV for feature windows). If any channel exceeds this threshold, the window is considered artifact-contaminated and discarded.
- **Flatline criterion:** The standard deviation of each channel within the window must exceed a minimum value (e.g., > 0.03 μV). A very low standard deviation indicates either a flatline (i.e., no measurable brain activity) or a disconnected or faulty electrode, and such windows are rejected.
- **Variance outlier criterion:** The variance of each channel is compared against the median variance across channels using a z-score. If the absolute z-score of any channel exceeds a set threshold (e.g., > 10), the window is considered to contain abnormal signal fluctuations or noise, and it is excluded from further processing.

Only windows that pass all these checks are retained for feature extraction.

Signal Normalization. For the purpose of stable feature computation, the EEG data is standardized. In practice, this means that for each channel, a z-score normalization is applied. Normalizing signals to have zero mean and unit variance should remove inter-subject or inter-session differences in overall EEG amplitude. This step should ensure that features like variance or power are comparable across different recordings and are not biased by one subject having generally higher amplitude signals than another.

Class Balancing. The class balancing copies the oversampling mechanism from the raw pipeline. Data augmentation was thought of, but later discarded.

Feature Computation. After the above preprocessing, each clean EEG window is transformed into a feature vector. A diverse set of features is extracted, aiming to capture information in multiple domains:

Common Spatial Patterns (CSP) CSP is a well-known technique in BCI that finds spatial filters maximizing the variance difference between two classes. CSP filters are computed using the training data's covariance matrices of the movement vs. rest epochs. Each window is projected onto a few of the most discriminative spatial filters, and the log-variance of the projected signals is taken as a feature. These CSP log-variance features reflect how the variance distribution across channels changes with the class, effectively capturing the spatial patterns of activation associated with movement.

Band Power Features The frequency-domain energy of the EEG is quantified by computing the Power Spectral Density (PSD) of each window. Band-specific power is extracted from the PSD, notably the average power in the mu band and beta band. These features (often log-transformed) indicate how much power the EEG carries in the mu and beta ranges. This is relevant because, for instance, actual or attempted movement typically causes mu power suppression and beta modulation in motor cortex signals [Ina+23].

Hjorth Parameters For each window and each channel, the three Hjorth parameters are computed – Activity, Mobility, and Complexity. **Activity** is essentially the signal variance (a measure of signal power in the time domain). **Mobility** is defined as the ratio of the standard deviation of the first derivative of the signal to the standard deviation of the signal itself (related to the dominant frequency or bandwidth of the signal). **Complexity** is the ratio of the mobility of the first derivative to the mobility of the signal (reflecting how the frequency content changes, i.e., the signal's resemblance to a pure sine wave). These parameters provide a compact description of the signal's shape and frequency characteristics. For example, a movement interval might show higher Activity (more variance) and different Complexity compared to rest.

Entropy To capture the signal's unpredictability or complexity from another perspective, the entropy of the signal amplitude distribution is calculated for each window. Shannon entropy is used on the histogram of EEG amplitudes in the window. A higher entropy indicates a more irregular or information-rich signal, whereas a lower entropy might indicate a more regular or periodic signal.

Time-Frequency (STFT) Power In addition to static band power, a Short-Time Fourier Transform (STFT) is applied to each window to examine how power in certain bands is distributed over time within the window. We divide the 2s window into shorter segments (using STFT with, e.g., 250ms segments) and calculate the average power in the mu and beta bands across time. Essentially, this yields features similar to bandpower but derived from a time-frequency representation, which can be more sensitive to transient bursts or temporal fluctuations in those bands. For instance, if a burst of beta oscillation occurs only in the second half of the window, the Welch PSD (which averages the whole window) and the STFT average might capture it differently.

Riemannian Covariance Features Finally, features based on the EEG signal's spatial covariance matrix, inspired by Riemannian geometry approaches in BCI, are incorporated. For each window, the 3×3 covariance matrix of the three EEG channels is computed. We then find its eigenvalues and take the logarithm of these eigenvalues as features. These log-eigenvalue features encapsulate how the power is distributed across the three channels in a single window (since eigenvalues of the covariance matrix correspond to the variance along principal component directions of the signals). The rationale is that different mental states may induce different spatial covariance patterns in the EEG.

Feature Standardization. Once all features are extracted for every window, the feature vectors are standardized using a `StandardScaler` so that each feature dimension has zero mean and unit variance across the training set. This standardization is applied after feature extraction, and the same scaling is used on validation/test data to avoid data leakage.

After these preprocessing steps, each EEG window is represented as a rich feature vector of 29 features, combining spatial, spectral, and statistical attributes. The thorough preprocessing and artifact rejection in the feature branch aim to ensure that the input to the MLP is as informative and noise-free as possible.

7.6.3 CNN

The convolutional neural network used in this work is designed specifically for end-to-end learning from raw EEG signal windows. It is loosely inspired by the EEGNet architecture [Law+18] but extends it with additional temporal modelling via an LSTM layer. The model accepts inputs in the shape of `(time_steps, channels)`, corresponding to sliding windows of raw EEG data from three electrodes (Cz, C3, and C4). Its architecture consists of four main stages:

1. **Temporal and Spatial Filtering:**
    - A `Conv1D` layer with 8 filters and a kernel size of 64 is used for initial temporal filtering.
    - This is followed by `DepthwiseConv1D` with a kernel size of 64 and depth multiplier of 2 to separate channel-wise (spatial) filtering.
    - Both layers are followed by batch normalization, ReLU activation, average pooling, and dropout (0.5) to reduce overfitting.
2. **Feature Extraction:**
    - A `SeparableConv1D` layer with 16 filters and a kernel size of 16 further extracts localized temporal features.
    - This is again followed by batch normalization, activation, pooling, and dropout (0.5).
3. **Temporal Modeling with LSTM:**
    - An `LSTM` layer with 64 units is added to capture sequential dependencies over time.
4. **Classification Head:**
    - A `Dense` layer with softmax activation outputs probabilities for the two target classes.
    - The model is trained using the Adam optimizer and categorical cross-entropy loss, allowing for soft label supervision.

This model is trained with early stopping and learning rate reduction on the plateau.

7.6.4 MLP

The second model is a feed-forward multilayer perceptron that operates on feature vectors extracted from preprocessed EEG data. This model does not use raw EEG but instead relies on engineered features such as CSP log-variance, bandpower, entropy, Hjorth parameters, and others (see Section 7.6.2 for details on preprocessing and feature extraction). The architecture of the MLP is as follows:

- **Input Layer:** The input vector includes all concatenated features extracted from each window of the EEG signal. These features are standardized using `StandardScaler`.

- **Hidden Layers:**
    - First hidden layer: 128 neurons with ReLU activation, batch normalization, and dropout (0.5).
    - Second hidden layer: 64 neurons with batch normalization and dropout (0.4).
- **Output Layer:** A dense layer with 2 output units and softmax activation provides class probabilities.

Like MLP, this model is trained with soft labels and uses categorical cross-entropy as the loss function. Training is conducted using the Adam optimizer, with early stopping and learning rate scheduling to improve generalization.

7.6.5 Meta-Classifier

A stacked ensemble model was implemented to combine the strengths of both the raw-signal CNN and the feature-based MLP. This approach aimed to improve overall classification performance by leveraging diverse representations of the EEG data. The ensemble follows a two-stage training process:

1. **Base Learners:**
    - The CNN and MLP models described in Sections 7.6.3 and 7.6.4 are trained independently on the same training folds during cross-validation.
    - Predictions (softmax probabilities) from both models are recorded on a validation split (meta-training set) within each fold.
2. **Meta-Classifier:**
    - The concatenated probability vectors from the CNN and MLP form the input features for a `RandomForestClassifier`.
    - This classifier is trained using only the outputs from base learners on the inner validation splits to prevent information leakage.
    - Once trained, it is used to make final predictions on the outer test fold.

Only samples with hard labels (0 or 1) are used in the final evaluation to ensure reliability. This stacked ensemble is implemented in `train_stack.py`, using `GroupKFold` for subject-wise validation and stratified inner splits for meta-training. The entire architecture of both MLP (Section 7.6.3) and MLP (Section 7.6.4) with the pipeline of the stacked ensemble model is visualized in Figure 7.13.

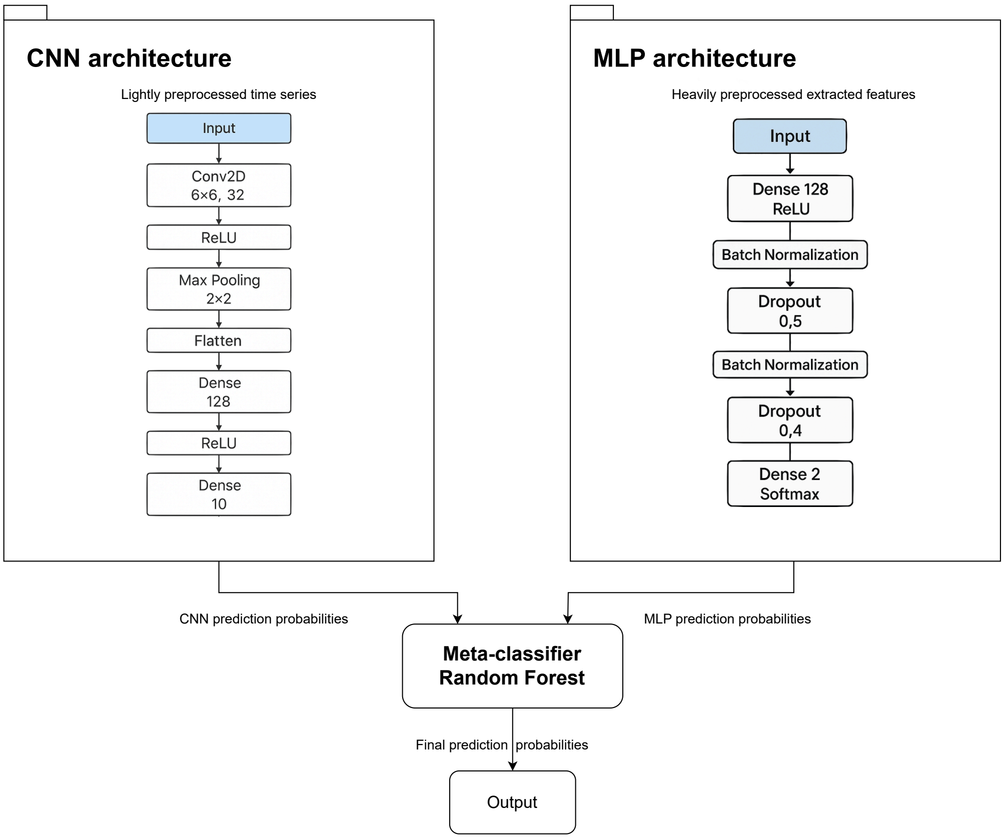


Figure 7.13: Architecture and pipeline of the stacked model

Evaluation and Results

8

The evaluation of the trained classifiers is performed by predicting the class membership of the test data. Based on the predicted and actual classes of individual feature vectors, various classification metrics are computed, primarily those derived from the confusion matrix

Accuraccy. Expresses the ratio of correctly made predictions to the total number of predictions made:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{8.1}$$

If we designate one class as positive and the other as negative, we can define the key elements of the confusion matrix as follows. True Positives (TP) represent the number of feature vectors correctly classified as positive, meaning the actual class is positive and the classifier also predicted the positive class. True Negatives (TN) are the number of feature vectors correctly classified as negative. False Positives (FP) occur when the actual class is negative, but the classifier incorrectly predicts the positive class. Conversely, False Negatives (FN) represent cases where the actual class is positive, but the classifier incorrectly predicts the negative class. These four values form the basis for computing various classification performance metrics.

Precision. The precision metric is defined as the ratio of correctly classified positive feature vectors to the total number of feature vectors predicted as positive.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8.2}$$

Precision focuses on how accurate the classifier's positive predictions are. A higher precision value indicates that most of the feature vectors predicted as positive were indeed correct, whereas a lower value suggests that many of the predicted positives were incorrect.

Recall. Recall focuses on how well the classifier is able to identify all positive instances. A higher recall value indicates that most of the actual positive feature vectors were successfully detected by the classifier, while a lower value means that many of the true positives were missed (i.e., classified as negative).

$$\text{Precision} = \frac{TP}{TP + FN} \tag{8.3}$$

F1 Score. The F1 score is defined as the harmonic mean of precision and recall. It combines both metrics into a single value to provide a balanced measure of classification performance, especially in cases where the class distribution is imbalanced.

$$\text{F1 Score} = \frac{2TP}{2TP + FP + FN} \tag{8.4}$$

The F1 score emphasizes the trade-off between precision and recall. A high F1 score indicates that the classifier achieves both high precision and high recall, meaning it correctly identifies a large portion of positive instances while keeping false positives low.

8.1 Offline Evaluation

To rigorously assess classifier performance prior to real-time deployment, a comprehensive offline evaluation using two cross-validation strategies: Stratified K-Fold (SKF) and Group K-Fold (GKF) was conducted with **5 folds**. These evaluations provide insights into both within-subject generalization (SKF) and across-subject generalization (GKF), as well as the models' tendencies toward overfitting, class imbalance, or decision boundary collapse. The three mentioned classifiers (MLP, MLP, and Meta-classifier) were evaluated under this framework. The metric overview is displayed in Figure 8.1, while each classifier's results are described in greater detail in the following subsections.

8.1.1 CNN Model

Under stratified 5-fold validation (where each fold preserved class balance but mixed data from all subjects), the MLP achieved consistently high accuracy across folds. The fold-by-fold accuracy plot shows only minor variation between folds, indicating

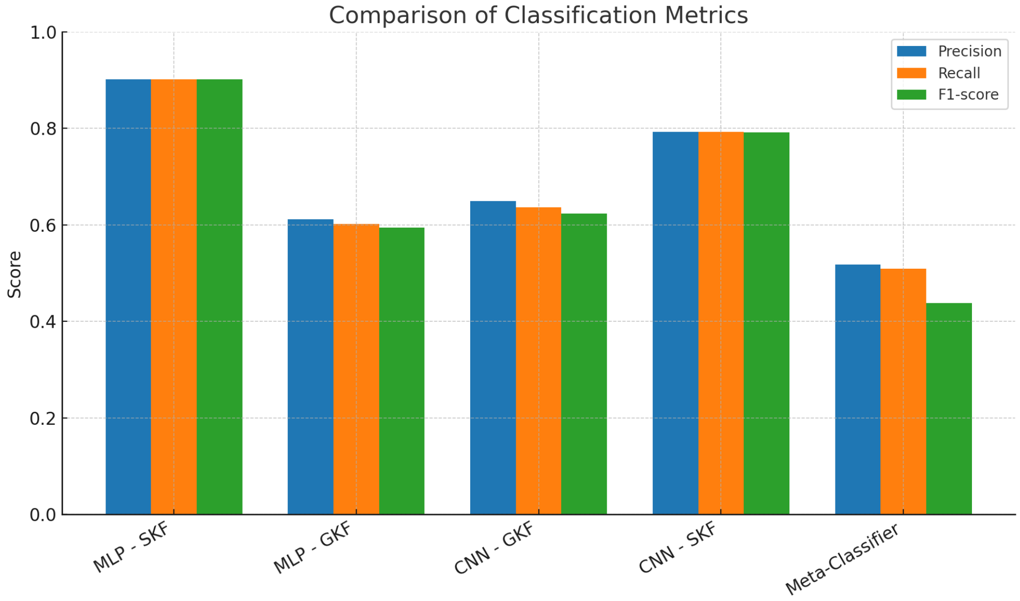
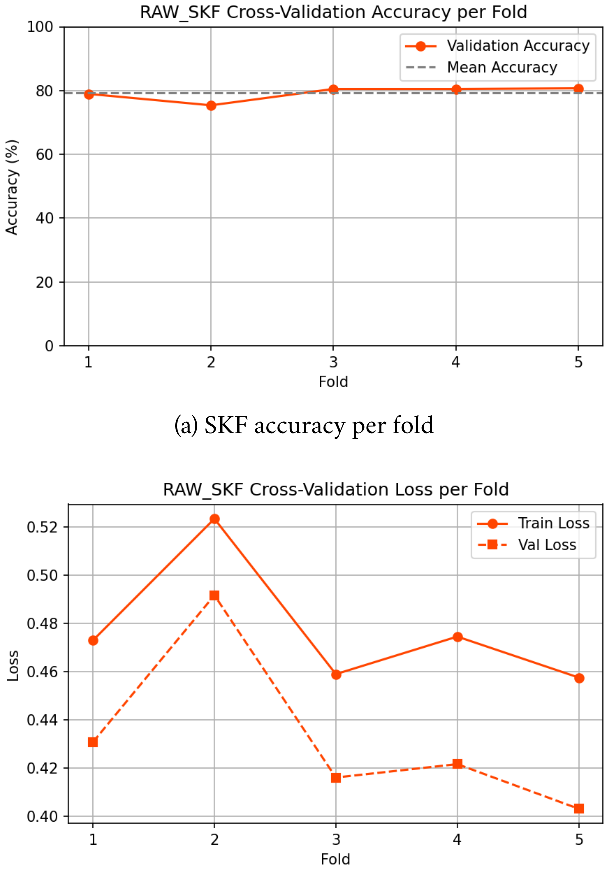


Figure 8.1: Precision, Recall, and F1-Score Across Model Setups

stable learning – most folds reached a similarly high level of accuracy, with one fold slightly lower than the others. This stability suggests that the MLP was able to learn robust patterns from the raw signals that generalized well to new data drawn from the same pool of subjects. The training and validation loss curves per fold in Figure 8.2b likewise remained close to each other, with no fold showing a dramatic divergence. In most folds, the validation loss was only marginally higher than the training loss, implying that overfitting was kept in check. In one fold, the model even achieved a validation loss comparable to (or lower than) the training loss, hinting that the network did not over-specialize on the training data. This can be attributed to regularization techniques (like dropout and early stopping) that helped the MLP maintain generality. Overall, during stratified cross-validation, the MLP model displayed a strong capacity to fit the data without severe overfitting, achieving low error on both training and validation splits.

The final aggregated confusion matrix for the stratified MLP model (Figure 8.3) reflects a high true positive rate for both classes, with the vast majority of instances on the diagonal. In other words, the MLP correctly classified most examples of both classes (rest vs. movement) when the evaluation data came from subjects it had partially seen during training. The error rates for each class were nearly equal, indicating balanced performance: the model did not favour one class over the other in the stratified scenario. This balanced confusion matrix suggests that the MLP learned to detect the distinguishing features of each class in the raw EEG signals effectively.

However, the same MLP was tested in a stricter setting using group k-fold val-



(a) SKF accuracy per fold

(b) SKF loss per fold

Figure 8.2: MLP-SKF validation accuracy and loss

idation. The accuracy (Figure 8.4) on these unseen subjects was much lower and less consistent, revealing a clear generalization gap. Some folds (i.e., some subjects) were classified more accurately than others, but all unseen-subject folds showed inferior performance compared to the stratified case. The MLP struggled especially with identifying the positive class (movement) for new individuals. The confusion matrix in Figure 8.5 from group-wise validation indicates that the model frequently misclassified movement trials as non-movement when the subject was not in the training set. In effect, the MLP had a tendency to default to predicting the “rest” class for unseen subjects, missing a large portion of actual movements – a sign of low sensitivity in the across-subject scenario.

8.1.2 MLP Model

MLP exhibited a somewhat different performance profile, though it was evaluated with the same cross-validation strategies. Under stratified k-fold validation, the MLP achieved very high accuracy consistently, with remarkably little variation between

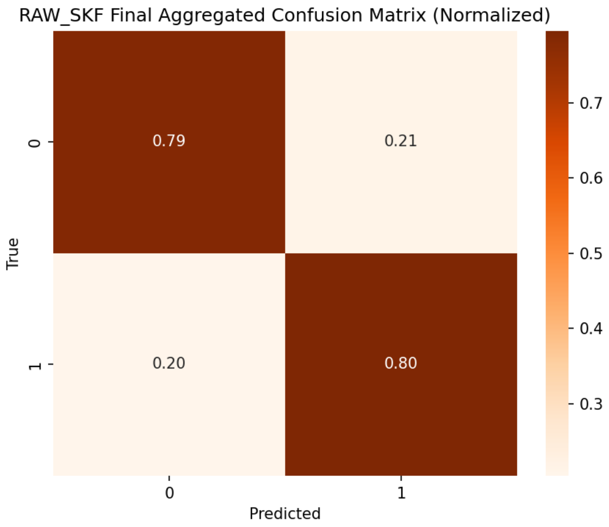
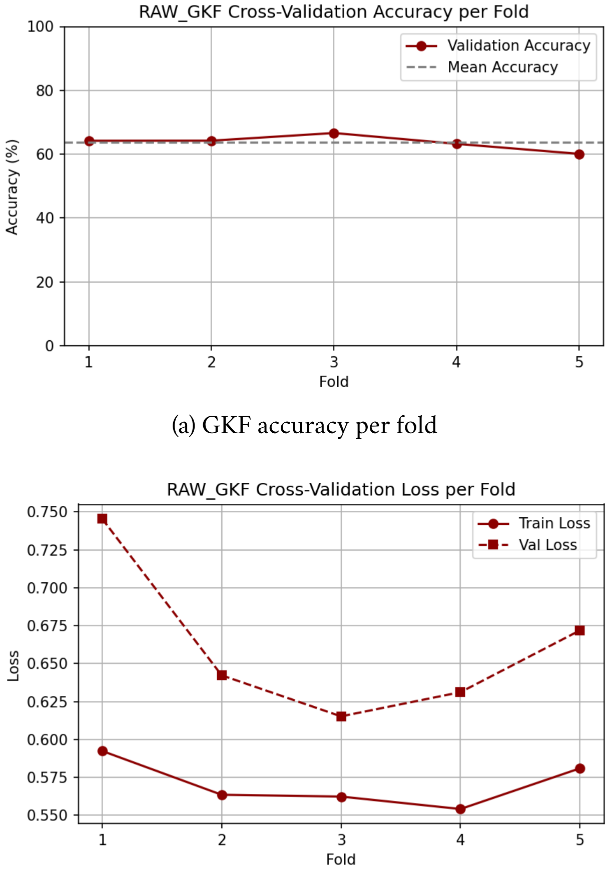


Figure 8.3: MLP-SKF aggregated confusion matrix

folds. The validation accuracy for each fold hovered at a uniformly high level, and the mean accuracy line was essentially flat across folds, indicating that no single fold fell substantially below the overall average. This consistency points to a stable learning process: the feature-based MLP found it easy to separate the two classes for every subset of data. The training vs. validation loss per fold in Figure 8.6b provides further insight into the model's dynamics. In all folds, the MLP reached a very low training loss, indicating it fit the training data almost perfectly. The validation loss was slightly higher than the training loss, but remained low in absolute terms, corresponding to the high validation accuracy. This small gap between training and validation loss suggests the MLP experienced only mild overfitting despite its capacity to memorize training features. In fact, the features themselves (such as band-power or other domain-specific attributes) likely distilled the signal in a way that reduced noise and subject-specific complexity, making the classification task more straightforward for the network. As a result, the MLP demonstrated efficient learning: it converged quickly and maintained strong performance on hold-out data in each fold.

The confusion matrix for the MLP (Figure 8.7) in the stratified scenario is nearly perfectly diagonal. It shows that the classifier correctly identified almost all instances of both classes, with only a very small fraction of misclassifications. Both the "no movement" class and the "movement" class were recognized with high fidelity. Unlike the MLP, which already had a good performance, the MLP achieved an even higher true positive rate on each class, indicating that the extracted features provided highly discriminative information for distinguishing the mental states.

When moving to group k-fold validation (unseen subjects), the MLP's perfor-



(a) GKF accuracy per fold

(b) GKF loss per fold

Figure 8.4: MLP-GKF validation accuracy and loss

mance declined, as expected, but the nature of its generalization shortfall differed somewhat from the MLP's. Overall accuracy on unseen individuals was lower than in the stratified case, underscoring that subject variability impacted the feature-based model as well. The fold-wise accuracy (Figure 8.8) in the group scenario varied more substantially, suggesting that some unseen subjects' feature distributions aligned better with the training data than others.

The aggregated confusion matrix for the group k-fold MLP in Figure 8.9 reveals an imbalanced performance across classes: the model retained a fair ability to detect the positive class (capturing a substantial portion of movement trials from new subjects), but it struggled with the negative class. About half of the instances of the rest class were incorrectly classified as movement for unseen users. This means the MLP was somewhat over-sensitive when encountering new subjects – it often predicted the occurrence of movement (the positive class) even when the person was actually at rest. Consequently, the false positive rate for movement increased on unseen subjects, reflecting a drop in specificity. On the other hand, the MLP did

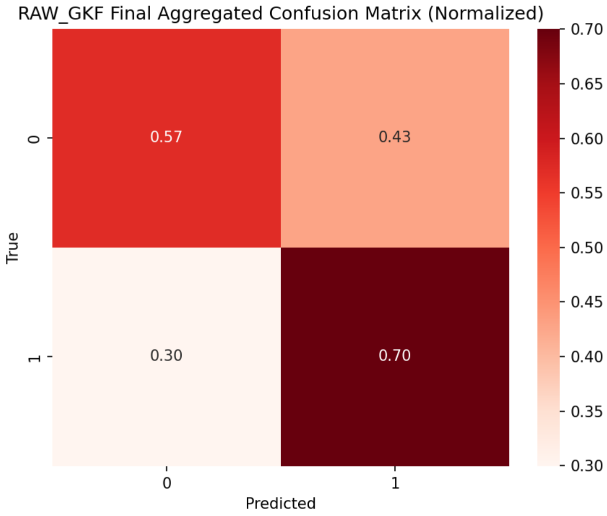


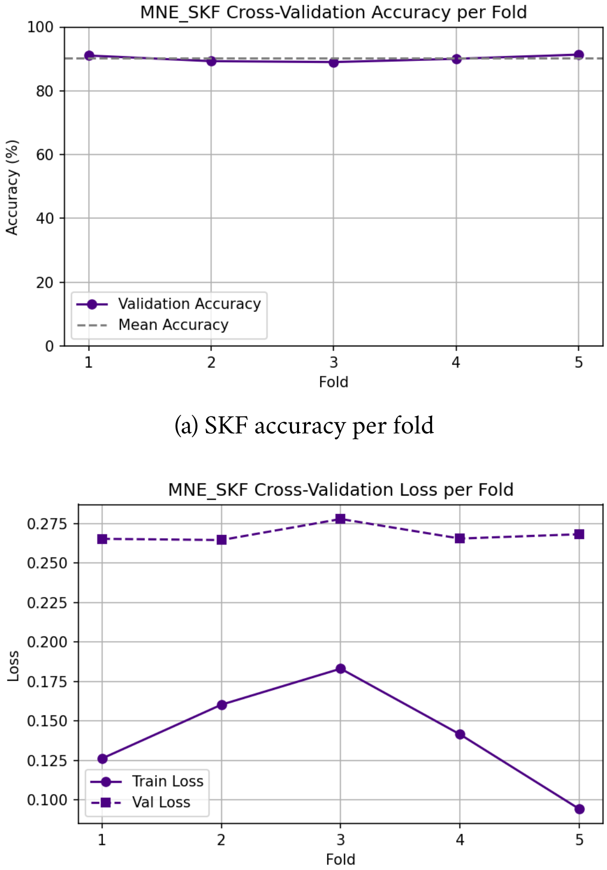
Figure 8.5: MLP-GKF aggregated confusion matrix

manage to identify a majority of the actual movement attempts in the new subjects, indicating its feature representation preserved some generalizable signatures of the task-related brain activity. Compared to the CNN, the MLP generalized slightly better in the sense that it did not completely miss the occurrences of the target class for new individuals.

8.1.3 Meta-Classifier

To capitalize on the complementary strengths of the two classifiers, a stacked ensemble was evaluated, which combined the CNN and MLP predictions for a final decision. The expectation was that the ensemble could correct the individual models' mistakes by leveraging their agreement and disagreements. The offline evaluation of this stacked model was carried out across all folds (using the same group-wise cross-validation splits, so that each test fold corresponds to an unseen subject's data, and the ensemble is trained on that fold's training portion). The resulting aggregated confusion matrix for the ensemble (combining predictions from all test folds) in Figure 8.10 provides insight into how the combined model performs on the two classes. The outcome, however, indicates that the ensemble did not achieve a balanced improvement across classes.

The ensemble's predictions were heavily skewed toward one class. The confusion matrix is dominated by a large number of correct predictions for one class and comparatively few for the other. Specifically, the ensemble correctly classified most of the Class 0 instances (non-movement), but it failed to detect the majority of Class 1 instances (movement). This means that when the CNN and MLP were combined, the resulting system leaned towards predicting the absence of movement in most



(a) SKF accuracy per fold

(b) SKF loss per fold

Figure 8.6: MLP-SKF validation accuracy and loss

cases, largely missing when a movement was actually present. This class-wise performance suggests that the ensemble model inherited the conservative tendencies of the CNN to some extent. It appears to prioritize avoiding false positives (erroneously predicting movement), but in doing so, it ends up with a very high false negative rate for the movement class. In other words, the ensemble is very confident in identifying rest intervals correctly, but it lacks sensitivity to the movement class; many attempted movement events went undetected. The net effect is that the ensemble's overall accuracy hovered around chance levels, with a strong bias in favor of the negative class. Notably, the ensemble did not significantly improve the balanced accuracy or remedy the generalization issues observed in the individual models. The hope that combining the feature-based and raw-signal models would cover each other's errors was not realized in practice. Instead, the meta-classifier likely learned that both models often signaled "rest" for unseen subjects (perhaps the CNN almost always predicted rest, and the MLP frequently did as well for certain difficult subjects), and thus the safest combined prediction was also "rest." The end

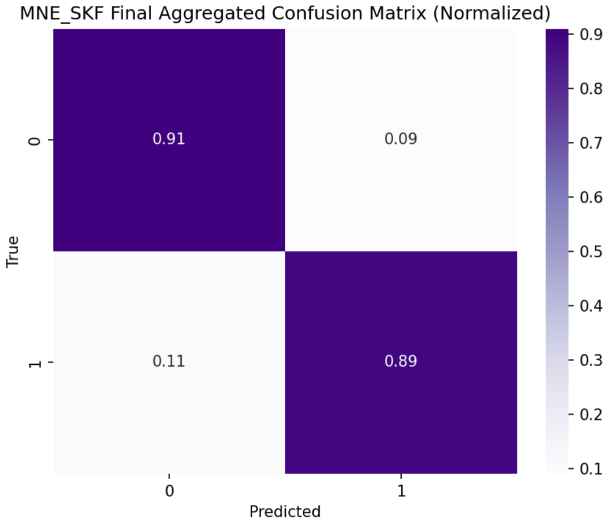


Figure 8.7: MLP-SKF aggregated confusion matrix

result is an imbalanced classifier that, while reliable in predicting one class, performs poorly on the other.

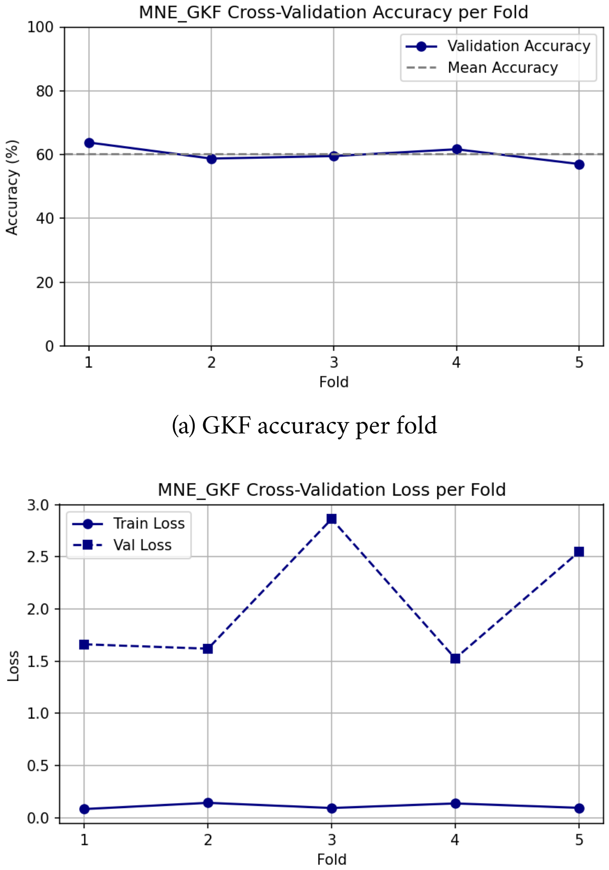
8.2 Real-Time Evaluation

A real-time evaluation was carried out to asses the CNN and MLP classifiers in a production-like environment. To make it as identical as possible to the training, the incoming data was filtered and windowed in the same way as in Section 7.6.2.1 and 7.6.2.2. To provide smoother prediction response, a sliding window with a step of 300 ms was implemented. The final prediction was obtained by averaging the class probability outputs of the last three consecutive windows and selecting the class with the highest averaged probability. This temporal smoothing reduces prediction jitter and improves stability in real-time feedback by mitigating the effect of noisy or borderline windows.

8.2.1 Classifier Performance: CNN vs MLP

In total, four subjects were tested for the real-time classifications. The first two patients were already a part of the created dataset (Section 7.4.2), while the other two represented a completely new subject. Table 8.1 (raw input) and Table 8.2 (feature input) summarize the performance metrics for each trial [1]. The *Matched* and *Unmatched* values signal how many windows were predicted during the rehabilitation

[1] The distinction between subject already featured in the dataset vs unseen subject is in the trial name (Y/N).



(a) GKF accuracy per fold

(b) GKF loss per fold

Figure 8.8: MLP-GKF validation accuracy and loss

scenario and how many before or after it. Overall, the two models achieved comparable above-chance accuracy on the subjects seen during training (hovering in the 50–57% range), but each exhibited different strengths.

Table 8.1: Evaluation metrics for each trial using raw EEG input.

Trial	Accuracy	Precision	Recall	F1	TP	TN	FP	FN	Matched	Unmatched
01Y	56.94	0.68	0.46	0.55	128	151	59	152	490	32
02Y	52.66	0.65	0.56	0.60	175	82	95	136	488	48
03N	40.94	0.72	0.32	0.44	180	132	71	379	762	105
04N	47.47	0.55	0.32	0.41	72	116	58	150	396	72

For example, in trial **01Y** (a seen subject), the CNN achieved 56.9% accuracy versus 49.6% for the MLP, with both yielding an F1-score around 0.55. The CNN was more precision-oriented—it had higher precision (0.68 vs. 0.53) by producing fewer false alarms (59 false positives vs. 113 for MLP). In contrast, the MLP showed

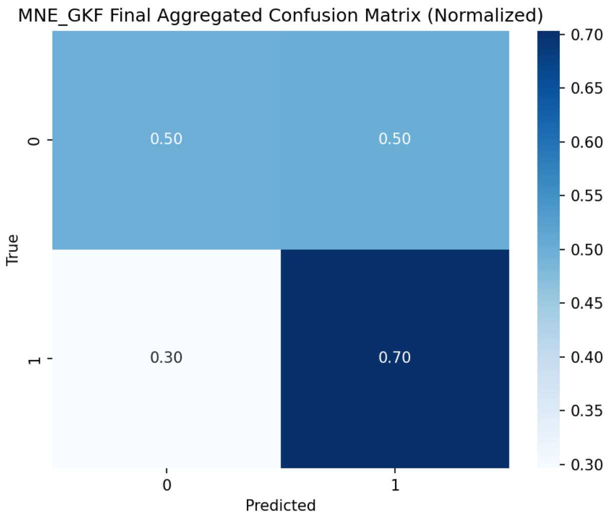


Figure 8.9: MLP-GKF aggregated confusion matrix

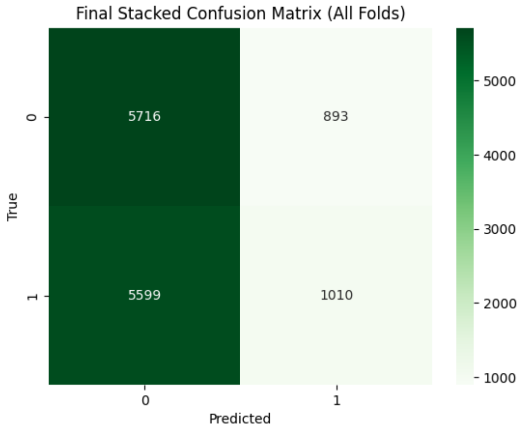


Figure 8.10: Meta-classifier confusion matrix

higher sensitivity on 01Y, with better recall (0.56 vs. CNN's 0.46) due to fewer missed detections (99 false negatives vs. 152 for CNN).

Table 8.2: Evaluation metrics for each trial using extracted features.

Trial	Accuracy	Precision	Recall	F1	TP	TN	FP	FN	Matched	Unmatched
01Y	49.64	0.53	0.56	0.55	127	82	113	99	421	25
02Y	57.27	0.60	0.91	0.72	259	5	172	25	461	55
03N	–	–	–	–	–	–	–	–	–	–
04N	42.36	0.50	0.25	0.33	57	112	57	173	399	49

A different pattern emerged in trial **02Y** (the other seen subject): here, the feature-based MLP outperformed the CNN, attaining 57.3% accuracy and an F1-score of

0.72, compared to the MLP's 52.7% accuracy and F1-score of 0.60. The MLP in 02Y achieved an exceptionally high recall of 0.91 (capturing 259 true positives with only 25 false negatives), albeit at the cost of many false positives (172, versus 95 for the MLP). The CNN's recall on 02Y was more modest (0.56 with 175 true positives), but its precision was slightly higher (0.65 vs. MLP's 0.60) thanks to fewer false positives.

These trial results indicate that neither classifier consistently dominates the other—instead, each has advantages depending on the subject and input type. The CNN tended to be more conservative (fewer false positives, higher precision), while the MLP often picked up more events (higher recall), reflecting a trade-off between specificity and sensitivity in the real-time setting.

8.2.2 Seen vs. Unseen Subjects

A clear trend in the real-time evaluation is the drop in performance when the system was tested on **unseen subjects** (trials **03N** and **04N**) versus **seen subjects** used in training (**01Y** and **02Y**). Both classifiers performed significantly worse on the new individuals.

For the CNN (raw input), accuracy fell from approximately 53–57% on seen users to only 41–47% on unseen users. Its F1-score likewise dropped from around 0.55–0.60 for seen trials to approximately 0.41–0.44 for unseen trials. The feature-based MLP showed a similar decline: on trial 04N (unseen), it reached only 42.4% accuracy with an F1-score of 0.33, compared to 49.6–57.3% accuracy (F1 0.55–0.72) on the seen subjects.

Notably, the recall *plummeted* for unseen cases, indicating many missed detections. For instance, the MLP's recall on 04N was just 0.25, meaning it caught relatively few of the attempted events (57 true positives) while missing 173 (false negatives). In contrast, on a seen subject like 02Y, the MLP's recall was 0.91 with only 25 misses. The CNN faced the same issue: on 03N and 04N, it only detected about one-third of the events (recall ~0.32). This stark performance gap between seen and unseen subjects highlights a clear generalization challenge.

Discussion 9

A diverse set of approaches was employed to reflect the interdisciplinary nature of this thesis. Due to this reality and the gradual insights gained during the development of this thesis, several questions and doubts have emerged. It is necessary to address these questions and discuss the possible interpretations, limitations, and implications of the chosen methods and their outcomes.

9.1 Credibility of scenario design

The credibility of the rehabilitation scenario (described in Section 7.4.1) needs to be addressed. The design of the scenario directly affects the labelling methodology. In the current scenario design, movement and resting states alternate in fast-repeating iterations (within tens of seconds or lower units of minutes). These quick state changes do not necessarily correspond to physiological brain states. The study of the academic articles and research papers in Sections 4.4 and 5 suggests that this design is not recognized as a standard within this area of research. A more common approach is to split the states into continuous and defined sections in a matter of minutes, well separated from one another. This approach corresponds to the physiological state of the brain much more accurately, allowing the signal to return to the baseline. Another concern stems from the definition of resting in this scenario. A well-established resting state is described by fully closed eyes, cutting off all visual information, diminishing focus, and reducing attention. In most cases, the cue for state transition is then audiovisual or depends on non-visual sensory input. This is not the case in this scenario design since the relaxation phase is in seconds, and the state transition cue is represented as a blinking diode.

The last problem is related to the mobility of the rehabilitation arm. The arm is controlled by servomotors with adjustable resistance. During dataset creation, the subject needed to exert great force to overcome the resistance. Possibly due to this, the final signal was corrupted by muscle contraction. This is outlined in Figure 9.1.

Based on data quality (Subsection 7.5.2), the C3 electrode signal exceeds a healthy EEG signal threshold. The electrode is positioned over the left motor cortex, the re-

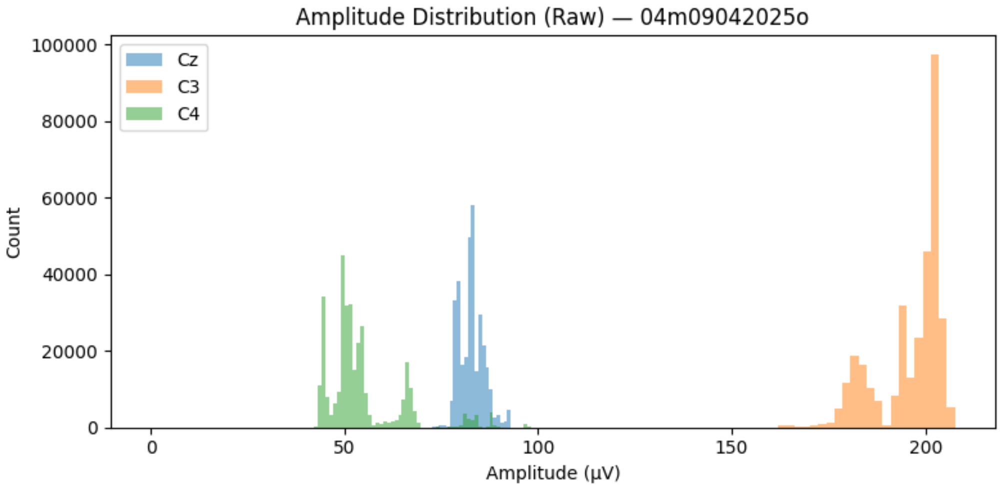


Figure 9.1: An example of a raw amplitude distribution histogram

gion responsible for initiating motor commands for right-hand movement. The 200 µV range corresponds to a possible muscle activity, further proving the hypothesis. If this were the case, then the classification (especially the minimal processing raw data CNN approach) would no longer represent pure MI classification, but rather the ability to move.

9.2 Validity of classifiers and robustness of KFold evaluation

The varying results of the classifiers based on the selected cross-validation method need to be discussed. The concern regarding the data quality was already expressed in Section 7.5 and 9.1, but the validation evaluation needs to be discussed further. The observed differences in classification performance between GroupKFold and StratifiedKFold stem from the fundamental way each method handles data splitting.

StratifiedKFold maintains class balance across folds but allows data from the same subject to appear in both training and validation sets. This can lead to overly optimistic results due to subject-specific signal leakage, especially in EEG datasets where inter-subject variability is high (see Section 7.5.8).

In contrast, **GroupKFold** enforces subject-level independence by assigning all data from a single subject to one fold only, thereby simulating a more realistic deployment scenario where the model must generalize to entirely unseen individuals. Consequently, performance metrics obtained using GroupKFold better reflect the true generalization ability of the classifier, though they may appear lower.

This highlights the importance of selecting a cross-validation strategy that aligns with the intended use case, particularly in biomedical applications, where robustness and generalizability across subjects are critical.

9.3 Limitations

A few limitations arose during the implementation process. All of them stem from the fact that the **FourMotors** rehabilitation application is developed by an external specialist. The communication sections between the parts of the proposed system and the rehabilitation software are directly dependent on the specialist's availability and the implemented features of the software. Because of this, some proposed functionality was not met and was either prepared for future work or simulated to a certain degree.

The feedback propagation from the classifier to the servomotors was not implemented. The classifier is capable of predicting the movement state in real-time; however, the interface component responsible for communicating these predictions to the rehabilitation software and adjusting the servomotors has not yet been implemented. Therefore, the proposed BCI loop is not fully closed.

Another missing functionality is regarding the real-time signal (S3, S4, S5, S6) propagation (see Section 7.4.1) to the **Unity** front-end and its VR version. To counter this gap in the development, a simulation of the signals in both versions was implemented. The synchronization is corrected by the user and is described in Appendix A.2.

The last limitation regarding the **FourMotors** software is that during the dataset creation and real-time testing, the states and their indicators were desynchronized after some time. This desynchronization was enhanced when the application was running in the background (it was designed this way in the proposed system solution of this thesis). It appears that the rehabilitation software operates on an internal time clock and experiences delays when the process does not receive sufficient CPU resources.

In the end, one physical limitation must be mentioned for potential future VR development in neurorehabilitation. With the current setup, the Head-Mounted Display (HMD) is equipped with a top strap that holds and stabilises the headset. This strap physically prevents the attachment of EEG electrodes, which must be firmly attached at specific locations in order to obtain quality data. This limitation leads to the conclusion that a new and effective way of holding the headset is needed to combine the EEG measuring with VR experience.

Conclusion 10

This thesis presented the design, implementation, and evaluation of a neurorehabilitation software system that integrates real-time EEG classification with a Unity-based rehabilitation front-end and robotic arm control. The project aimed to close the Brain-Computer Interface (BCI) loop for motor rehabilitation by developing a reliable classifier capable of distinguishing between movement intention and resting states, while providing real-time feedback through an interactive virtual and physical environment.

A novel dataset was created using the **OpenBCI** platform in conjunction with the **FourMotors** rehabilitation system. This dataset served as the foundation for evaluating several classification pipelines, including a CNN trained on minimally preprocessed raw EEG, an MLP trained on features extracted through the MNE framework, and a stacked ensemble meta-classifier combining both. Extensive preprocessing, artifact removal, and signal normalization steps were employed to ensure data quality. The evaluation results showed promising accuracy for seen subjects, but also revealed significant generalization challenges when applied to unseen individuals, underscoring the difficulty of subject-independent EEG classification in real-world BCI scenarios.

The real-time implementation demonstrated that smooth, responsive classification is achievable with adequate temporal smoothing, although the lack of robustness in unseen subject performance and incomplete feedback integration with the robotic arm limited the system's full potential. Additionally, several technical and organizational limitations—particularly regarding external software integration—highlight areas for future development.

Despite these challenges, this thesis successfully delivered a modular and extensible platform for neurorehabilitation research. The insights gained during development, especially regarding EEG preprocessing, subject variability, and classifier design, provide a solid foundation for further work. Future efforts should focus on improving classifier generalization, fully integrating feedback loops, and testing the system with patients undergoing rehabilitation.

User Guide A

This appendix is intended for everybody who wants to either use/test the created neurorehabilitation system or to continue developing or researching this area of expertise. The chapter is divided in the same way as the proposed solution. It must be noted that all applications in the proposed BCI system are intended either for Windows or Android (VR case) and were not tested in a Linux environment.

A.1 FourMotors setup

A public release version of **FourMotors** software is not available, but it is installed on the computer in the neurolab at UWB and therefore available for anybody with access. The application runs as a normal `.EXE` file. A notification is needed here for a reader - the application runs two separate components that need to be executed in a given order. This condition is sometimes not satisfied, resulting in infinite loading. It is advised to restart the application if this happens.

First thing the user needs to do after loading is to check the connection to the server in the upper left corner, and then to the machine via the serial port in the bottom right corner (see Figure 7.2 for graphical representation). Then the user needs to refer drivers to the home position through **Referovat drivery** button. For setting up the scenario, use the **Nastavení** button located in the upper right corner. Set the parameters as the user deems appropriate, or as in this thesis in Section 7.4.1. The last thing is to choose a rehabilitation curve, number of repetitions and start the scenario by clicking the button **Start CVIČENÍ**. **Once the visualization of the arm is on the path of the curve, the setup is completed.**

A.2 Unity front-end

The standalone front-end was developed in **Unity** version **2022.3.21f1**, which is a LTS version widely used. It is installed through *Unity Hub*, downloadable from the **Unity** official website. The whole **Unity** project is attached to the electronic version of this thesis or is located in a private GitHub repository[1]. Add the project to the Hub and open it. All the packages and meta files should download and install automatically (keep this in mind since the first opening of the project takes significantly longer). The project, like this, is ready to be developed further or run in an editor environment. If the user wishes to build the project into a standalone application, he/she needs to go to *File* → *Build Settings* and **Build**.

A.2.1 Usage

It is advised to run the **FourMotors** application first before running the **Unity** front-end. The application is run through a built `.EXE` file. The interaction with the UI is done, in this case, through mouse clicks. To simulate the scenario synchronization (since the **FourMotors** does not have the functionality to forward the scenario signal mentioned in Table 7.2) user has to manually start the rehabilitation by clicking on the **Start** button. In order to successfully synchronize the states, the user needs to click the button **only after the FourMotors setup was completed** (see Appendix A.1).

A.3 VR front-end

The VR uses the same version of **Unity** as its standalone version (**2022.3.21f1**); therefore, the installation process remains the same. A complement to the installation is a module for `Android Build Support`. This module can be selected directly when installing the new version or later added in the Hub by going into the *Installs* section, selecting the appropriate version and clicking on the associated gear wheel. All the other VR packages are presented in the project and will be automatically downloaded with the first time opening of the project. The VR version resides in the same GitHub repository as its PC counterpart, only on a different branch. Navigating in the project and building the project is exactly the same as in the Appendix section A.2.

> When building the VR version, the developer **must** switch the target platform to `Android` in the *Build Settings*. The result of this build will be an `.APK` file.

The built `.APK` file must be installed into the VR headset through **SideQuest** software. A registration is needed in order to download and use the software. The reader is advised to check the official **SideQuest** websites for more detailed information regarding the installation.

[1] The repository is private because of the copyright law regarding the assets used in the project. For access, contact `tkment@students.zcu.cz`.

A.3.1 Usage

The application is launched in the VR headset as any other VR application. The user begins in AR with a **Start** button floating and following his/hers view. The user sees the real world through the HMD cameras and can position himself/herself in front of the rehabilitation robot or finish the **FourMotors** setup.

The interaction with the world in done through index fingertip contact on both hands. After pressing the **Start** button, the user is seamlessly transitioned into the VR environment. The dojo environment can be explored, but caution in real world is advised. The visualization of the UI changed but the functionality remains the same as in Appendix section A.2.1. On the left, there is a color picker with adjustable sliders. On the right, there is scenario parameter setting panel with another **Start** button which synchronizes the VR application with **FourMotors** software (same as in Appendix section A.2.1).

A.4 Classifier project

Prior to the installation and launch of the project, it is necessary to prepare the working directory (preparation of source code and data). The source code is available in the GitHub repository:

https://github.com/Tkmentt/DP-Classification

It contains a detailed `README` describing installation process and code overview. Working directory with the project source code can be obtained as follows:

```
C:\Workspace>git clone
    https://github.com/Tkmentt/DP-Classification.git

C:\Workspace> cd DP-Classification-main

C:\Workspace\eeg-motion-detection>
```

It is also possible to download the project source code directly, for example in `.zip` format, by clicking the **Download** button and selecting the appropriate format. The source folder must then be unzipped to the desired location and then switch the current working directory to the unzipped folder from within the operating system. The dataset that was used for the training (Section 7.4.2) can be downloaded on here:

https://zenodo.org/records/15399490

The packed data folder must be unzipped into the `src` directory of the project. The project directory structure should look like this:

```
C:\Workspace\DP-Classification-main\src>dir /b
classification
data
preprocessing
testing
utils
requirements.txt
train_MNE.py
train_old.py
train_raw.py
train_stack.py
```

The project dependencies can be installed either through Conda package manager or directly using the Python programming language. Both choices are described in README. Conda is the recommended installation method if the user wants to use *CUDA GPU*.

It is possible to use the *Anaconda* distribution or the *Miniconda* distribution.

After installation, a new Conda environment needs to be created (to avoid conflicts between dependencies), in the following example the environment used is named *eeg*, any arbitrary alias can be used.

```
C:\Workspace\DP-Classification-main\src>conda create -n eeg
    python=3.10
```

The created environment must be activated.

```
C:\Workspace\DP-Classification-main\src>conda activate eeg

(eeg) C:\Workspace\DP-Classification-main\src>
```

Next, the user needs to install dependencies to enable running Keras/Tensorflow training models using CUDA GPUs. This step can be skipped if this feature is not required.

```
(eeg) C:\Workspace\DP-Classification-main\src> conda install -n eeg
    -c conda-forge cudatoolkit=11.3 cudnn=8.1.0

(eeg) C:\Workspace\DP-Classification-main\src> conda install -n eeg
    -c nvidia cuda-nvcc=11.3
```

The last step is to install the Python packages from the `requirements.txt` file.

```
(eeg) C:\Workspace\DP-Classification-main\src>pip install -r
    requirements.txt
```

Bibliography

[AAM15] ABDULKADER, Sarah N.; ATIA, Ayman; MOSTAFA, Mostafa-Sami M. Brain computer interfacing: Applications and challenges. *Egyptian Informatics Journal*. 2015, vol. 16, no. 2, pp. 213–230. ISSN 1110-8665. Available from DOI: `https://doi.org/10.1016/j.eij.2015.06.002`.

[And04] ANDERSON, John R. *Cognitive Psychology*. 6th ed. New York, NY: Worth, 2004.

[Ang18] ANGELI, C. A. et al. Recovery of over-ground walking after chronic motor complete spinal cord injury. *The New England Journal of Medicine*. 2018, vol. 379, pp. 1244–1250.

[Aur+04] AURLIEN, H et al. EEG background activity described by a large computerized database. *Clin. Neurophysiol.* 2004, vol. 115, no. 3, pp. 665–673.

[BSB86] BARTH, D S; SUTHERLING, W; BEATTY, J. Intracellular currents of interictal penicillin spikes: evidence from neuromagnetic mapping. *Brain Res.* 1986, vol. 368, no. 1, pp. 36–48.

[BCP20] BEAR, Mark F.; CONNORS, Barry W.; PARADISO, Michael A. *Neuroscience: Exploring the Brain*. 4th. Jones & Bartlett Learning, 2020.

[Big+15] BIGDELY-SHAMLO, Nima; MULLEN, Tim; KOTHE, Christian; SU, Kyung-Min; ROBBINS, Kay A. The PREP pipeline: standardized preprocessing for large-scale EEG analysis. *Frontiers in Neuroinformatics*. 2015, vol. 9, p. 16. Available from DOI: `10.3389/fninf.2015.00016`.

[Bir+99] BIRBAUMER, N et al. A spelling device for the paralysed. *Nature*. 1999, vol. 398, no. 6725, pp. 297–298.

[BC07] BIRBAUMER, N.; COHEN, L. G. Brain-computer interfaces: communication and restoration of movement in paralysis. *The Journal of Physiology*. 2007, vol. 579, pp. 621–636.

[Blá+24] BLÁZQUEZ-GONZÁLEZ, Patricia et al. Efficacy of virtual reality on Health Literacy in patients with stroke: randomized pilot clinical trial. *Revista Científica de la Sociedad de Enfermería Neurológica (English ed.)* 2024, p. 100161. ISSN 2530-299X. Available from DOI: `https://doi.org/10.1016/j.sedeng.2024.100161`.

[BC93] BLISS, T V; COLLINGRIDGE, G L. A synaptic model of memory: long-term potentiation in the hippocampus. *Nature*. 1993, vol. 361, no. 6407, pp. 31–39.

[Buz12] BUZSÁKI, G. et al. The origin of extracellular fields and currents—EEG, ECoG, LFP and spikes. *Nature Reviews Neuroscience*. 2012, vol. 13, pp. 407–420.

[CP06] CAHN, B. Rael; POLICH, John. Meditation states and traits: EEG, ERP, and neuroimaging studies. *Psychological bulletin*. 2006, vol. 132 2, pp. 180–211. Available also from: `https://api.semanticscholar.org/CorpusID:2151810`.

[Can+23] CANTILLO-NEGRETE, Jessica; CARINO-ESCOBAR, Ruben I.; ORTEGA-ROBLES, Emmanuel; ARIAS-CARRIÓN, Oscar. A comprehensive guide to BCI-based stroke neurorehabilitation interventions. *MethodsX*. 2023, vol. 11, p. 102452. ISSN 2215-0161. Available from DOI: `https://doi.org/10.1016/j.mex.2023.102452`.

[Cio+24] CIOFFI, Elena et al. EEG-based sensorimotor neurofeedback for motor neurorehabilitation in children and adults: A scoping review. *Clinical Neurophysiology*. 2024, vol. 167, pp. 143–166. ISSN 1388-2457. Available from DOI: `https://doi.org/10.1016/j.clinph.2024.08.009`.

[CC83] COHEN, David; CUFFIN, B Neil. Demonstration of useful differences between magnetoencephalogram and electroencephalogram. *Electroencephalogr. Clin. Neurophysiol.* 1983, vol. 56, no. 1, pp. 38–51.

[Dec96] DECETY, J. Do imagined and executed actions share the same neural substrate? *Brain Res. Cogn. Brain Res.* 1996, vol. 3, no. 2, pp. 87–93.

[Dec+93] DECETY, J; JEANNEROD, M; DUROZARD, D; BAVEREL, G. Central activation of autonomic effectors during mental simulation of motor actions in man. *J. Physiol.* 1993, vol. 461, no. 1, pp. 549–563.

[DSM07] DELORME, Arnaud; SEJNOWSKI, Terrence; MAKEIG, Scott. Enhanced detection of artifacts in EEG data using higher-order statistics and independent component analysis. *Neuroimage*. 2007, vol. 34, no. 4, pp. 1443–1449.

[Dob05] DOBKIN, B. H. Clinical practice. Rehabilitation after stroke. *The New England Journal of Medicine*. 2005, vol. 352, pp. 1677–1684.

[Doy00] DOYA, Kenji. Complementary roles of basal ganglia and cerebellum in learning and motor control. *Current Opinion in Neurobiology*. 2000, vol. 10, no. 6, pp. 732–739.

[DHS01] DUDA, Richard O.; HART, Peter E.; STORK, David G. *Pattern Classification*. Wiley-Interscience, 2001.

[Dun16] DUNCAN, P. W. et al. Neurostimulation in motor rehabilitation after stroke. *Neurorehabilitation and Neural Repair*. 2016, vol. 30, pp. 1001–1011.

[FRV01] FESHCHENKO, Vladimir A; REINSEL, Ruth A; VESELIS, Robert A. Multiplicity of the α rhythm in normal humans. *J. Clin. Neurophysiol.* 2001, vol. 18, no. 4, pp. 331–344.

[Gul+15] GULATI, Tanuj et al. Robust neuroprosthetic control from the stroke perilesional cortex. *J. Neurosci.* 2015, vol. 35, no. 22, pp. 8653–8661.

[Har17] HARDIMAN, O. et al. Amyotrophic lateral sclerosis. *Nature Reviews Disease Primers*. 2017, vol. 3, p. 17085.

[Heb49] HEBB, Donald O. *The Organization of Behavior: A Neuropsychological Theory*. Wiley, 1949.

[Hil01] HILLE, Bertil. *Ion Channels of Excitable Membranes*. 3rd. Sinauer Associates, 2001.

[Hoc+12] HOCHBERG, Leigh R et al. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. *Nature*. 2012, vol. 485, no. 7398, pp. 372–375.

[Ina+23] INAMOTO, Takashi et al. Motor-related mu/beta rhythm in older adults: A comprehensive review. *Brain Sci.* 2023, vol. 13, no. 5.

[IMM10] ITURRATE, I.; MONTESANO, L.; MINGUEZ, J. Robot reinforcement learning using EEG-based reward signals. In: *2010 IEEE International Conference on Robotics and Automation*. 2010, pp. 4822–4829. Available from DOI: `10.1109/ROBOT.2010.5509734`.

[JBT19] JIANG, Xiao; BIAN, Gui-Bin; TIAN, Zean. Removal of artifacts from EEG signals: A review. *Sensors (Basel)*. 2019, vol. 19, no. 5, p. 987.

[JR02] JOHANSEN-BERG, H.; RUSHWORTH, M. F. Functional imaging of motor control and recovery after brain injury. *Current Opinion in Neurobiology*. 2002, vol. 12, pp. 685–690.

[Joh+02] JOHANSEN-BERG, Heidi et al. The role of ipsilateral premotor cortex in hand movement after stroke. *Proceedings of the National Academy of Sciences*. 2002, vol. 99, no. 22, pp. 14518–14523.

[Jun+00] JUNG, Tzyy-Ping et al. Removing electroencephalographic artifacts by blind source separation. *Psychophysiology*. 2000, vol. 37, no. 2, pp. 163–178.

[KL15] KALIA, L. V.; LANG, A. E. Parkinson's disease. *The Lancet*. 2015, vol. 386, pp. 896–912.

[Kan+13] KANDEL, Eric R.; SCHWARTZ, James H.; JESSELL, Thomas M.; SIEGELBAUM, Steven A.; HUDSPETH, A. J. *Principles of Neural Science*. 5th. McGraw-Hill Education, 2013.

[KBA21] KAWALA-STERNIUK, Aleksandra; BROWARSKA, Natalia; AL-BAKRI, Amir. Summary of over Fifty Years with Brain-Computer Interfaces—A Review. *Brain Sciences*. 2021, vol. 11. Available from DOI: `10.3390/brainsci11010043`.

[Kle+99] KLEM, G H; LÜDERS, H O; JASPER, H H; ELGER, C. The ten-twenty electrode system of the International Federation. The International Federation of Clinical Neurophysiology. *Electroencephalography and clinical neurophysiology. Supplement*. 1999, vol. 52, pp. 3–6.

[Kod23] KODERA, Jakub. *Detekce pohybu z EEG dat*. Plzeň, 2023. Diplomová práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd.

[Kod+23] KODERA, Jakub et al. *EEG motor imagery*. Zenodo, 2023.

[KG11] KOLB, Bryan; GIBB, Robbin. *Brain Plasticity and Behavior*. Vol. 62. Annual Review of Psychology, 2011.

[KM19] KOTHE, Christian; MEDINE, David. *Introduction; Labstreaminglayer 1.13 documentation* [`https://labstreaminglayer.readthedocs.io/info/intro.html`]. 2019. [Accessed 01-05-2025].

[Kru+16] KRUCOFF, Max O; RAHIMPOUR, Shervin; SLUTZKY, Marc W; EDGERTON, V Reggie; TURNER, Dennis A. Enhancing nervous system recovery through neurobiologics, neural interface training, and neurorehabilitation. *Front. Neurosci.* 2016, vol. 10, p. 584.

[LBK11] LANGHORNE, Peter; BERNHARDT, Julie; KWAKKEL, Gert. Stroke rehabilitation. *The Lancet*. 2011, vol. 377, no. 9778, pp. 1693–1702.

[Law+18] LAWHERN, Vernon J. et al. EEGNet: A compact convolutional neural network for EEG-based brain–computer interfaces. *Journal of Neural Engineering*. 2018, vol. 15, no. 5, p. 056013. Available from DOI: `10.1088/1741-2552/aace8c`.

[Lea+14] LEAMY, Darren J et al. An exploration of EEG features during recovery following stroke - implications for BCI-mediated neurorehabilitation therapy. *J. Neuroeng. Rehabil.* 2014, vol. 11, no. 1, p. 9.

[Lev+00] LEVINE, S P et al. A direct brain interface based on event-related potentials. *IEEE Trans. Rehabil. Eng.* 2000, vol. 8, no. 2, pp. 180–185.

[LC23] LIN, Chun-Ling; CHEN, Liang-Ting. Improvement of brain–computer interface in motor imagery training through the designing of a dynamic experiment and FBCSP. *Heliyon*. 2023, vol. 9, no. 3, e13745. ISSN 2405-8440. Available from DOI: `https://doi.org/10.1016/j.heliyon.2023.e13745`.

[Lo+24] LO, Yu Tung et al. Neural Interface-Based Motor Neuroprosthesis in Poststroke Upper Limb Neurorehabilitation: An Individual Patient Data Meta-analysis. *Archives of Physical Medicine and Rehabilitation*. 2024, vol. 105, no. 12, pp. 2336–2349. ISSN 0003-9993. Available from DOI: `https://doi.org/10.1016/j.apmr.2024.04.001`.

[Log+01] LOGOTHETIS, N K; PAULS, J; AUGATH, M; TRINATH, T; OELTERMANN, A. Neurophysiological investigation of the basis of the fMRI signal. *Nature*. 2001, vol. 412, no. 6843, pp. 150–157.

[Lop10] LOPES DA SILVA, F. EEG: Origin and Measurement. In: MULERT, C; LEMIEUX, L (eds.). *EEG-fMRI: Physiological Basic, Technique, and Applications*. New York: Springer, 2010, pp. 19–38.

[Luc05] LUCK, S. *Comparison with Behavioral Measures". An Introduction to the Event-Related Potential Technique*. MIT Press, 2005.

[Luc14] LUCK, S. J. *An Introduction to the Event-Related Potential Technique*. MIT Press, 2014.

[Maa+17] MAAS, Andrew I. R. et al. Traumatic brain injury: integrated approaches to improve prevention, clinical care, and research. *The Lancet Neurology*. 2017, vol. 16, no. 12, pp. 987–1048.

[Mak+11] MAK, J N et al. Optimizing the P300-based brain–computer interface: current status, limitations and future directions. *Journal of Neural Engineering*. 2011, vol. 8, no. 2, p. 025003. Available from DOI: `10.1088/1741-2560/8/2/025003`.

[MPT21] MATHE, Mariyadasu; PADMAJA, M.; TIRUMALA KRISHNA, Battula. Intelligent approach for artifacts removal from EEG signal using heuristic-based convolutional neural network. *Biomedical Signal Processing and Control*. 2021, vol. 70, p. 102935. ISSN 1746-8094. Available from DOI: `https://doi.org/10.1016/j.bspc.2021.102935`.

[MM12] MICHEL, C. M.; MURRAY, M. M. EEG source imaging. *Clinical Neurophysiology*. 2012, vol. 123, pp. 621–636.

[Mra+17] MRACHACZ-KERSTING, N et al. The effect of type of afferent feedback timed with motor imagery on the induction of cortical plasticity. *Brain Res.* 2017, vol. 1674, pp. 91–100.

[NG12] NICOLAS-ALONSO, Luis Fernando; GOMEZ-GIL, Jaime. Brain computer interfaces, a review. *Sensors (Basel)*. 2012, vol. 12, no. 2, pp. 1211–1279.

[NS05] NIEDERMEYER, E.; SILVA, F. L. da. *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields*. Lippincott Williams & Wilkins, 2005.

[NL04] NIEDERMEYER, Ernst; LOPES DA SILVA, F H (eds.). *Electroencephalography*. 5th ed. Philadelphia, PA: Lippincott Williams and Wilkins, 2004.

[Nud13] NUDO, Randolph J. Recovery after brain injury: mechanisms and principles. *Frontiers in Human Neuroscience*. 2013, vol. 7, p. 887.

[NP91] NUNEZ, Paul L; PILGREEN, Kenneth L. The spline-Laplacian in clinical neurophysiology. *J. Clin. Neurophysiol.* 1991, vol. 8, no. 4, pp. 397–413.

[ONe+18] O'NEIL, Owen et al. Virtual Reality for Neurorehabilitation: Insights From 3 European Clinics. *PM&R*. 2018, vol. 10, no. 9, Supplement 2, S198–S206. ISSN 1934-1482. Available from DOI: `https://doi.org/10.1016/j.pmrj.2018.08.375`. Innovations Influencing Physical Medicine and Rehabilitation.

[Ope25] OPENBCI. *Setting up for EEG | OpenBCI Documentation — docs.openbci.com* [`https://docs.openbci.com/GettingStarted/Biosensing-Setups/EEGSetup/`]. 2025. [Accessed 01-05-2025].

[Opi21] OPIE, Nicholas Lachlan. The StentrodeTM Neural Interface System. In: 2021. Available also from: `https://api.semanticscholar.org/CorpusID:234102889`.

[PL99] PFURTSCHELLER, G; LOPES DA SILVA, F H. Event-related EEG/MEG synchronization and resynchronization: basic principles. *Clin. Neurophysiol.* 1999, vol. 110, no. 11, pp. 1842–1857.

[PM24] PHAM, Duc Thien; MOUČEK, Roman. Automatic Motor Imagery Classification by CNN-Transformer-LSTM Using Multi-Channel EEG Signals. In: 2024, pp. 4555–4562. ISBN 9781643685489. Available from DOI: `10.3233/FAIA241048`.

[Pra+24] PRATS-BISBE, Alba et al. Virtual Reality–Based Neurorehabilitation Support Tool for People With Cognitive Impairments Resulting From an Acquired Brain Injury: Usability and Feasibility Study. *JMIR Neurotech*. 2024, vol. 3, e50538. ISSN 2817-092X. Available from DOI: `10.2196/50538`.

[Pur+18] PURVES, Dale et al. *Neuroscience*. 6th. Oxford University Press, 2018.

[Red+24] REDWAN, Sadi Md.; UDDIN, Md Palash; ULHAQ, Anwaar; SHARIF, Muhammad Imran; KRISHNAMOORTHY, Govind. Power spectral density-based resting-state EEG classification of first-episode psychosis. *Scientific Reports*. 2024, vol. 14, no. 1, p. 15154. Available from DOI: `10.1038/s41598-024-66110-0`.

[Ric+20] RICHARD, Nelly et al. Steady-state visual evoked potential temporal dynamics reveal correlates of cognitive decline. *Clinical Neurophysiology*. 2020, vol. 131, no. 4, pp. 836–846. ISSN 1388-2457. Available from DOI: `https://doi.org/10.1016/j.clinph.2020.01.010`.

[Roj+18] ROJAS, Gonzalo et al. Study of Resting-State Functional Connectivity Networks Using EEG Electrodes Position As Seed. *Frontiers in Neuroscience*. 2018, vol. 12. Available from DOI: `10.3389/fnins.2018.00235`.

[Sán+25] SÁNCHEZ-GIL, Juan J.; SÁEZ-MANZANO, Aurora; LÓPEZ-LUQUE, Rafael; OCHOA-SEPÚLVEDA, Juan-José; CAÑETE-CARMONA, Eduardo. Design and validation of PACTUS: A gamified electronic device for stroke rehabilitation. *Computer Methods and Programs in Biomedicine*. 2025, vol. 260, p. 108563. ISSN 0169-2607. Available from DOI: `https://doi.org/10.1016/j.cmpb.2024.108563`.

[SD06] SELLERS, Eric W; DONCHIN, Emanuel. A P300-based brain-computer interface: initial tests by ALS patients. *Clin. Neurophysiol.* 2006, vol. 117, no. 3, pp. 538–548.

[SD04] SERRUYA, Mijail; DONOGHUE, John. DESIGN PRINCIPLES OF A NEUROMOTOR PROSTHETIC DEVICE. In: *Neuroprosthetics*. 2004, pp. 1158–1196. Available from DOI: `10.1142/9789812561763_0040`.

[Sil+11] SILVONI, Stefano et al. Brain-computer interface in stroke: a review of progress. *Clin. EEG Neurosci.* 2011, vol. 42, no. 4, pp. 245–252.

[Siv17] SIVAKUMAR, Shruthi. *The Wave - The characteristics of an EEG — Firstclass — firstclassmed.com* [`https://www.firstclassmed.com/articles/2017/eeg-waves`]. 2017. [Accessed 11-05-2025].

[Soe15] SOEKADAR, S. R. et al. Brain–machine interfaces in neurorehabilitation of stroke. *Neurobiology of Disease*. 2015, vol. 83, pp. 172–179.

[Spe+23] SPECHT, Julian; STEGMANN, Barbara; GROSS, Hanna; KRAKOW, Karsten. Cognitive Training With Head-Mounted Display Virtual Reality in Neurorehabilitation: Pilot Randomized Controlled Trial. *JMIR Serious Games*. 2023, vol. 11, e45816. ISSN 2291-9279. Available from DOI: `10.2196/45816`.

[Sri+96] SRINIVASAN, R; NUNEZ, P L; TUCKER, D M; SILBERSTEIN, R B; CADUSCH, P J. Spatial sampling and filtering of EEG with spline laplacians to estimate cortical potentials. *Brain Topogr.* 1996, vol. 8, no. 4, pp. 355–366.

[Tat14] TATUM, W O. Ellen R. Grass Lecture: extraordinary EEG. *The Neurodiagnostic journal*. 2014, vol. 54, pp. 3–21.

[Tun+13] TUNG, Sau Wai et al. Motor imagery BCI for upper limb stroke rehabilitation: An evaluation of the EEG recordings using coherence analysis. In: *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. Osaka: IEEE, 2013.

[WM92] WANG, Youde; MORGAN, William P. The effect of imagery perspectives on the psychophysiological responses to imagined exercise. *Behav. Brain Res.* 1992, vol. 52, no. 2, pp. 167–174.

[Wil+21] WILLETT, Francis R; AVANSINO, Donald T; HOCHBERG, Leigh R; HENDERSON, Jaimie M; SHENOY, Krishna V. High-performance brain-to-text communication via handwriting. *Nature*. 2021, vol. 593, no. 7858, pp. 249–254.

[Wil99] WILLINGHAM, Daniel. The Neural Basis of Motor-Skill Learning. *Current Directions in Psychological Science - CURR DIRECTIONS PSYCHOL SCI*. 1999, vol. 8, pp. 178–182. Available from DOI: `10.1111/1467-8721.00042`.

[Wol+02] WOLPAW, J. R.; BIRBAUMER, N.; MCFARLAND, D. J.; PFURTSCHELLER, G.; VAUGHAN, T. M. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*. 2002, vol. 113, pp. 767–791.

[Xia+20] XIANG, Jing et al. Kurtosis and skewness of high-frequency brain signals are altered in paediatric epilepsy. *Brain Commun.* 2020, vol. 2, no. 1.

[YD16] YTTRI, Eric A; DUDMAN, Joshua T. Opponent and bidirectional control of movement velocity in the basal ganglia. *Nature*. 2016, vol. 533, no. 7603, pp. 402–406.

List of Acronyms

ALS Amyotrophic Lateral Sclerosis 15, 31

API Application Programming Interface 64

AR Augmented Reality 51, 99

BCI Brain-Computer Interface 5, 8, 10, 13–17, 25–31, 33, 34, 37, 41, 73, 74, 93, 95, 113

BMI Brain-Machine Interface 34

BOLD Blood-Oxygen-Level Dependent 29

BSS Blind Source Separation 24

CAR Common Average Reference 70

CIMT Constraint-Induced Movement Therapy 10, 14

CNN Convolutional Neural Network 5, 24, 38, 65, 67, 68, 76, 85–90, 92, 95

CSP Common Spatial Pattern ii, 29, 67, 68, 73, 75

CSV Comma-Separated Values 43, 55

DBS Deep Brain Stimulation 10, 15

DC Direct current 70

ECG Electrocardiography 20

ECoG Electrocorticography 26, 28

EEG Electroencephalography 5, 13, 14, 17–21, 23, 24, 26, 28–31, 33, 37, 39, 41–43, 53–55, 57–60, 62, 63, 67–76, 81, 88, 91–93, 95, 113, 115

EES Epidural Electrical Stimulation 16

EMG Electromyography 20

EOG Electrooculography 20

ERD Event-Related Desynchronization 54

ERF Event-Related Field 29

ERP Event-Related Potential 24, 29–31

ERS Event-Related Synchronization 54

FBCSP Filter Bank Common Spatial Pattern 29

FES Functional Electrical Stimulation 15, 34

FFT Fast Fourier transform 41

FIRDA Frontal Intermittent Rhythmic Delta 22

fMRI functional MRI 14, 18, 29

fNIRS Functional Near-Infrared Spectroscopy 26

GKF Group K-Fold 80, 84, 85, 88, 89, 114

GPIO General-Purpose Input/Output 55

GPU Graphics Processing Unit 64

GUI Graphical User Interface 41, 69

HMD Head-Mounted Display 34, 51, 93, 99

ICA Independent Component Analysis 24

IPD Individual Patient Data 34

LDA Linear Discriminant Analysis 30

LIS Locked-in Syndrome 31

LSL Lab Streaming Layer 37, 43, 67

LSTM Long-Short Term Memory 74, 75

LTD Long-Term Depression 8, 13

LTP Long-Term Potentiation 8, 13

LTS Long-Term Support 97

M1 Primary Motor Cortex 9, 10

MEG Magnetoencephalography 18, 26, 28, 29

MI Motor Imagery 13, 39, 53, 54, 92

MLP Multilayer Perceptron 5, 67, 68, 70, 71, 74–76, 80–90, 95, 114

MS Multiple Sclerosis 15

OIRDA Occipital Intermittent Rhythmic Delta 22

PCA Principal Component Analysis 24

PD Parkinson's Disease 15

PGA Programmable Gain Amplifier 69

PSD Power Spectral Density 24, 58–60, 63, 73, 74, 113

ReLU Rectified Linear Unit 75, 76

RF Random Forest 68

RMS Root Mean Square 59

RNN Recurrent Neural Network 24, 27

RTC Randomized Controlled Trial 34

SCI Spinal Cord Injury 5, 15, 16

SCPs Slow Cortical Potentials 31

SKF Stratified K-Fold 80, 82, 83, 86, 87, 114

SMA Supplementary Motor Area 10, 11

SSVEP Steady-State Visually Evoked Potential 30, 31

STFT Short-Time Fourier Transform 74

SVMs Support Vector Machines 24

TBI Traumatic Brain Injury 5, 15, 34

TCP Transmission Control Protocol 41

tDCS transcranial Direct Current Stimulation 15

TMS Transcranial Magnetic Stimulation 15

UDP User Datagram Protocol 38, 41, 44, 47, 48

UI User Interface 44, 49, 51, 98, 99, 113

UML Unified Modeling Language 50, 52, 66, 113, 114

UV Ultraviolet radiation 47

UWB University of West Bohemia 38, 97

VEP Visual Evoked Potential 30

VR Virtual Reality 34, 35, 38, 45–47, 51, 52, 93, 98, 99, 113

List of Figures

List of Tables

List of Listings