PÁZMÁNY
Pázmány Péter Catholic University
Faculty of Information Technology and Bionics

MSc Thesis

Development of a Hardware-Software Solution for Real-Time Sleep Spindle Detection in Rodents

HÁZI, Veronika
Info-Bionics Engineering MSc

Supervisor: Dr. Kiss Tamás
Faculty Mentor: Dr. Fekete Zoltán

2024

Pázmány Péter Catholic University
Faculty of Information Technology and Bionics

1083 Budapest, Práter utca 50/a www.itk.ppke.hu

Thesis Proposal Form

Ikt.: I/302/2024

Name: **Házi Veronika Panna**	Neptun ID: **J06T6N**
Study program: **Info-Bionics Engineering MSc (IMNM-AIB)**	

Supervisor

Name: **Dr Kiss Tamás**
Academic Degree, Position: **Senior Research Fellow**
Institution, Company: **Wigner Research Centre for Physics**

Faculty mentor

Name: **Dr. Fekete Zoltán**

Thesis

Title: Development of a Hardware-Software Solution for Real-Time Sleep Spindle Detection in Rodents

Summary of the thesis:
In contemporary neuroscience, sleep spindles, characterized by their distinctive electrical oscillation pattern, has garnered considerable attention. Their crucial role in the consolidation of episodic memories, alongside their contribution to memory integration and reorganization, has been well-established through prior research. Consequently, the detection and analysis of sleep spindles within EEG time series not only serve as a means to discover memory processes but also offer valuable insights into neurocognitive mechanisms.

Amidst the pursuit of further understanding and modifying sleep-related memory consolidation and integration processes, the Department of Computational Sciences of HUN-REN Wigner RCP has embarked on pioneering research in the realm of transcranial focused ultrasound stimulation. Within this innovative paradigm, a novel endeavor is underway to develop a closed-loop stimulation system. The primary objective of this system is to augment memory function, leveraging the potential of the non-invasive transcranial ultrasound stimulation to enhance the occurrence and efficacy of sleep spindles. Such a multifaceted project necessitates the integration of an online algorithm, implemented in hardware capable of real-time detection and detection-triggered output generation.

A review of offline and online methodologies has been started, with an emphasis on comparing efficacy of a number of detection algorhitms across various metrics and resource demands. The prospective result of this literature review is the proposal of an online detection method poised for implementation. The forthcoming phase brings the translation of this theoretical framework into practical application, wherein a suitable hardware platform will be identified and harnessed. Subsequent experimentation, to be conducted either utilizing animal models with surgically implanted electrodes or simulated data in their absence, will serve to validate the efficacy and reliability of the proposed system.

Detailed task description:
Conducting thorough literature review surrounding online spindle detection algorithms to gain insights and inform decision-making.

Evaluation of the capabilities of available hardware platforms, with a view toward proposing optimal applications or considering alternative hardware solutions.

Selection of a programming language - be it LabView, Python, or Matlab - to implement the chosen algorithm within the hardware framework.

Testing of the implemented algorithm's reliability and performance metrics, ensuring robustness across varied scenarios.

Execution of real-world trials within an experimental environment, thus validating the efficacy and practical viability of the developed system.

Date: Budapest, 27 March 2024

Hereby I apply for the approval of the topic of my thesis.

Signed in the Space system
27 March 2024 21:50:00
Házi Veronika Panna
Student

Hereby I undertake to supervise the thesis work of the student.

Signed in the Space system
29 March 2024 17:59:41
Dr Kiss Tamás
Supervisor

Signed in the Space system
29 March 2024 17:59:41
Dr. Fekete Zoltán
Faculty mentor

The topic of the thesis has been approved by the Faculty of Information Technology and Bionics.

Signed in the Space system
07 April 2024 23:41:17
Dr. Cserey György Gábor
Dean

Declaration of Authenticity

I, the undersigned Veronika Házi, student of the Faculty of Information Technology and Bionics at the Pázmány Péter Catholic University, hereby declare that I have prepared the present thesis myself without any unauthorized help, and that I have only used the sources specified in the thesis. I have clearly marked all parts that I have literally taken from other sources or have rewritten or translated while keeping the meaning of the original text, also indicating the source thereof. I have not submitted this thesis to any other study programs.

Házi Veronika

Veronika Házi

Abstract

Sleep spindles are bursts of electrical oscillatory activity in the brain of sleeping mammals exhibiting a waxing-waning envelope of 0.5–1.5 seconds duration, and a characteristic frequency in the 11–16 Hz range. They play a crucial role in memory consolidation, and enhancing this process through non-invasive brain stimulation, such as transcranial focused ultrasound stimulation, may improve memory and learning. Achieving such improvements requires a robust and precise real-time detection system, both in terms of software and hardware. However, most of the existing detection systems are specialized for human spindle activity, limiting their application in animal studies that are essential for in-depth research. Additionally, stimulation capabilities are often absent from these frameworks. In this thesis, I aim to address these limitations by developing a complex software-hardware system for real-time sleep spindle detection in rodents, designed to be interfaced with an ultrasound transducer to trigger neural stimulation.

I began my research by testing various sleep spindle detection algorithms offline. After reviewing the literature, I developed a simple detector and implemented four additional promising detectors in Matlab. I conducted parameter optimization and evaluated their performances using two expert labelled datasets as reference. The analysis included measures classically used in the literature to characterize sleep spindle oscillations, such as the average frequency spectra, spindle duration, inter-spindle intervals, and by-sample comparisons, considering both between- and within-subject variations. The results of three detectors demonstrated an efficiency comparable to expert-to-expert performance. Based on its stability, simplicity and accuracy, the detector I developed based on previous examples I studied was selected for further evaluation in real-time applications.

For online detection of sleep spindle oscillations, I designed and developed a hardware system using a Raspberry Pi 4 and an ADS1115 analog-to-digital converter (ADC). The setup was assembled in a breadboard using general-purpose input/output (GPIO) cables. Python was used to control the GPIO ports and access the ADC. The hardware components were tested one-by-one before real-time application. Following successful signal

acquisition, synthetic spindles were generated embedded within previously recorded electroencephalographic (EEG) signals and fed to an oscilloscope as a digital signal. The oscilloscope acted as the analog source for the tests, utilizing its arbitrary function generator feature. My detector was adapted for online operation in Python, and I tested it with the synthetic spindles. Once parameters of the algorithm were set to allow for reliable detection of artificial spindles, real EEG data was used to confirm the result. After fine-tuning the parameters, the online detection performance matched the results achieved in offline mode.

One GPIO pin was chosen to carry the control signal upon sleep spindle detections to drive the ultrasound transducer. The transducer is incorporated into the setup to deliver transcranial focused ultrasound stimulation when it is triggered. In order to unleash the full potential of the stimulation, experiments should be conducted during the natural sleeping period of the animals. This requirement necessitated the development of a device capable of securely holding the transducer on the head of the rodents. To address this, a plastic headgear was designed and 3D-printed. The stability of the structure was tested on a 3D printed rat skull by consulting with an expert of this field.

To conclude, the developed hardware-software solution with the mechanical components enables the exploration of a research paradigm that is at the forefront of current neuroscience.

Acknowledgements

This thesis was developed as part of a research project conducted at the HUN-REN Wigner Research Centre for Physics on transcranial focused ultrasound stimulation. Its completion would not have been possible without the invaluable support of the research group, who guided me through every step of the process. I am deeply grateful to my external supervisor, Dr. Tamás Kiss, for his exceptional guidance. His recommendations were indispensable, and he also contributed his own data and algorithm to this study. I would also like to thank Dr. Zoltán Fekete, who served as my faculty mentor and provided crucial assistance during administrative processes. My heartfelt thanks go to Márton Bese Naszlady for sharing his expertise in 3D printing and to Eszter Sipos for her deep insights into designing a surgery-compatible system. I am also profoundly grateful to my family and friends for their unwavering support and their valuable recommendations, which gradually helped to improve this study. Finally, I extend my deepest appreciation to Dr. Zoltán Somogyvári, who introduced me to this research area and allowed me to be part of such an exciting and pioneering project.

Contents

Thesis Proposal Form **3**

Thesis Authenticity Statement **5**

Abstract **6**

Acknowledgements **8**

1 Introduction **11**

2 Literature Review **14**

2.1 Sleep Spindles 14

2.2 Detection Algorithms 16

3 Design and Development **19**

3.1 Development of My Detector 19

3.2 Offline Evaluation of Sleep Spindle Detectors 20

3.2.1 Data 20

3.2.2 Evaluation Metrics 21

3.2.3 Parameter Optimization 22

3.2.4 Offline Performances 24

3.3 Hardware design 30

3.3.1 Components and connectivity 30

3.3.2 Testing 32

3.4 Software design 34

3.4.1 Software Set Up for Analog Signal Processing 34

3.4.2 Parallel Programming 36

3.4.3 Online Sleep Spindle Detector 36

3.4.4 Real-time Signal Visualization 38

3.4.5 Continuous Signal Storage 38

3.5 Headgear design 39

4 Results 43

4.1 Online Evaluation of the Sleep Spindle Detector 43

4.1.1 Data 43

4.1.2 Synthetic Signal 45

4.1.3 Real EEG Signal 49

5 Discussion 56

5.1 Limitations and Future Development 56

5.1.1 Software 56

5.1.2 Hardware 57

5.1.3 Headgear 58

5.1.4 Experimental Environment 58

6 Summary 59

6.1 Future Plans 59

Bibliography 64

A Appendix 65

Chapter 1

Introduction

Today, brain stimulation is no longer just a concept from science fiction movies; for many neurologists, it has become part of everyday practice, particularly with electrical, magnetic and infrared stimulation techniques. However, currently available safe methods to non-invasively modulate neural activity all have some caveats. While electrical or magnetic stimulation has very precise temporal resolution, these methods cannot target neural regions at a fine scale [1]. Furthermore, while infrared neural stimulation (INS) is capable of high-resolution spatial stimulation, it cannot reach neural structures deeper than the cortex [2]. The potential of transcranial focused ultrasound stimulation (tFUS) is clear, with its non-invasive nature, deep tissue penetration and precise focus. However, despite experimentation with ultrasound stimulation dating back to the late 1920's [3], many aspects of its effects remain unclear to this day [4]. Further experiments are definitely needed to explore this promising area of neuroscience.

In contrast to tFUS, sleep spindle oscillations are one of the most thoroughly researched phenomena of mammalian neuroscience. They are characteristic EEG events during non-rapid eye movement (NREM) sleep. Their role has been linked to memory consolidation [5], learning, cognitive functions, and neural plasticity, attracting a significant focus of research in both sleep science and neurostimulation [6]. In addition to their functional significance, their distinctive features are conserved across species, making them a key subject for translation biomarkers used in preclinical and potentially clinical experiments.

Scientists had already begun to explore the impact of tFUS on neural activities, during rapid eye movement (REM) and NREM [7] sleep. After in-phase stimulation during slow oscillations, changes occurred in spindle activity: an increase of coupling of spindle-ripple in NREM sleep was observed. This provides a hopeful outlook for future tFUS experiments aiming to influence sleep spindles and concurrently memory and learning processes.

In my long-term research, I plan to follow this direction and create a system that integrates these two fields. The envisioned framework will encompass EEG data acquisition, real-time sleep spindle detection, and a tFUS module that activates upon detection. This closed-loop system would be fitted to rodents, to be utilized during the preclinical phase of experiments. Figure 1.1 demonstrates the workflow of the closed-loop system. The process begins with recording the animal's EEG signal and culminates in the stimulation module. Such device could be invaluable in further tFUS and sleep research.

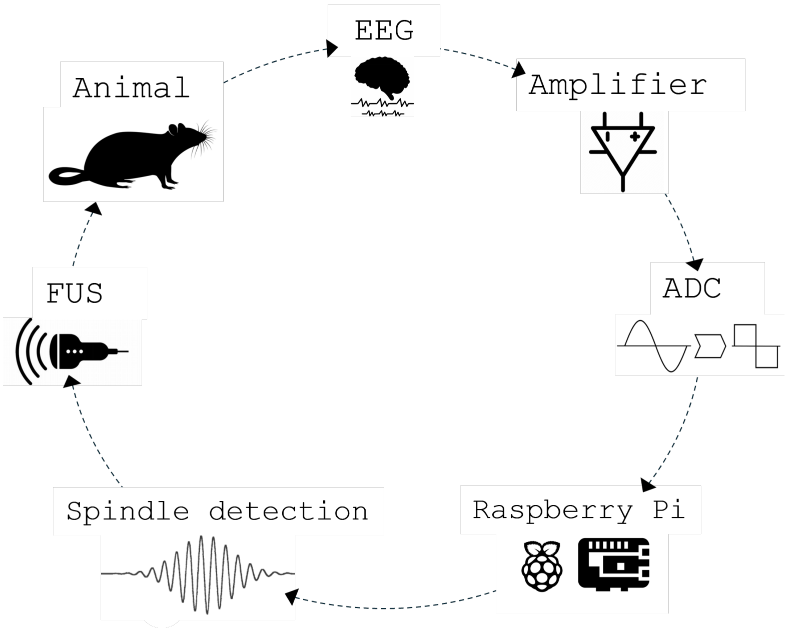


Figure 1.1: The workflow of the closed-loop system.

In this thesis, my work toward creating this closed-loop system by developing a real-time sleep spindle detection hardware-software solution is described. In Chapter 2, I first provide a short description of the background of sleep spindles. Their functions, cellular origin and most importantly, their distinctive features are detailed. This is followed by a brief introduction to the implemented offline detectors and the reasoning behind their choice. In Chapter 3 the development process is presented. It includes the offline evaluation of the algorithms, and how the conclusion was drawn for the further application of the detector developed by me. The parameter optimization process and various evaluation metrics are both explained there. Afterwards, the focus shifts to the hardware. The components and how they are connected are described in detail. Next, the real-time software is unveiled. I emphasize the solutions that are unique in the online system and the ones that were challenging to find. At the end of the chapter, a 3D-printed headgear design for rodents is presented. Incorporating this device into the system establishes a seamless connection between measurement and stimulation. To obtain the presented results, the analog signal was simulated by the oscilloscope, helping us to validate the developed framework as it is discussed in Chapter 4. The accuracy of the device is confirmed with synthetic and with real spindles. I conclude this thesis in Chapter 6 with a brief summary and an outlook to future applications and research.

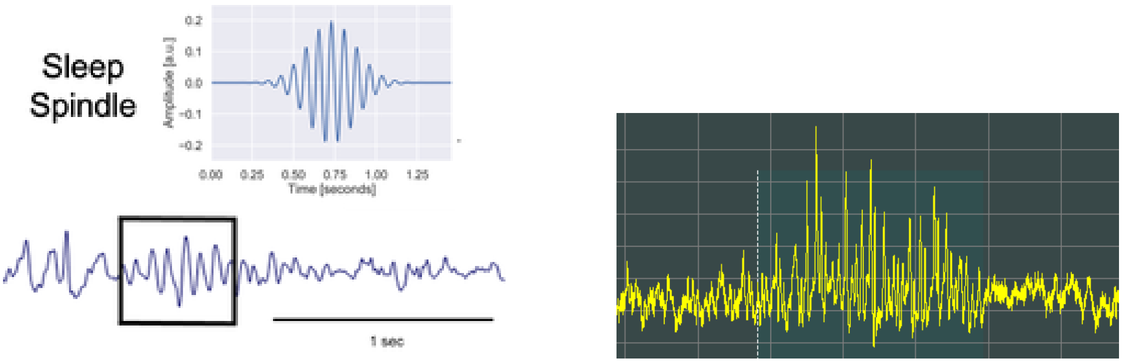
Chapter 2

Literature Review

2.1. Sleep Spindles

In neuroscience sleep spindles have been a constant topic of research for decades [6]. Their distinct appearance made it possible to identify them almost at the same time as electroencephalographic (EEG) recording was discovered. Since then, a lot of progress has been made. Now spindles are known as signs of memory consolidation and are frequently used in conversations about learning or cognition [5]. In the following section their dominant features, their cellular origin and their believed functions are all introduced briefly.

Sleep spindles are hallmark events on EEG recordings. The name speaks for itself, as they appear during sleep, specifically non-rapid eye movement (NREM) sleep, and their changing amplitude makes them resemble spindles. Figure 2.1 demonstrates the characteristic features with a schematic depiction (2.1a) and a real example (2.1b).



(a) Schematic spindles from Schmeichel Lab [8].

(b) Real spindle activity from one of the dataset I studied.

Figure 2.1: The signature appearance of sleep spindles.

Their unique features often categorize them into two groups: slow and fast spindles [6]. Slow ones tend to have higher amplitude and smaller frequency, while they preserve the typical waxing and waning pattern. On the other hand, fast spindles occupy a higher frequency range and have lower amplitude, mostly keeping an only waning shape. In general, it can be said that the spindle-like shape, the 0.5-3 s duration and the 9-15 Hz frequency range are characteristic features they carry. The specific numbers cited in various articles may differ slightly, but the overall order of magnitude has remained consistent since spindles were first described [9]. Most detection methods are designed to identify these attributes in EEG recordings. It is important to note that many detectors are specialized for human EEG. While sleep spindles are conserved across species, fine-tuning the parameters related to these features is necessary to accommodate data from animals.

The cellular origin of these events lies in the intrathalamic network of nucleus reticularis thalami (nRt) cells and thalamocortical (TC) cells, together referred to as the nRt-TC loop [10]. Specific genes like CaV3.3 and SK2 are responsible for expressing different channels, which act together to create the rhythmic firing patterns [6]. Modification of such genes was successfully used to selectively alter sleep spindles. The temporal organization of sleep spindles is regulated by cortical slow oscillations through cortico-thalamic synapses and by hippocampal ripples. Detailed studies show that ripples align with specific phases of spindles, creating "spindle-ripple events". This coalescence of rhythms is thought to play a role in transferring information for long-term memory storage [5]. That is why the finding by Dong *et al.* [7] claiming to influence this coupling with transcranial focused ultrasound stimulation is so powerful. Predominantly, the production of slow spindles happens above the frontal cortex, while the fast spindles can be detected above the parietal and central sites [6]. Additionally, they can be found in the parahippocampal gyrus, hippocampus, and, to a lesser extent, in the entorhinal cortex and amygdala. This information is key for defining the stimulation sites and the target cells for non-invasive brain stimulation intended to influence the spindle activity.

Lastly, what truly makes these oscillations interesting is their potential function. As I have already mentioned, studies indicate that sleep spindles may be contributing to neural plasticity, learning and memory consolidation [6]. The bursting activity of cells producing these events are generally believed to be an effective way of triggering synaptic plasticity. Experimental data confirmed that the generated Ca^{2+} bursts could induce both short-term potentiation and long-term potentiation in cortical neurons, and hence contribute to neural plasticity. Moreover, lowering arousal threshold therefore maintaining NREM sleep under disturbing environment is also one function of spindles that is supported by evidence. For example, the overexpression of SK2 channels led to prolonged spindle activity, and consequently, in a higher arousal threshold in mice. One of the most frequently referenced function involves the memory. Spindles have been associated with improved recall of both declarative [11] and procedural memories [12], and their activity seems to correlate with learning performance [13]. However, it remains unclear whether spindles directly trigger memory consolidation or simply enhance sleep quality, and further research is needed to fully understand their role in these processes. The closed-loop system we aim to develop has the potential to significantly advance this research.

2.2. Detection Algorithms

Sleep spindle detection can be approached in two ways: one requires the continuous presence of an expert for accurate monitoring, while the other relies on detection algorithms, allowing the computer to perform the task. While expert analysis remains the gold standard, it is highly time-consuming (and subjective), which is why many scientists are actively working to address this challenge. Therefore, I encountered a wide range of algorithms during my literature research, leaving me spoiled for choice. During the selection process, I focused on choosing detectors that demonstrated superiority in specific aspects. Additionally, I considered various working mechanisms to identify those most effective on our datasets. For potential future real-time applications, I included both simple and complex models. This approach ensures flexibility in hardware selection, allowing for either simple or complex setups without being constrained by the choice of software.

The pattern followed by most of these algorithms are easy to recognize. Their pipeline typically consists of filtering in the characteristic frequency band of spindles, fitting an envelope and applying a threshold for event detection. Finally, the duration of the identified spindles is verified to ensure it falls within the expected range. Building on this foundation, I developed my own detector (which I will refer to as My Detector), which will be detailed in Section 3.1. The skeleton of most of the spindle detecting algorithms, which also inspired the creation of My Detector, is shown in Figure 2.2.

The first algorithm from the literature was selected for two main reasons. First, it employs wavelet transformation, a method representative of many other approaches. Second, it had previously demonstrated superior performance compared to four other

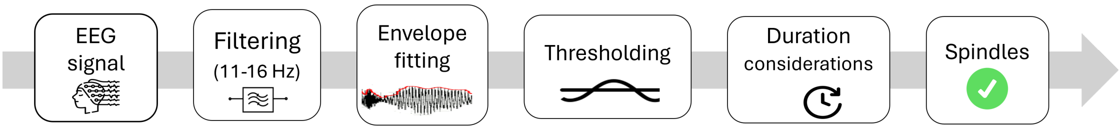


Figure 2.2: The pipeline of many sleep spindle detecting algorithms, including the one developed by me.

detectors. Originally introduced by Wamsley *et al.* in 2012 [14], its effectiveness was further evaluated by Warby *et al.* in 2014 [15]. The general idea behind the algorithm is to replace the filtering with a continuous wavelet transformation using a complex Morlet wavelet with center frequency of 13.55 Hz. Then an envelope is created by the moving average (MA) and spindles are detected over a threshold calculated from the mean of the MA signal multiplied by a constant. The MATLAB code was provided in the supplementary information of Warby *et al.*'s work [15]. I slightly modified the code to better fit my large and noisy data. First, I added a bandpass filter between 0.5 and 35 Hz to exclude noisy segments from the analysis. To avoid other noise artifacts, I replaced outlier voltages with the average of the signal.

The second detector originated from my supervisor's previous work, and it had already been applied to part of our testing data. Previously working well under similar settings, Pálfi *et al.*'s algorithm [16] got a place between the detectors. Wavelet transformation is also a key component here, but after performing the decomposition, the regions of maximal power within the spindle frequency band were identified on the time-frequency plane. Then a lower and an upper threshold is used to identify the spindles. The algorithm also introduces methods to concatenate short spindle sections that are close to each other. The last step considers the duration of the detected spindles, as many of the algorithms do. The original code was given to me by my supervisor Dr. Tamás Kiss and I only implemented its parallelization. The same approach used in My Detector was followed: the data was split into separate parts, and the analysis was run in parallel on these segments. One major advantage of this detector is that it is specifically designed for rodent sleep data, whereas all the other detectors were tested on human data.

The series of conventional detectors was concluded by the A7 detector [17]. In 2019 this algorithm surpassed its peers by using the sigma frequency band (12-16 Hz) of the EEG and some of its features as a more sophisticated detection approach. Beside the absolute sigma power, the relative sigma power, and the correlation/covariance of the sigma band-passed signal to the original EEG signal are utilized. The parameters are calculated in a 0.3 s long time window with 0.1 s long steps. This method could be compatible with online detections. The source code of the detector was uploaded on GitHub [18]. The algorithm was implemented in Python and no changes were added beside parameter optimization.

Last, but not least, as machine learning models have started to rise, the spindle detection field was reached as well. For the evaluation a neural network (NN), called SUMO was selected. The SUMO is a U-Net-type deep neural network model, which had already outperformed the A7 detector [19]. The network was trained on EEG data from 180 human participants, sourced from the Massive Online Data Annotation (MODA) project. The trained model takes two-channel EEG data as input and outputs a raw vector of probabilities, indicating whether each point is likely to represent a spindle. This is later smoothed with a moving average to get the final output. The trained model was available in the same GitHub repository as the A7 detector [18]. Because of the lack of data for training, the model trained on human EEG was used without modifications.

To summarize, five sleep spindle detection algorithms were implemented. My Detector, along with the algorithms by Wamsley *et al.* and Pálfi *et al.*, was used in MATLAB, while the A7 detector and the Sumo NN were utilized in a Python environment. The owners of the algorithms all provided their source code via GitHub or as supplementary information. This made the adaptation a quick process and only slight changes were added, like including parallelization for a faster analysis on large data. Parameters were also identified and tuned later, as detailed in Section 3.2.3.

Chapter 3

Design and Development

In this chapter, the main steps of the development process are detailed. First, the development of my algorithm is described and the offline performances of the sleep spindle detectors are evaluated. Based on the results, one detector is chosen for online applications, and it is implemented in a form to fit the constraints introduced by the real-time environment. A hardware that is capable for analog signal processing is assembled, and its components are tested one by one. The basis code that includes data acquisition, visualization and saving is also described. In addition, a 3D printed headgear design is presented, which can hold the ultrasound transducer on the animal's head during *in vivo* experiments. By the end of the chapter, every element of the real-time sleep spindle detection system has been described, and the solution is ready to demonstrate its performance in a test environment.

3.1. Development of My Detector

As previously mentioned, based on the typical pipeline of detection algorithms found in the literature, I developed my own approach following the steps shown in Figure 2.2. For filtering, My Detector employs a 4th-order Butterworth bandpass filter with a default frequency range of 12–16 Hz. Envelope detection is performed using MATLAB's built-in envelope function, which utilizes the Hilbert transform. Detections are recorded when the envelope exceeds a predefined, constant threshold and the duration falls between 0.2 and 6 seconds. This deviates slightly from the conventionally used 0.5–3 s range, however, an examination of the available expert-labeled datasets revealed that this interval was the best fit for the identified events.

The algorithm was implemented using parallel computing to accelerate its execution.

The input data is divided into six chunks, with any leftover data points discarded. Filtering and envelope fitting are performed in parallel on these segments using MATLAB's `parfor` module. Finally, the results are concatenated into a single filtered array and a single envelope array.

Figure 3.1 illustrates the working mechanism of the detector along with its specific parameters. The parameters with the most significant impact on the final output are highlighted in red. The first one defines the minimum duration of the peaks for the envelope function, while the second is the constant threshold applied to the entire signal, which must be exceeded for detections to occur. The exact values of these will be later optimized in Section 3.2.3.

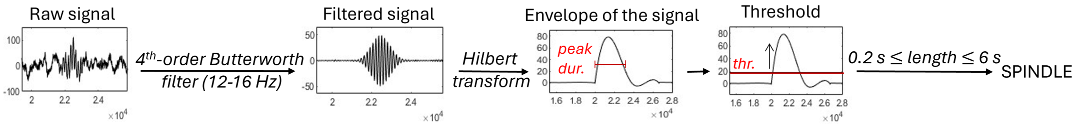


Figure 3.1: A detailed workflow of My Detector, including its specific elements and parameters.

3.2. Offline Evaluation of Sleep Spindle Detectors

In the previous chapter, some of the available sleep spindle detecting algorithms were outlined. Many of them claiming to surpass the performance of the others. In order to verify this ourselves the offline evaluation of them is necessary. Moreover, the majority of them were tested on human data and our goal is to design an accurate spindle detector for rodents. On top of that, optimal parameters could always differ between datasets. Overall, offline evaluation could provide valuable insights across multiple aspects. Before presenting the results, the utilized data sets, the evaluation metrics and the process of parameter optimization are described.

3.2.1 Data

The detection algorithms were evaluated on cortical EEG recordings of mice and rats. The data came from two different sources, both from previous research conducted by my supervisor. The first dataset contained 70 recordings from 14 mice. For the first four animals, five of their recordings were scored by an expert, but only three of them contained spindles. This leaves us with 12 annotated EEG signals. The three different measurements taken from the same animals allow us to do within-subject performance evaluation. Further information about this data acquisition can be found in Pálfi *et al.*'s article [16]. The second dataset only involved recordings from three rats, with one recording per rat. However, they were all scored by 2 experts, creating a consensus, which I used as gold standard during certain parts of the evaluation process.

3.2.2 Evaluation Metrics

To compare the algorithms, various evaluation metrics were employed. Following the methodology outlined in Warby *et al.* [15], I implemented a sample-based analysis that calculates the F1-score, precision, and recall of the detections relative to the manually labeled data. Based on true positive (TP), false positive (FP) and false negative (FN) hits, these metrics can be calculated with the following equations:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.1}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.2}$$

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.3}$$

In simple terms, a high precision score indicates that most of the identified events are true spindles, while a high recall means that the detector successfully identifies most of the expert-labeled spindles. The F1-score combines these two metrics to provide a single value that represents both precision and recall. Ideally, a perfect result would approach 1, while a result less than 0.5 would indicate lower accuracy. This can be overwritten by analyzing the gold standard dataset and comparing the performances of the two experts to establish an upper limit for the achievable F1 scores. Table 3.1 presents the F1 scores between experts on the three measurements from the rat dataset. The key takeaway from this analysis is that if the experts' agreement reaches only a mean F1 score of 0.5971, achieving a similar score with automated methods can be considered sufficient.

	Rat - 3662	Rat - 3664	Rat - 3668	Mean
F1 score between experts	0.5581	0.6751	0.5580	0.5971

Table 3.1: The F1 scores between the experts in the gold standard dataset indicate the upper limit of performance for automated methods.

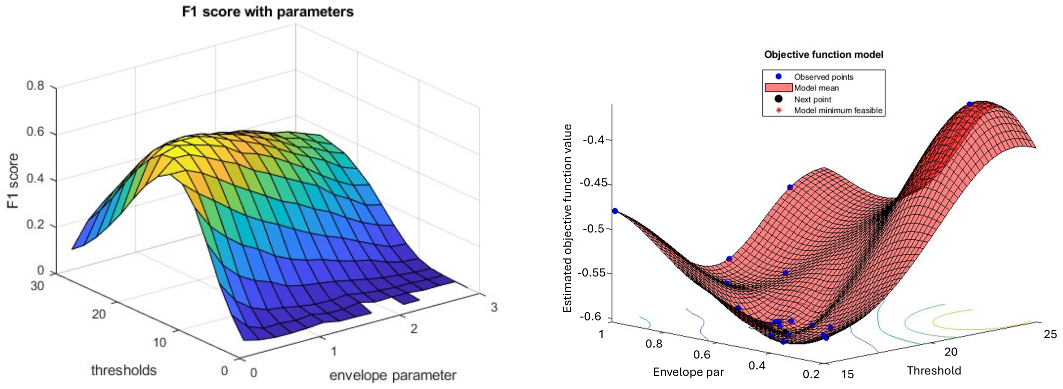
This metric formed the basis of the evaluation process and the subsequently presented parameter optimization. Additionally, further analysis took place to confirm the results. This included the extraction of the average frequency spectrum for all signals, along with the inter spindle interval distribution and length distribution. The mean squared errors between the distributions of the experts and the algorithms were calculated to quantify these results.

3.2.3 Parameter Optimization

Inherently, even the simplest signal processing pipelines require the adjustment of certain parameters. To fulfill the full potential of the algorithms, parameter optimization was performed. Because the mice dataset contained more data, it was used for the optimization task.

Initially, I identified the possible parameters by examining both the original articles and the source codes. For My Detector, I selected two parameters that were believed to have the greatest impact on the output. One parameter determined the minimum duration of the envelope's peaks, while the other set the threshold the envelope had to exceed for event detection. In the algorithm of Wamsley *et al.*, four parameters were identified as potentially influential, including the frequencies and scale for the wavelet transformation, as well as a scaling factor for the amplitude. In the A7 detector's architecture, I identified another four parameters, all of which are thresholds for features in the sigma frequency band. I concluded that for the Sumo NN, reasonable parameter choices are only available during training. Since I did not intend to retrain the network, no changes were made to the parameters in this regard. Pálfi *et al.*'s algorithm allowed for a total of six parameters to be adjusted. However, optimizing all of them would have been computationally overly expensive due to the exceptionally long run time. Therefore, the original values were retained. Overall, the parameters of the three detectors were optimized.

Subsequently, the reasonable ranges for parameter search were manually adjusted through experimentation. My initial choice for the optimization method was Grid search optimization. This technique was quite slow, as all the potential variation of the parameters are examined. However, it was useful to confirm the selected intervals that could contain the optimal values. Additionally, Bayesian optimization was implemented for a more robust, faster and automated solution. Figure 3.2 presents examples of the results for both methods side by side. The 3D plots represent the optimization of My Detector, which only had two parameters to adjust, allowing for a 3D visualization of the achieved performances. By comparing the axes between the two types of analyses, differences in the parameter ranges can be observed, which were narrowed down for the Bayesian algorithm. It is also important to note that the Bayesian optimization solves a minimization problem, while Grid search optimization seeks to maximize the performance.



(a) Grid search optimization

(b) Bayesian optimization

Figure 3.2: Parameter optimization.

Ultimately, the optimal parameters of each algorithm for the different EEG signals were provided by the Bayesian algorithm. The parameter values were collected in a CSV file with their corresponding F1 scores. Using this table, the mean values across the same animals (mean across the 3 measurements, see Table A.1) were calculated and later applied during the evaluation processes on the same dataset. The final parameters, which are ready for application to unknown animals, were the mean across the 12 signals, and they are summarized in the 3.2 Table.

	Param1		Param2		Param3		Param4	
	Name	Value	Name	Value	Name	Value	Name	Value
My Detector	Envelope duration	0.589	Threshold	18.572	-	-	-	-
Wamsley *et al.*	Center frequency	13.349	Bandwidth of the wavelet	0.451	Scale of the wavelet	3.145	Scale of the amplitude	1.699
A7 detector	Threshold for absolute sigma power	1.848	Threshold for relative sigma power	0.445	Threshold for sigma covariance	2.369	Threshold for sigma correlation	0.603

Table 3.2: The final parameter values are the averages of the optimal values calculated across the 12 signals.

These values provide an initial estimate for novel signals. Nevertheless, parameter optimization of data gathered from new animals or in different measurement environments is always beneficial and is essential for the best results.

3.2.4 Offline Performances

Taking advantage of the optimal parameters calculated for each animal, the first part of the evaluation was carried out using the mice dataset with the corresponding parameters. For the exact values, see Table A.1 in the Appendix.

As a starting point, detections by the expert were collected in a 2D-array, with the beginning of the spindles in the first dimension and the durations in the second. This array is later referred to as *Es* and is compared to a similar one, which is formed from the spindles found by the detectors, called *Ds*. All analysis was done using these two objects.

It should be emphasized that during the evaluation, the potential of the algorithms for the online application had to be prioritized. Hence, time consumption was a critical point of the process and that often made us sacrifice detectors if their accuracy showed no promise either.

F1 score

The F1 score served as the foundation for evaluation. This means that the detectors were primarily assessed based on their performance in these metrics. The recalls, precisions, and F1 scores were calculated using equations 3.1, 3.2, and 3.3. The results for all detectors across the 12 signals were averaged and are presented in Table 3.3, with the highest performances highlighted in gray. Pálfi *et al.*'s algorithm, which required several minutes to process, significantly longer than the others (for reference, My Detector took approximately 10–20 seconds) yielded an F1 score that was much below the others', and therefore it was excluded from this analysis.

	F1 score	Precision	Recall
My Detector	0.553	0.502	0.623
Wamsley *et al.*	0.425	0.420	0.433
Pálfi *et al.*	-	-	-
A7 detector	0.566	0.611	0.529
Sumo NN	0.578	0.594	0.571

Table 3.3: Mean F1-score, Precision, and Recall for the different models between subjects.

The Sumo NN achieved the highest overall F1 score, with both the A7 detector and My Detector coming close to the top. In contrast, Wamsley *et al.*'s method lagged behind the others. Despite Sumo's strong F1-score, the A7 detector stood out for its high precision, while My Detector excelled in recall. These differences may become crucial in real-time applications, where favoring precision or recall could enhance outcomes based on experimental insights. Figure 3.3 visually presents these results, highlighting Wamsley *et al.*'s notably lower performance.

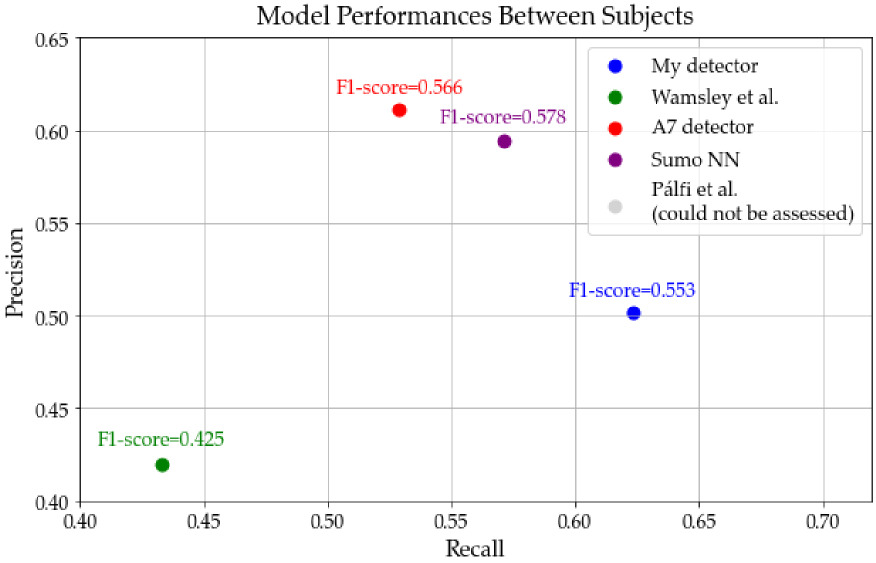


Figure 3.3: Between subject performances on the recall-precision axis with the mean F1-score displayed.

The dataset included multiple repeated measurements for each animal, which opened the door for within subject analysis. This could be a meaningful tool regarding the stability of the algorithms. Low variance of the results within a subject could indicate a stable performance and the significance of parameter optimization before the signal acquisition, while varying values can question the reliability of the detector. The algorithms were evaluated by calculating the mean performance across the three measurements within each animal, with the standard deviation representing variability in these measurements. All the performances of the algorithms are presented on Figure A.1 in the Appendix, where the proximity of the points with the same colour represents the stability of the performances. Table 3.4 contains the standard deviations. To summarize these results, high fluctuations in the Sumo NN's performance and a stable output for the A7 detector and My Detector can be observed. This suggests, that the preliminary calibration of these two detectors on the animal, could lead to accurate detections. Keep in mind that the Sumo NN had not been optimized and is currently trained on human data. With further optimization, this anomaly may be resolved, but that requires a lot of data on animals for training.

Standard deviation within subject			
	F1 score	Precision	Recall
My detector	0.0340	0.0350	0.0486
Wamsley *et al.*	0.0353	0.0702	0.0346
Pálfi *et al.*	-	-	-
A7 detector	0.0239	0.0411	0.0320
Sumo NN	0.0510	0.0312	0.0916

Table 3.4: Standard deviation results of different detectors within subject.

The by-sample analysis demonstrated strong performance for three of the five detectors. To further evaluate these findings, the annotated rat dataset was utilized. The primary goal of these calculations was to determine whether the characteristic features of automatically detected spindles aligned with those identified by expert consensus. This dataset contained much shorter signals, providing an opportunity to test the efficacy of Pálfi *et al.*'s algorithm. The parameters were derived from Table 3.2, except for the threshold for My Detector, which was increased to 50 to accommodate the amplitude of the new signals. The detection array (Ds) was extracted using the same logic as before, but applied to the new dataset. The consensus array (Es) included spindles identified by both experts, representing the shared ground truth or consensus. Ultimately, conclusions were drawn from this offline analysis and one detector was selected for real-time application.

Average Frequency Spectrum

In the average frequency spectrum of the spindles, a peak typically appears within the range of 9–16 Hz. For better visibility, this interval is highlighted in Figure 3.4. The upper subplot displays the spindle spectra as identified by the various algorithms, while the lower subplot illustrates the characteristics of the consensus spectrum.

All detectors, except the Wamsley *et al.*, along with the experts, have a distinctive peak in the expected band. It implies correct detections for most, and poor performance for the Wamsley *et al.* To better demonstrate these differences, the mean squared errors between the algorithms and the consensus were calculated based on the data of three rats. The results are collected in Table 3.5, supporting our assumption that the Wamsley *et al.* was the weakest link during this analysis.

An interesting, but not surprising result belongs to the algorithm of Pálfi *et al.*,

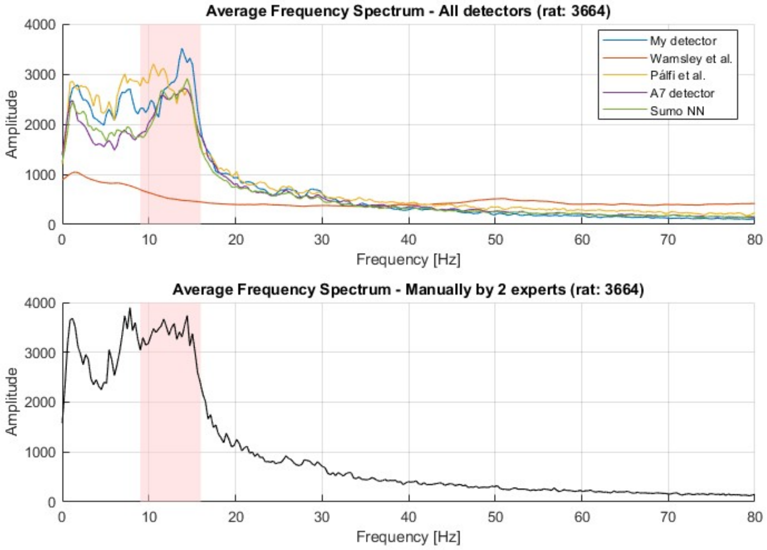


Figure 3.4: Average frequency spectrum of the detected sleep spindles by the algorithms and the experts.

	Mean squared error of the spectra		
	3662	3664	3668
My Detector	$6.79 \cdot 10^4$	$6.21 \cdot 10^4$	$6.89 \cdot 10^4$
Wamsley *et al.*	$4.94 \cdot 10^5$	$7.21 \cdot 10^5$	$6.13 \cdot 10^5$
Pálfi *et al.*	$5.57 \cdot 10^4$	$4.48 \cdot 10^4$	$3.93 \cdot 10^4$
A7 detector	$1.06 \cdot 10^5$	$1.39 \cdot 10^5$	$3.62 \cdot 10^5$
Sumo NN	$1.46 \cdot 10^5$	$1.44 \cdot 10^5$	$3.62 \cdot 10^5$

Table 3.5: Mean squared errors of the average frequency spectra of the algorithms using the consensus as reference.

which produced the smallest errors. This was also expected, because that algorithm was developed and optimized on this dataset. My Detector had no such advantage, but still exhibited a solid performance.

Inter Spindle Interval

The second key feature analyzed was the inter-spindle intervals (ISI). An increased density of spindles is believed to indicate learning and has been correlated with improved performance in motor tasks [6]. This highlights the importance of accurately detecting spindles with correct time intervals between them. However, given the wide variability of ISI, a direct calculation may be less meaningful. Instead, comparing the detected

ISI to those identified by experts provides a more insightful measure. To evaluate this, the mean squared errors (MSE) between the detected spindles and the expert consensus were calculated and summarized in the Table 3.6. Additionally, Figure 3.5 illustrates an example of ISI distributions, demonstrating the alignment between the detection method and expert assessments. The log function is applied to the intervals, making the x axis show values that correspond to the log-transformed intervals. This transformation helps handle the typically wide range of ISI values by compressing the scale and allows easier comparison of distributions. On the y axis, the normalized frequency of the ISI values is shown.

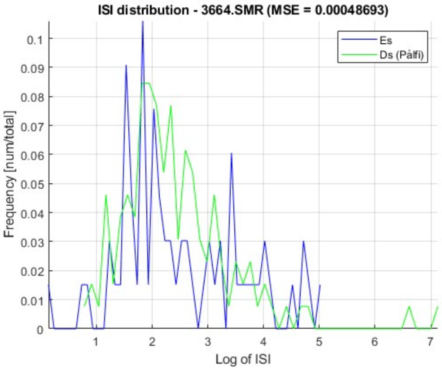


Figure 3.5: Example distributions.

	Mean squared error of the ISI		
	3662	3664	3668
My Detector	$1.01 \cdot 10^{-3}$	$4.40 \cdot 10^{-4}$	$4.15 \cdot 10^{-4}$
Wamsley *et al.*	$6.57 \cdot 10^{-4}$	$1.09 \cdot 10^{-3}$	$1.03 \cdot 10^{-3}$
Pálfi *et al.*	$6.39 \cdot 10^{-4}$	$4.87 \cdot 10^{-4}$	$5.01 \cdot 10^{-4}$
A7 detector	$6.98 \cdot 10^{-4}$	$4.88 \cdot 10^{-4}$	$5.89 \cdot 10^{-4}$
Sumo NN	$5.32 \cdot 10^{-4}$	$7.07 \cdot 10^{-4}$	$3.99 \cdot 10^{-4}$

Table 3.6: MSE of the ISI between the algorithms and the consensus.

Comparison of spindle detectors based on inter spindle intervals (ISI).

Overall, the results were highly consistent among the detectors and fell within the same range as the variability observed between expert scorers (10^{-4}). My Detector and the method by Wamsley *et al.* revealed slightly weaker performance, but without significant difference.

Spindle Duration

The duration of the spindles is a characteristic feature of the event, even though the exact range has not been established. Generally, they are expected to be between 0.5 to 3 seconds [15], but examples are often seen which do not strictly follow these margins. Consequently, instead of solely analyzing the length of the detections, comparison with the consensus could give more informative picture about accuracy. Table 3.7 shows the mean squared errors between the algorithms and the consensus, while Figure 3.6 demonstrates an example of the distributions.

The experts identified spindles with a maximum duration of 3.3 seconds, while Pálfi's

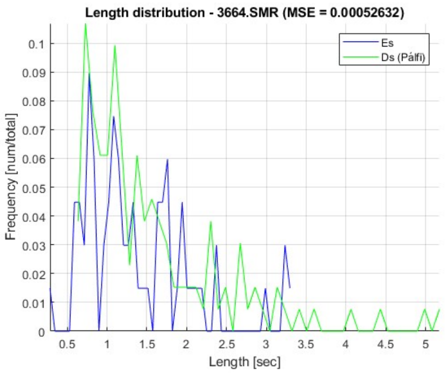


Figure 3.6: Example distributions.

	Mean squared error of the durations		
	3662	3664	3668
My Detector	$2.42 \cdot 10^{-3}$	$7.31 \cdot 10^{-4}$	$1.17 \cdot 10^{-3}$
Wamsley *et al.*	$2.40 \cdot 10^{-3}$	$9.98 \cdot 10^{-4}$	$2.12 \cdot 10^{-3}$
Pálfi *et al.*	$1.36 \cdot 10^{-3}$	$5.26 \cdot 10^{-4}$	$6.36 \cdot 10^{-4}$
A7 detector	$2.41 \cdot 10^{-3}$	$8.11 \cdot 10^{-4}$	$9.48 \cdot 10^{-4}$
Sumo NN	$2.57 \cdot 10^{-3}$	$4.89 \cdot 10^{-4}$	$1.17 \cdot 10^{-3}$

Table 3.7: MSE of the durations between the algorithms and the consensus.

Comparison of spindle detectors based on durations of spindles.

detector was capable of detecting spindles up to 5.1 seconds. For both methods, most of the spindles fell within the 0.5–1.5 second interval. The scorer agreement analysis yielded a mean squared error (MSE) of approximately $8.4 \cdot 10^{-4}$ for spindle length. Considering this, the algorithms demonstrated reasonable accuracy, with Pálfi *et al.*'s detector slightly outperforming the others, though without major distinction.

Conclusion

The primary conclusion from the evaluation is that the detector by Wamsley *et al.* struggled with both time consumption and accuracy across different datasets and did not demonstrate sufficient performance in any key area to suggest their suitability for further use. The algorithm of Pálfi *et al.* could reach similar results as the others on the second dataset, but failed on the first one containing larger signals with more noise. This implies a less reliable mechanism, which is better to avoid in the future. The other three algorithms demonstrated great potential for online usage with convincing results. The Sumo NN reached the highest F1 score of 0.58, which is almost as good as the average F1 score between experts. The A7 detector and My Detector showed stable performance and remarkable results in precision or recall. Based on these, all three of them would be a good candidate for real-time detections. However, beside performance, consistency and time consumption, hardware and software compatibility was also a crucial factor in my choice for online application. While the Sumo NN inherently has a complex structure, My Detector is built from the simplest signal processing units, that would run on any hardware. Ultimately, that benefit helped to shift the choice toward My Detector. The A7 detector fell of the radar at the moment, because of its slightly more complicated structure. However, it is not completely out of the scope. During the animal experiments, high precision might be prioritized, especially if ultrasound stimulation reveals harmful or unwanted effects when applied without spindles. In such cases, the A7 detector should be preferred to the others.

3.3. Hardware design

The hardware of the real-time sleep spindle detection system comprises two primary components: one for acquiring the EEG signal and another for processing it with the selected algorithm. In addition, a closed-loop stimulation system requires a stimulation component, which, in this case, is an ultrasound transducer. EEG measurements are taken using small, surgically implanted M1x2 stainless steel screws that penetrate the skull. Wires from these screws are soldered to a connector mounted on the head of the animal. Given the low amplitude of EEG signals, they must be amplified before entering into an analog-to-digital converter (ADC). The converter is necessary as only a digital signal can be processed according to the detection algorithms performed by a microcomputer. This microcomputer executes the selected algorithm and generates a control signal to drive the transducer. The following section presents each component, examining their advantages, suitability, and connectivity for this project.

3.3.1 Components and connectivity

The essential parts of the hardware setup include a microcomputer, an ADC, and an amplifier, as well as an oscilloscope and LED for testing purposes. First, the specific components are described in detail.

A Raspberry Pi (RPi) is a common choice for signal processing tasks, more broadly, for a range of applications where real-time data handling and embedded processing are needed. The RPi 4 model B has quad-core ARM Cortex-A72 CPU running at 1.5GHz and a 8GB of LPDDR4 RAM, allowing the running of memory intensive applications and higher performance than previous models [20]. The updated hardware behind the 40 GPIO pin-layout provides a more efficient use with external devices. Its capability to handle complex calculations quickly, connect with multiple input/output devices, manage precise timing, and its compatibility with Python made it ideal for the implementation of my real-time sleep spindle detection application.

In addition to the Raspberry Pi, an analog-to-digital converter (ADC) is required to process the analog signals. The ADS1115 16 bit ADC module is popular among RPi users. Its input voltage range can be scaled through the programmable gain amplifier (PGA), which allows to represent voltages as small as $\pm 7.8125\ \mu V$. For the exact gain and resolution parameters, see Table 3.8. The sampling rate can also be adjusted to a maximum of 860 samples/second. This rate is suitable for the 9–16 Hz spindle events, which we would like to detect. Another main advantage is the I2C interface type. Using this protocol the RPi can easily communicate with the ADC and due to the popularity, accompanying Python libraries already exist. After contemplating the low price, availability and beneficial characteristics, we decided on behalf of the ADS1115 ADC. We ended up purchasing a complete module from Hestore [21].

The magnitude of EEG signals is usually between 10 and 100 μV. Taking into account the ADC's input voltage range, the need for an amplifier during animal experiments is clear in order to have an appropriate resolution of the EEG. At the research centre a BL-096/16 Multifunctional Biological Amplifier (ELSOFT Bt., Budapest) was available. This has a maximum gain of 1000, which shifts the input range into 10 to 100 mV. It is still far from the optimum, which could fulfill the -256 mV to +256 mV range, but roughly a $5 \cdot 7.8 = 39\ \mu$V resolution can be achieved in this way. After conducting some experiments, we concluded that it would be sufficient, hence the BL-096/16 Amplifier was utilized.

Connections between these elements were built up on a breadboard using multiple male and female jumper cables. Additional components used during the testing included an LED, a 1 kΩ resistor and a Voltcraft DSO-6104F oscilloscope. The block diagram of the components is shown on Figure 3.7.

A schematic diagram was created using KiCad to illustrate the primary connections in the circuit, as shown in Figure 3.8. This diagram includes all elements, which were either used in the testing phase or are supposed to be utilized during the real experiments. The LED and oscilloscope belong to the former group, acting as substitutes for the transducer and EEG signal, respectively. The analog signal is represented by either the oscilloscope or the true EEG signal, serving as the input for the amplifier. The amplifier is a complex device with multiple input and output channels. As for now, the first non-inverted input channel, the ground and the first output channel were utilized, hence the amplifier in the KiCad model is only a simplification of the real component. The amplified signal enters the first input channel of the ADC. The ADC is again simplified and the whole module is not represented. The Raspberry Pi's 3.3 V GPIO pin (pin 1) supplies the voltage, while pin 6 provides ground (GND) for both the ADC and Raspberry Pi. I2C communication

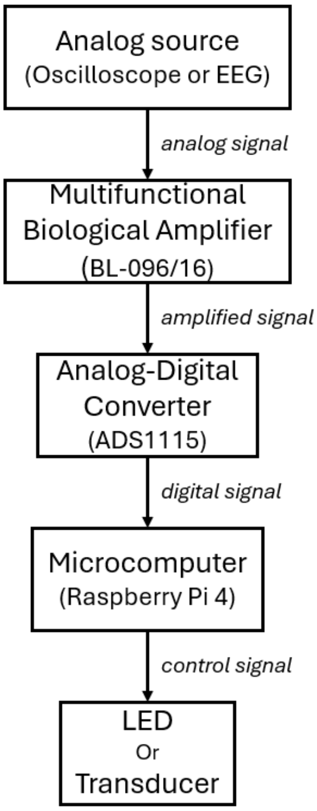


Figure 3.7: Block diagram of the hardware components.

is established between the devices via SDA and SCL pins, corresponding to GPIO2 and GPIO3 on the Raspberry Pi, respectively. Output is generated through GPIO17 (pin 11), which, during testing, drives an LED with an in-series resistor for current regulation. In actual experiments, GPIO17 will control the transducer.

3.3.2 Testing

Each component was individually tested to ensure proper functionality. First, the Raspberry Pi was set up, and a test Python script (like 'Hello World') was executed successfully. Next, the ADC was verified through communication with the Raspberry Pi. The amplifier was tested using a small signal generated by an oscilloscope, and finally, the transducer was driven directly from the Raspberry Pi. Below, a thorough breakdown of these tests can be found.

To test the ADC and its communication with the Raspberry Pi, an oscilloscope was connected to the system, generating a simple sinusoidal signal. The primary objective was to display the acquired sinusoidal waveform on the RPi for verification of receiving the analog signal. The signal was created with the help of the oscilloscope's function generator. The frequency was set to 1 Hz, the peak-to-peak voltage to 100 mV and the

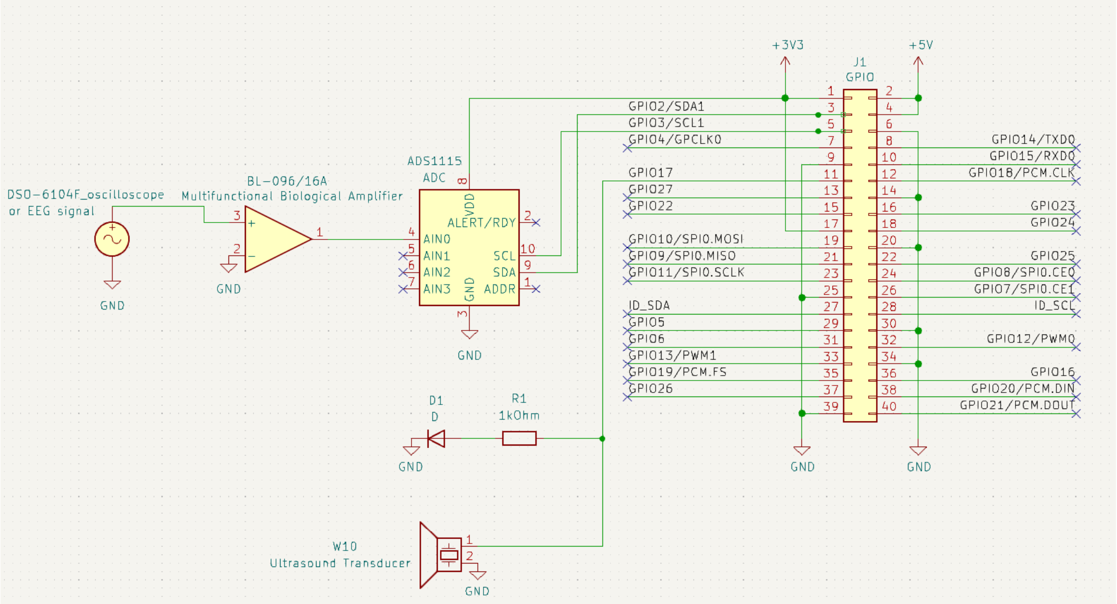


Figure 3.8: Schematic diagram of the connections of the hardware.

offset to 50 mV to avoid negative voltage values. The signal was coupled to the ADC input via an oscilloscope probe with a BNC connector and alligator clips. After setting up the connections in the program and running it, the output is shown on Figure 3.9 next to the original signal on the oscilloscope. This confirmed the correct operation of the ADC and RPi system.

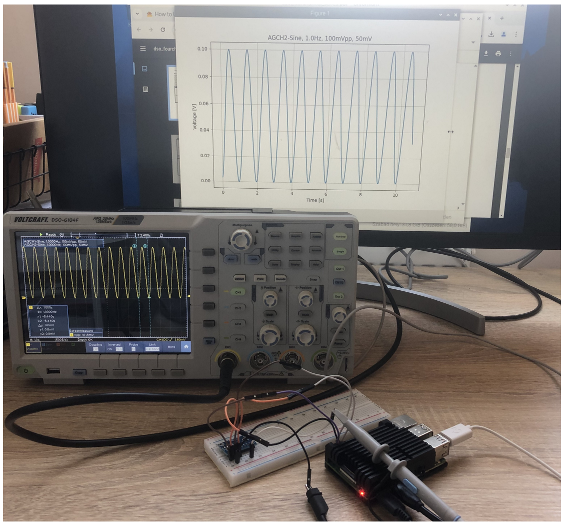


Figure 3.9: A simple sinusoidal signal is received and displayed by the Raspberry Pi.

The amplifier was tested using a similar setup, with an additional oscilloscope measuring the amplified signal instead of the Raspberry Pi. The output of the first oscilloscope was connected to the amplifier's non-inverted input and ground. This amplified signal was then fed into the second oscilloscope. The function generator of the first oscilloscope was limited to a minimum amplitude of 2 mV, resulting in a sinusoidal signal with a peak-to-peak voltage of 2 mV. On the amplifier, Channel 1 was selected in a non-inverted mode and a gain setting of 1000. The second oscilloscope recorded an output signal with a 2 V amplitude, confirming successful amplification.

The control signal generated by the RPi was first tested with an LED. The LED was switched on when the sinusoidal signal exceeded a certain threshold, and was switched off otherwise. This setup ensured the functionality of the driving GPIO pin. Next, the transducer, which was custom-built with a configuration that enables this type of control, was connected to the GPIO17 pin. The transducer drove another transducer, which was connected to an oscilloscope. The idea was to generate an electrical signal in the second transducer from the vibrations produced by the first one, allowing the oscilloscope to record this response. The transducer had a configuration to block the radiation upon getting a HIGH input, so in that case we expected a flat response on the oscilloscope. The test initially revealed an error with the electronics of the transducer, but after repairs, it was ready for use.

3.4. Software design

Following successful hardware testing, software development began. To perform real-time animal experiments, well-structured software implementation with specific functionalities is essential. The necessary functions include real-time EEG signal display for immediate data verification, data storage for future analysis, and rapid, accurate sleep spindle detection. The program's core architecture involves establishing I2C communication between the Raspberry Pi and the ADC, configuring GPIO, reading data from the ADC, and routing it through multiple processing functions to implement these key features. Alongside these functions, the reading from the ADC, the timing foundations and parallelization aspects are also discussed below.

3.4.1 Software Set Up for Analog Signal Processing

The analog signal needs to be digitized before processing, so the ADS1115 ADC is included in the circuit for this purpose. On the software side, the output of the ADC must be read to enable data processing on the Raspberry Pi. Fortunately, Python libraries are available for direct communication with the ADC from the RPi. The following section documents this setup.

Communication between the Raspberry Pi and the ADC is established using the I2C protocol, by the means of several imported modules. On the microcomputer, Python libraries like `board` and `busio` streamline the process, while the ADC interface relies on the `adafruit_ads1x15.ads1115` library. This library enables the gain and sampling rate of the ADC to be configured through the `gain` and `data_rate` parameters. Table 3.8 summarizes the available gain values (PGA). The sampling rate can be adjusted from 8 to 860 Hz, with a default of 128 Hz. For the communication tests, the gain was set to 4, providing a resolution of about 0.3 mV, suitable for the expected 100 mV peak-to-peak amplitudes. The data rate was set to 860 Hz, but the actual reading speed was set to considerably lower, averaging around 300 Hz. These numbers were later adjusted for the different measurements.

Gain Setting	**Full-Scale Range**	**LSB Size**
2/3	±6.144 V	187.5 μV
1	±4.096 V	125 μV
2	±2.048 V	62.5 μV
4	±1.024 V	31.25 μV
8	±0.512 V	15.625 μV
16	±0.256 V	7.8125 μV

Table 3.8: The gain settings, the resulting full-scale ranges and the corresponding LSB sizes for the ADS1115 ADC [21].

The input channel of the ADC is specified using the `adafruit_ads1x15.analog_in` module, allowing the Raspberry Pi to read directly from that channel via the I2C protocol without manually defining input pins on the board. For output control, however, the `RPi.GPIO` library is used with BCM numbering, setting GPIO 17 as an output. During testing, the default setting was LOW, but for transducer control, it was changed to HIGH. Upon detection, this setting was briefly toggled to the opposite state for approximately 0.5 seconds to initiate ultrasound stimulation or to activate the LED.

Timing

The appropriate timing is one of the most crucial point of any real-time system. In Python the `time` module and its `sleep()` function is a popular choice in similar projects. However, during testing this did not perform accurately for us and caused random delays in the reading from the ADC. Hence, the timing in our pipeline is implemented with the `time.monotonic()` function. A sampling rate (e.g., 1/300 second/samples) is defined at the beginning of the script, and that is always used to calculate the next time point for reading. If not enough time passed since the last sampling, a while cycle holds up the execution. With this solution the delays compared to the sampling rate were negligible. For a more time-efficient application, data visualization is managed at a separate rate. Following the same logic as before, the signal is plotted only if the predefined time (e.g., 0.3 seconds) has elapsed, reducing the demand on the CPU core.

3.4.2 Parallel Programming

As I have mentioned previously the proper timing is indispensable. The system must be able to plot the data, write it to a file and process it for spindle detections all in a specific time interval. Fortunately, these tasks can be executed parallel, which would minimize the time spent without reading. To access the RPi's four processors Python's `multiprocessing` library was utilized. Data is passed through FIFO (First-In, First-Out) queue structures, ensuring that each piece of information is processed in the precise order it was enqueued, thereby preserving the sequential integrity of EEG data points and spindle events. A total of four processes are launched, for data acquisition, plotting, saving, and analysis activities. One queue for each is sent from the acquisition function to convey the captured data points. Two other FIFO queues are generated from the analysis function to send information of the detected spindles to the storage and visualization functions. A shared stop signal coordinates the termination of all processes, enabling a synchronized and orderly shutdown. This design ensures smooth and reliable data handling across processes.

3.4.3 Online Sleep Spindle Detector

Data analysis operates as one of the parallel processes in our system. While data acquisition occurs continuously, analysis begins only after the data reaches a minimum length of 0.2 seconds, which corresponds to the expected minimum length of spindles. Once this condition is met, the signal is sent to the online detector, which is based on My Detector used for offline analysis. The basic pipeline follows the same steps as before (see Figure 2.2), but with slight modifications to better suit the real-time nature of the task.

Originally developed in MATLAB, the algorithm has been fully implemented in Python on the Raspberry Pi. One significant change involves the envelope fitting method: while offline analysis utilized the Hilbert transform, which was computationally intensive for real-time applications, the online version calculates the rectified signal and processes it through a low-pass filter to extract the amplitude envelope. Additionally, the duration criteria have been adjusted to focus on immediate stimulation upon spindle detection. Rather than detecting the entire spindle length between 0.2 and 6 seconds, the system now prioritizes detecting a spindle that is at least 0.2 seconds long, triggering a control signal immediately upon identification. Subsequent data points are still analyzed, allowing the corresponding control signal to remain active. For instance, if a spindle lasts 0.6 seconds, three distinct spindles would be detected, maintaining the correct output for that duration.

To manage the analysis within the constraints of the experiment, where longer intervals without spindle activity can occur, signal pieces are analyzed with a maximum length of 6 seconds. If there are no spindles within this timeframe, the first data point is discarded (using the `.pop()` function), and the next is read to maintain the required 6-second signal length. As the signal window is consistently processed through a band-pass filter, we have introduced a parameter to control the size of the filtering output. This allows for the trimming of the beginning and end of the output, as filtering can introduce anomalies at those points. This solution ensures that the values obtained after filtering are accurate and reliable. The envelope fitting and the thresholding follows the filtering process just as in the offline pipeline. When valid detections occur, the results are forwarded to the visualization and storage processes, while the output voltage is generated to control the LED (and later the transducer). Subsequently, the temporary array containing the current processing data is cleared and the queue from data acquisition is read again. The workflow of the online detections is shown on Figure 3.10.

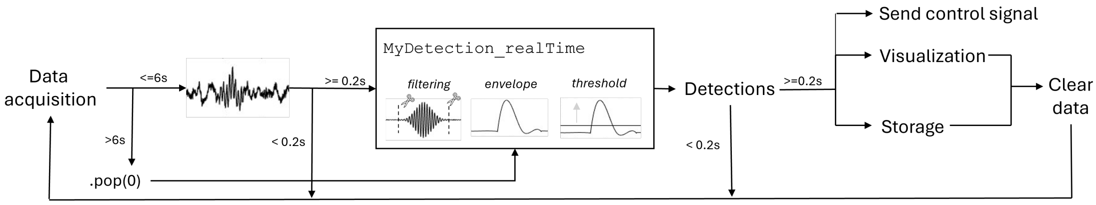


Figure 3.10: Workflow of the online detections.

The analysis function takes the three queues, the stop event, the sampling rate, the maximum length of the analyzed data and the size of the cut-off segments during filtering as input variables. The latter one with the sampling rate was inherited by the detection function complemented with the voltages. Additionally, the detection function allows setting parameters such as threshold size, frequency range, and the order of the Butterworth filter.

3.4.4 Real-time Signal Visualization

The visualization function is the second process running in parallel with data acquisition. Its purpose is not only to verify detections, but also to monitor the EEG signal, allowing for quick identification of any issues with the measuring electrodes. If anomalies are observed in the visual output, measurements can be paused and corrected immediately, preventing delayed recognition of potential issues.

The figure is created using the `ion()` function from the `matplotlib` library, enabling interactive plotting. After setting up the plot elements, the `figure.canvas.draw()` function is called to ensure the axis labels and title are displayed correctly from the start. For efficiency, the visualization is updated in intervals rather than continuously, reducing computational load. The `plot_update_interval` parameter controls the time between each update, ensuring the figure refreshes at regular intervals. This interval can be adjusted depending on the requirements of new measurements. When a display update is timely, the time and voltage data arrays are refreshed, and the window size of the plot is adjusted accordingly. For optimal memory usage, the function limits the temporary arrays for voltages and timestamps to twice the maximum analyzed data length (by default, 12 seconds), while only a 10-second window of data is shown, with older data continuously removed. The function also reads spindle events from a dedicated queue that the analysis process sends. Spindle events are marked by dashed lines at their start and end times. These lines are removed along with the background whenever the plot is refreshed, and the function then checks for any new events. After the stop event is triggered, interactive plotting is turned off, but the figure remains open until it is manually closed.

The input variables include the two queues, the stop event, the interval to update plotting, and the maximum size of data.

3.4.5 Continuous Signal Storage

The final process is the storage function. To enable future analysis, this function saves the EEG signal along with the timestamps of spindle events and stimulations. By using this saved file, the recorded data can be reconstructed, allowing for an assessment of detection accuracy. Including the stimulation data also makes it possible to analyze the timing delays between spindle onsets and stimulations, which is crucial in experimental setups where the effects of ultrasound may depend strictly on precise timing.

The output file is a CSV with three columns: Voltages [V], Time [s], and Spindle [T/F], designed for straightforward interpretation. Voltages and timestamps are logged to the file immediately as they enter into the queue, with minimal delay or effort. However, saving spindle events is more complex. These events are received from the analysis function via a dedicated queue after the relevant data segment has already been saved. To ensure these data points are not missed, a buffer temporarily stores recent entries. When a spindle event is detected, the buffer checks for timestamps within the spindle's start and end times. If matching timestamps are found, the third column is updated to mark these entries as `True` for spindle events, and the entire buffer is then written to the output file. This approach, however, may create duplicate time and voltage values in the file. Managing these duplicates is handled in post-processing, where data is sorted in ascending order, and for duplicate timestamps, only the entries marked with a `True` spindle event are retained. To save memory during extended experiments, the buffer size is limited to the predefined maximum data length. If the buffer exceeds this limit, it shifts to accommodate new data points, discarding the oldest entries.

Stimulation times are not yet included in the CSV file. Currently, spindle detection times are printed to the command window, showing the detection timestamp along with the spindle's start and end times. After the measurement, this information can be saved into a separate TXT file for documentation. When a spindle is detected, an output voltage is triggered on the GPIO immediately, so the detection timestamps approximate the stimulation times. This setup enables efficient recording and provides a close estimate of the stimulation timing without directly logging it in the CSV file.

3.5. Headgear design

As the original pipeline implies (Figure 1.1) the hardware was designed to drive a transducer upon spindle detections, which aims to stimulate certain brain areas to enhance memory consolidation. The concept is that behavioral tests are conducted during the rodents' awake periods to create episodic memories. During resting, animals enter slow wave sleep within approximately 30 minutes and start to generate sleep spindle oscillations. When spindles are detected, ultrasonic stimulation is applied to cortical areas, enhancing the generation of spindles. Subsequenty, when the animal wakes up, the same experimental task is performed and recall performance is evaluated. The results are compared to rodents who do not receive the stimulation. This means that instead of doing the stimulation during anesthesia, a headgear can be designed which connects to an implanted pedestal, so the animal can wear it during the day and their natural sleep rhythm can be assessed. In this subsection, the design of such headgear is described. The idea and the basis of the headgear originates from Lee *et al.* [22].

The designs were created in Fusion 360. To ensure accurate dimensioning and component testing, a 3D model of a rat skull was downloaded from Sketchfab [23]. Animal comfort was a top priority in the design process, particularly by minimizing the weight of the device, which is closely related to its size. A method introduced by Lee *et al.* was adopted, using an implanted pedestal to act as a connector for the headgear, allowing the animal to avoid carrying the entire setup continuously. Our headgear is based on a plastic cylinder mounted onto the skull's surface, likely secured with dental cement or glue. Onto this pedestal, an additional cylinder fits and is fastened with a screw on an extra material surface. A thin board is attached to this cylinder using a clamp, allowing height adjustment of the device. The transducer is then connected to this board through a 3D-printed holder structure featuring a ball joint. The ball joint is designed to be glued into a hole in the board, which would be drilled during the surgery upon determining the ideal placement for the transducer. This solution allows both adjustment of the transducer's height and angle resulting in precise positioning over the desired brain region. The holder also features a side opening to accommodate the transducer's cable. The last piece of the headgear is a cone put on the transducer and filled with ultrasound conductive material (like ultrasound gel or water) allowing ultrasound transmission. Figure A.2 shows the individual parts one-by-one alongside the rat skull model. The assembled headgear in Fusion is shown in Figure 3.11. The dimensions are provided for reference, without claiming to be exhaustive.

On the Fusion model, the locations of the different elements approximate the experimental setup. To facilitate further experimentation with their actual placements, the components were printed in multiple instances. The configuration assembled from the printed parts is shown in Figure 3.12. The different colors indicate the materials used during printing: the black material is brittle, lacking elasticity and prone to breaking under stress, making it suitable only for modeling the setup. In contrast, the gray material is more flexible while still offering excellent stability, making it the likely choice for use with surgically implanted rats. After the initial print, in addition to the insights gained about the materials, the dimensions were also reconsidered. The height of the cylinder was increased, and the length of the board was reduced. Moreover, the width of the walls was consistently reduced in most of the components. As of the time of writing this thesis, the redesigned parts have not yet been printed, but it is hoped that they will be suitable

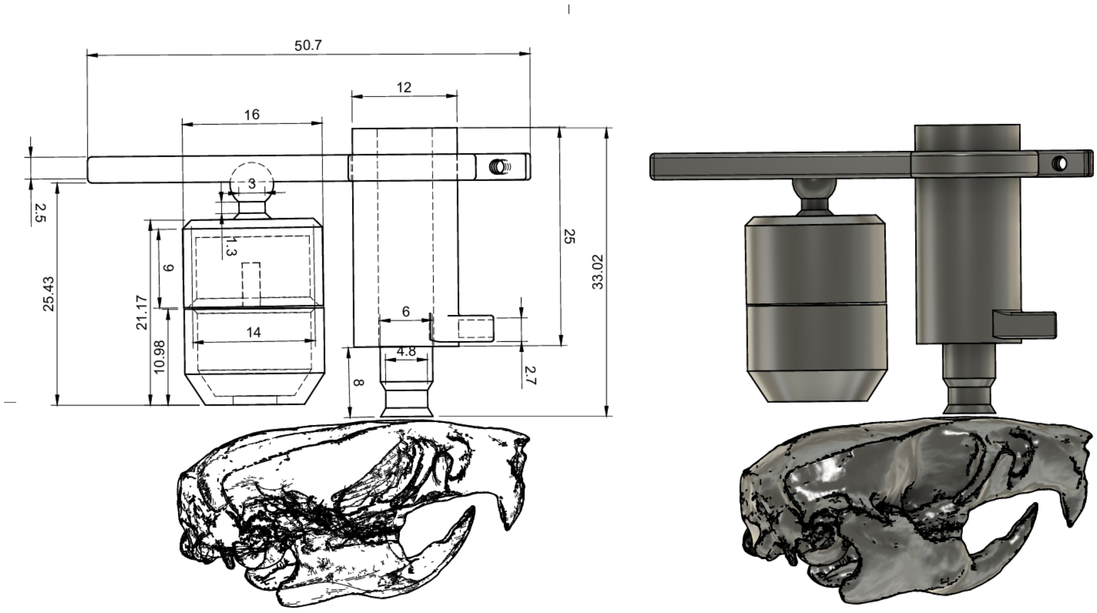


Figure 3.11: Side views of the assembled headgear above the rat skull model with a drawn and a shaded version. The unit of the dimensions is mm.

for the surgery once printed.

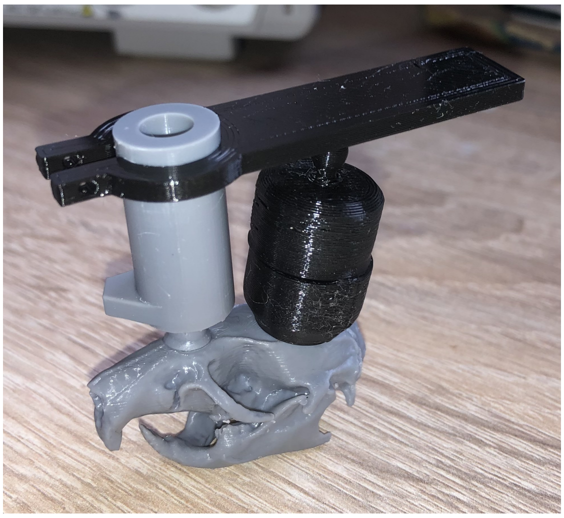


Figure 3.12: The assembled headgear with the 3D-printed components.

In our system, stimulation is not the sole feature, as presented in the original headgear concept. We integrate real-time EEG measurements alongside the stimulation. These measurements are obtained through small screws implanted into the skull to record the signals. Cables connected to the screws are routed into a connector, which is also mounted on the animal's head. This connector then couples the signals to the amplifier. The precise positioning of the connector in relation to the headgear components is initially experimented with using the 3D skull model. However, the final placement can only be determined during the actual procedure, as it depends on the physical constraints and positioning within the animal's anatomy. This introduces some level of uncertainty in the implementation of the system. To mitigate such issues, a simpler design is proposed that builds upon the completed cone element. A thin-walled cylinder is added to the cone, with the lengths of its walls slightly extended. This component would be 3D printed from a more flexible material, allowing for better adaptation to the contours of the skull surface. Once mounted with dental cement or glue, the transducer could be docked inside the cylinder for stimulation. This setup would also create more space for the EEG connector, improving the overall system layout and functionality. However, this would not be ideal from the animal's perspective, as it should carry a much larger device, probably in an uncomfortable placement.

Chapter 4

Results

This section evaluates the performance of the developed hardware-software solution. The tests aimed to assess the effectiveness of the system in real-time EEG signal processing and spindle detection, as well as to compare its accuracy with results obtained in an offline environment.

4.1. Online Evaluation of the Sleep Spindle Detector

In the previous chapters, the detailed structures of the hardware and software were presented. In the following analysis, the developed device was evaluated in an environment designed to approximate realistic measurement conditions. During this evaluation, the oscilloscope was used as a substitute for an analog signal source, while the LED served as an output indicator. The amplifier was excluded from the setup due to the challenges of relocating it, given its size. However, prior testing confirmed its proper functionality, ensuring it would not cause issues later. The system consisting of the oscilloscope, the ADC, the Raspberry Pi and the LED was assembled based on the previous descriptions in Section 3.3.1. The data used for testing was loaded into the oscilloscope via a USB drive formatted with the FAT file system. The USB drive must have had a maximum capacity of 244 MB to be recognized. To ensure the data closely resembled real EEG signals, a signal from the datasets utilized in the offline evaluation was modified as detailed in the following subsection.

4.1.1 Data

Two types of signals were used for the test data: one containing synthetic spindles embedded within real EEG data and the other containing original, unaltered spindles. The background signal for both is from the mice dataset, specifically from the measurement referred to as RMK0000014_saline. This was originally sampled with a 2000 Hz frequency. The oscilloscope could read up to 6,000 points with the external signals loaded via a USB drive. Cutting 6,000 points from a 2000 Hz signal provides a 3-second-long piece. That is too short to realize a realistic signal with multiple sleep spindles. To expand the range for analysis, the signals were resampled at 200 Hz, extending the duration to 30 seconds. Although the signals are no longer than 30 seconds, the oscilloscope generates an infinite loop from the valid points. This looping is useful to do longer measurements, but introduces additional challenges, such as uncertainty about the start of the original signal. To address this, the beginning of the signal was marked by changing the first 20 data points to an outlier value (around 0.1 V), allowing consistent comparisons with offline results. The significance of this will be further explained in the following subsections. Prior to resampling, a low-pass Butterworth filter with a cutoff frequency at half of the new sampling rate was applied to prevent aliasing. The signals were written into a binary file containing 16-bit integers. They were structured to include a header section specifying the number of data points, extreme values, and a baseline, allowing the oscilloscope's arbitrary function generator to read them. The menu of this also enabled control over the frequency, period, offset and peak-to-peak voltage of the signal. The period could be set to 30 seconds with the corresponding frequency value, to match the length of the transmitted signal to the original one. The peak-to-peak voltage and offset were observed during the signal generation and the proper values were selected. The setting of these values was essential for the accurate EEG signal simulation. The process of generating test data, from adding spindles to the background signal to setting the final parameters on the oscilloscope, is illustrated in Figure 4.1. Signals are always analyzed offline prior to being loaded into the oscilloscope. This offline analysis, a crucial step for online evaluation, is emphasized in red to highlight its importance.

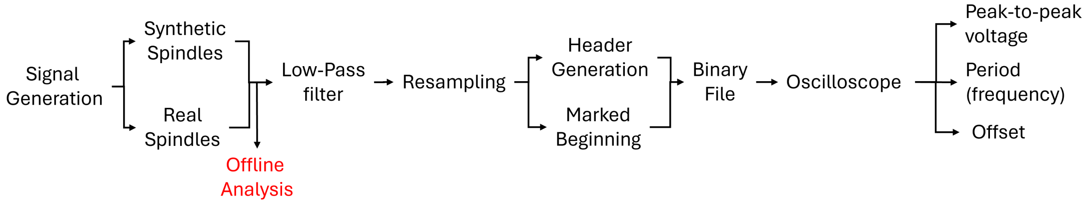


Figure 4.1: The process of generating signals for the online tests.

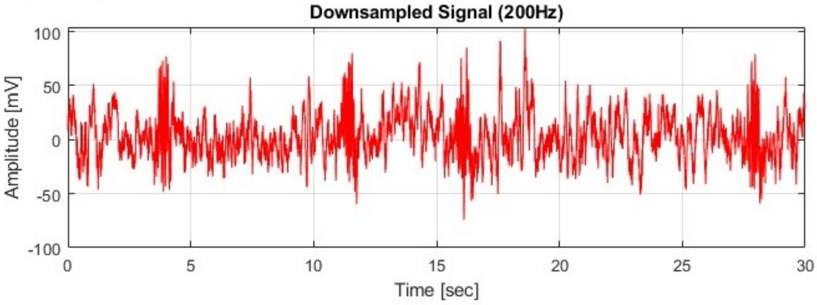
4.1.2 Synthetic Signal

Synthetic spindles were generated using a MATLAB script based on the description in Kulkarni *et al.*'s work [24]. These spindles were crucial for online evaluation, as their locations can be manually identified with ease, reducing result uncertainty. This approach greatly aids program debugging by allowing real-time confirmation of detections. Given the limited size of our dataset, this approach proved particularly advantageous for introducing novel events into the system, thereby enhancing the robustness and reliability of the detection process.

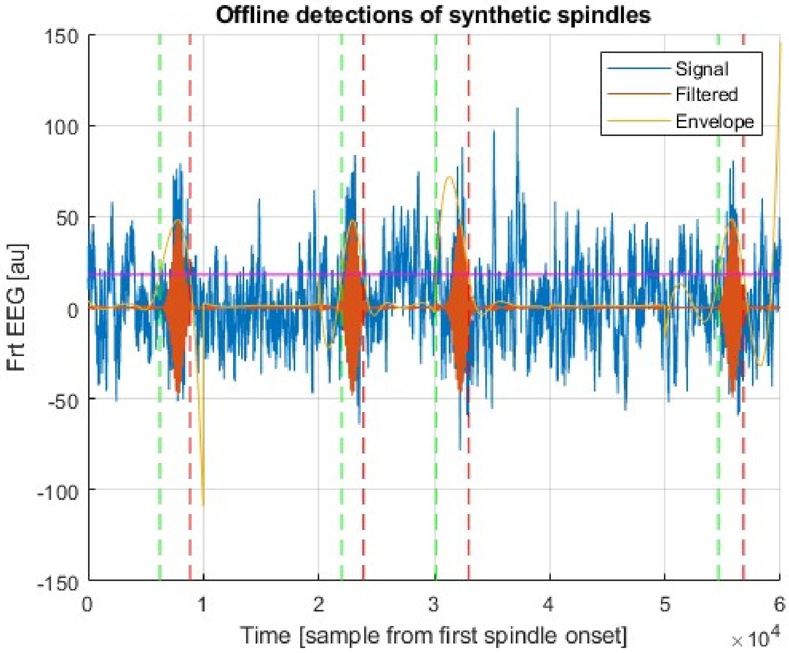
One synthetic signal was created by extracting a segment from the original EEG signal and resampling it with 200 Hz. Four synthetic spindles were added. The offline detection method (My Detector) was automatically applied to evaluate the algorithm's effectiveness on this modified data. In Figure 4.2, the synthetic, downsampled signal is displayed (4.2a) alongside the processed version (4.2b). The outputs of the filtering and envelope fitting are both shown, with additional reference lines indicating the threshold and the offline detections identified by My Detector. The green color marks the beginning, while red denotes the end of the event. The four added spindles are easily identifiable, and the offline detector clearly performs well. The beginnings and the durations of the detected spindles were saved to be used during the evaluation of the online detector. They can be observed in Table 4.1.

After the offline analysis, the generated signal was transferred to a USB drive formatted with the FAT file system. Using the oscilloscope's arbitrary function generator menu, a 30-second period, a 210 mV peak-to-peak amplitude, and a 4 mV offset were configured. The oscilloscope's output is connected to the first input channel of the ADC, allowing the Raspberry Pi to read the signal in alignment with previous settings.

The parameters of the software were revised to fit the new signal. The gain of the ADC was set to 16 to better suit the input voltage range. The data rate of the ADC was 860 Hz, but the reading only happened at every 0.005 second, realizing a 200 Hz sampling rate. Other parameters were left at their default values, and only the output file name had been set before the measurement was started. As the measurement starts, a new window pops up displaying the EEG signal in real-time. When a spindle is detected, two dashed vertical lines appear on the screen: red indicates the beginning of the spindle, while blue marks its end. Additionally, the relative detection times, calculated from the measurement start, are displayed in the command window. To provide a visual indicator, the LED is turned on for 0.5 seconds. The voltage and time values are continuously



(a) Downsampled signal with synthetic spindles.



(b) Detections made on the synthetic signal by My Detector offline.

Figure 4.2: Analysis of synthetic spindles offline.

written into the CSV file, where the locations of the spindles are marked. Figure 4.3 shows the real-time EEG plot including the detection lines. Direct comparison with the offline results can be challenging to make, because the infinite loop generated by the oscilloscope introduces uncertainty regarding the start of the signal. Therefore, the captured signal does not align perfectly with the original seen on Figure 4.2. The small segment of outlier values, which are easy to find, was incorporated into the data to solve this issue. On Figure 4.3 a black arrow marks the artificial data points, which correspond to the beginning of the offline signal. Moreover, the end of the signal was not displayed because of the nature of the acquisition and visualizing functions, but it was not missed. Also note that normally, a 10-second window is used, it was only changed for the sake of better illustration.

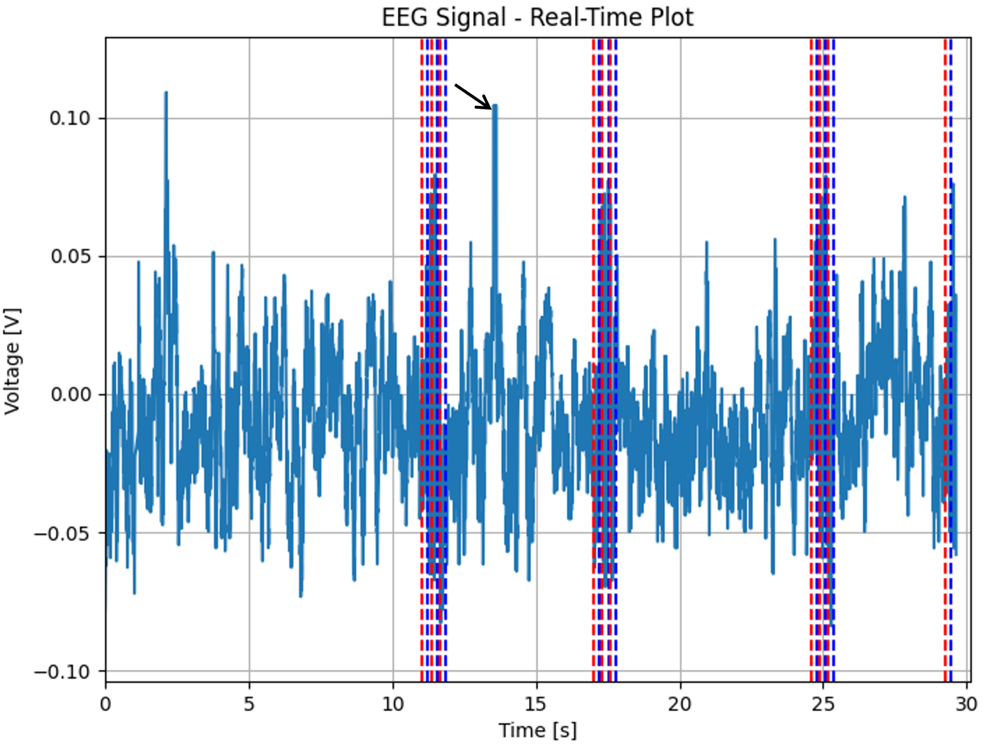


Figure 4.3: Detections made on the synthetic signal by My Detector online.

After the measurement, the signal was reconstructed from the saved CSV file to make further analysis possible. Spindles were combined if the interval between them was less than 0.3 seconds. This approach was feasible because the stimulation duration is likely to be at least 0.3 seconds, making the distinction negligible in the final outcome. Figure 4.4 shows the reconstructed signal, with the identified spindles highlighted in light blue and the start of the original signal indicated by the black arrow. It is visible to the naked eye that the detections align with the spindles quite well.

The evaluation of these detections was conducted on an extended measurement, where the oscilloscope's loop was recorded for 300 seconds. This recording included the signal repeated 10 times, containing a total of 40 spindles. To evaluate the results, the reconstructed signal was shifted to align with the original timestamps. First, the marked data points exceeding 0.1 V were located and the time of the first such point was subtracted from all the others, making it the zero time point. Time values which became negative were added 300 s to transfer them to the end. This alignment allowed for direct comparison of spindle detections between the online and offline analysis. The start times and durations of the spindles, recorded during the offline detections, provided a reference for the expected spindle occurrences. Table 4.1 shows this information.

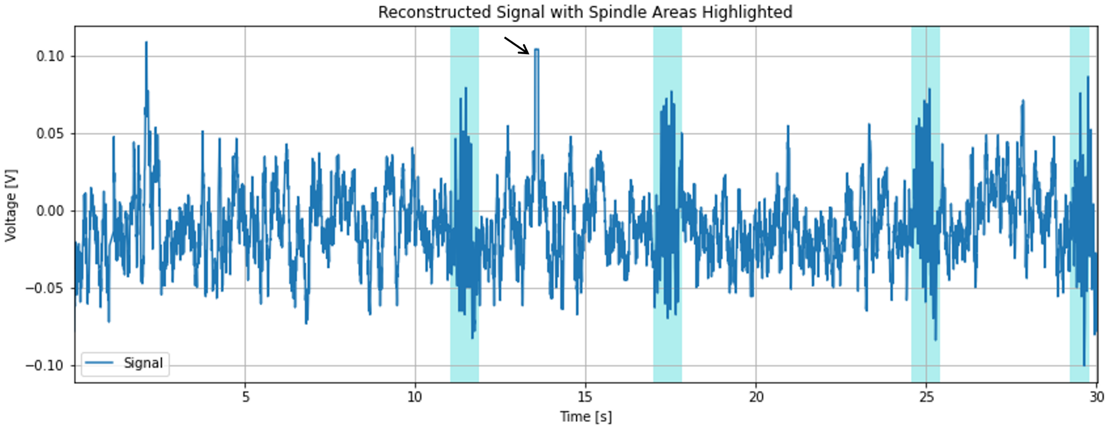


Figure 4.4: Reconstruction of the real-time detections on the synthetic spindles.

	Start [s]	Duration [s]
1. Spindle	3.125	1.3
2. Spindle	10.945	0.93
3. Spindle	15.185	1.295
4. Spindle	27.51	0.92

Table 4.1: The start and duration times of the generated synthetic spindles on the 30-seconds-long signal.

Because the same signal was replayed 10 times over the 300-second period, the spindles also repeated periodically. For instance, the first spindle appeared at 3.125 seconds and then recured at 33.125, 63.125, and so on, up to 283.125 seconds, maintaining a consistent duration of 1.3 seconds. Similarly, the second spindle was visible at 10.945, 40.945, 70.945, and so forth, always lasting 0.93 seconds. This periodicity applied to all other spindles as well. Using this logic the expected time intervals of spindles can be compared to the start times and durations of detected spindles. Table 4.2 contains the quantified version of this comparison. First, the average duration of the events over one period (30 s) were calculated and are displayed in the first two coloumns. The difference between the original and online durations can provide insight into the extent of false negative data points. For instance, the largest discrepancy was observed for spindle number 1, with an average of approximately 0.42 seconds missing from its 1.3-second duration. In contrast, for spindle 4, the difference was only 0.04 seconds per event. These results indicate that the occurrence of false negatives was minimal. The next coloumns include the number of detections made on each spindle. It was observed that a single spindle is often identified as multiple shorter detections, which can inflate the total count. To address this, the average number of detections corresponding to each original spindle was calculated. A detection was classified as a hit if any part of it overlapped with the intervals defined by the original spindle timings or their periodically shifted versions. This classification emphasizes the importance of detections within these intervals for triggering timely stimulations, as the goal is to stimulate during or shortly after a spindle. The results show that all 40 spindles were successfully identified, with each detection corresponding to a genuine event and with a minimum amount of false negative points.

	Original Duration [s]	Online Duration [s]	Original Count	Online Count	Average Detections per Original Spindle	Hits
1. Spindle	1.3	0.878	10	30	3.0	10
2. Spindle	0.93	0.795	10	30	3.0	10
3. Spindle	1.295	0.874	10	30	3.0	10
4. Spindle	0.92	0.881	10	20	2.0	10

Table 4.2: Results of the online detection on synthetic spindles.

The quantified results confirmed the accuracy of the system. During the 300-second-long measurement the detector continuously made precise detections, generated a file that was well-suited for the analysis and maintained a real-time display without lagging.

However, although accurate detections are essential, they only represent part of the success. The timing delay between spindle onset and subsequent stimulation can be a critical factor in the experimental outcomes. Minimizing this delay ensures that the stimulation aligns more closely with the natural spindle activity, potentially enhancing the experiment's efficacy and relevance. In practice, the stimulation timing can be approximated by the detection moment, as the output is immediately available on the GPIO pin. Analysis of the saved data revealed an average delay of 0.294 ± 0.0302 seconds from spindle onset and 0.088 ± 0.0294 seconds from offset until detection occurred. The estimated maximum acceptable delay from the offset for the experiments is 0.5 seconds, hence the results were approved.

4.1.3 Real EEG Signal

To simulate the real experimental environment as closely as possible, unaltered EEG data was also used to evaluate online detection. This provided us with the advantage of knowing the expert-annotated spindles beforehand, allowing us to validate our results without solely relying on offline detections. The signal underwent the same downsampling and low-pass filtering as previously described, and a binary file was saved using the same procedure. Figure 4.5 shows the downsampling process, where the signal's characteristics remained the same in the time domain, but the number of data samples, as well as the number of frequency components, were reduced. In the lower right corner of the figure, the peak in the spindle frequency band (9–16 Hz) is visible, confirming the existence of the spindles. The beginning was marked again with 20 points of 0.1 V. The key difference from the synthetic signal is that another segment of the original data, rich in expert-detected spindles, was selected for analysis.

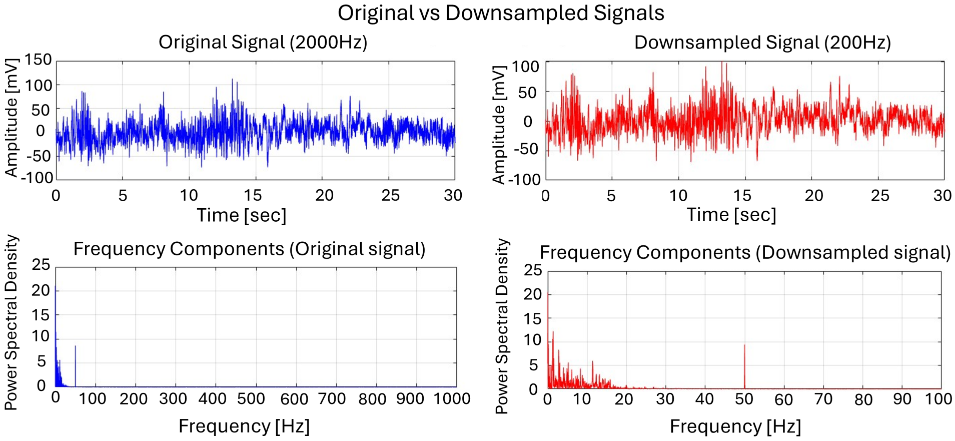


Figure 4.5: Downsampling real EEG data from 2000 Hz sampling frequency to 200 Hz.

Before being loaded into the oscilloscope for real-time measurements, the selected signal segment was automatically analyzed using the offline detector, incorporating expert annotations. Figure 4.6 presents the raw signal alongside its processed version, which includes filtering and envelope fitting. The threshold is indicated by a solid horizontal line. Dashed vertical lines show spindle detections by the algorithm, while solid vertical lines represent expert annotations. For each event, the beginning is marked with green lines, and the end is marked with red lines. Within this segment, the algorithm detected four spindles. Among these, three exhibit substantial overlap with expert-identified spindles, while one appears to be a false positive. To enhance the clarity of the following analysis, the spindles were assigned numbers and are marked with color coded lines at the top of the figure. Oscillations classified by the expert were labeled as 1, 2, and 3, while the false positive event was designated as number 4.

The start and duration times of these events were saved, just as in the previous anal-

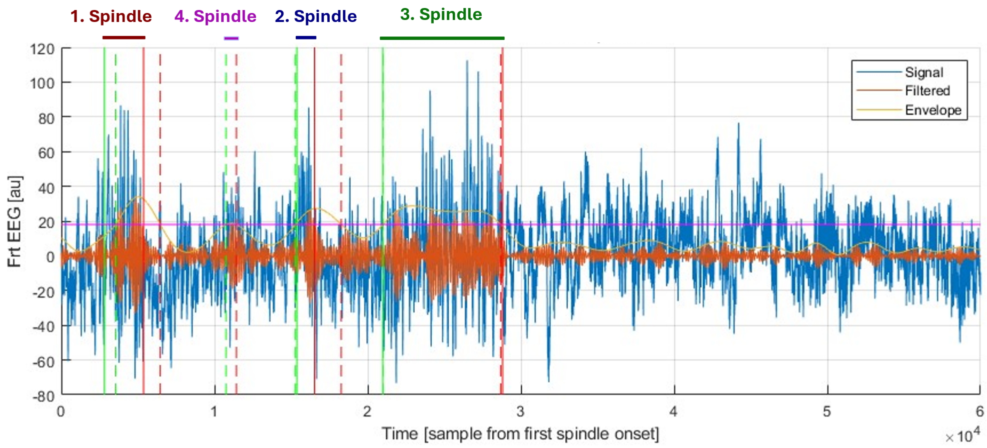


Figure 4.6: Offline detections of real spindles.

ysis, and are displayed in Table 4.3. It first lists the spindles identified by the expert, followed by any false positive events detected by the offline algorithm. This categorization emphasizes their original classification while serving as a reference for the online detector. Detecting expert-identified spindles is the primary objective of the system, but successfully identifying offline-detected spindles demonstrates that the online mechanism performs similarly to the offline one. This is expected, as the online algorithm is an adaptation of the offline detector. Furthermore, additional detections might indicate a faulty mechanism; however, analyzing their timing in detail may uncover alternative explanations. All in all, this table provides valuable insights for estimating the timing of online detections. For example, during a 300-second measurement with a 30-second periodicity, it predicts that the first spindle will appear at 1.411 seconds, 31.411 seconds, 61.411 seconds, and so on, with the same logic applying to subsequent events.

	Start [s]	Duration [s]
1. Spindle	1.411	1.276
2. Spindle	7.69	0.577
3. Spindle	10.492	3.911
4. Spindle	5.379	0.336

Table 4.3: The start and duration times of the real spindles on the 30-second-long signal.

After the offline analysis, the generated binary file was loaded into the oscilloscope. The period was again set to 30 seconds, the peak-to-peak voltage was 185 mV and the offset was 20.5 mV. The parameters of the software were not changed since analyzing the synthetic signal, only the output file was renamed. The measurement was conducted the same way as before, and the captured data was subsequently saved in a CSV file. Upon finishing the measurement, Figure 4.7 was displayed. To improve the clarity of the detections, the corresponding spindle events are labeled at the top with their respective numbers and colors. Any additional detections that do not correspond to previously identified spindles were assigned new numbers and colors for distinction. The extra voltages marking the beginning of the signal is shown with a black arrow around the 19th second.

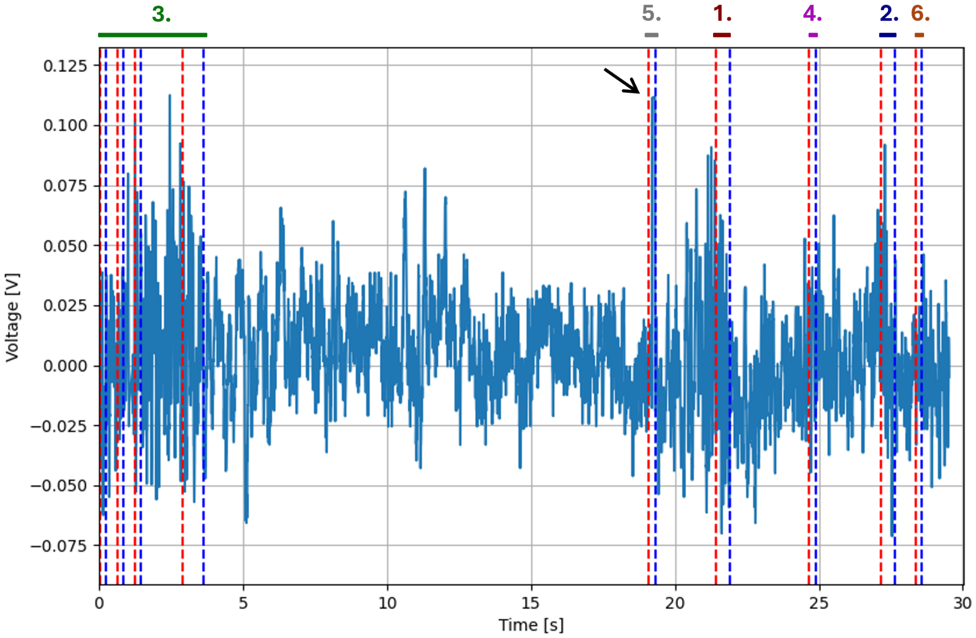


Figure 4.7: Online detections of real spindles.

The captured signal was reconstructed using the CSV file, with spindles merged if the interval between them was less than 0.3 seconds. Figure 4.8 presents the reconstructed version with the artificial points marked with an arrow and the spindles having the same number and color code as in Figure 4.7.

Next to the known spindles, two additional ones are visible. One, around the 19th second, is equivalent to the marked start of the original signal indicated by the arrow. This detection can be disregarded given the artificial nature of these points. The other extra detection is around the 28th second, within 1 s distance to spindle number 3. This may suggest that the event is an actual extension of the third spindle, which was overlooked by the expert. However, it is also plausible that the discrepancy arises due to a fault in

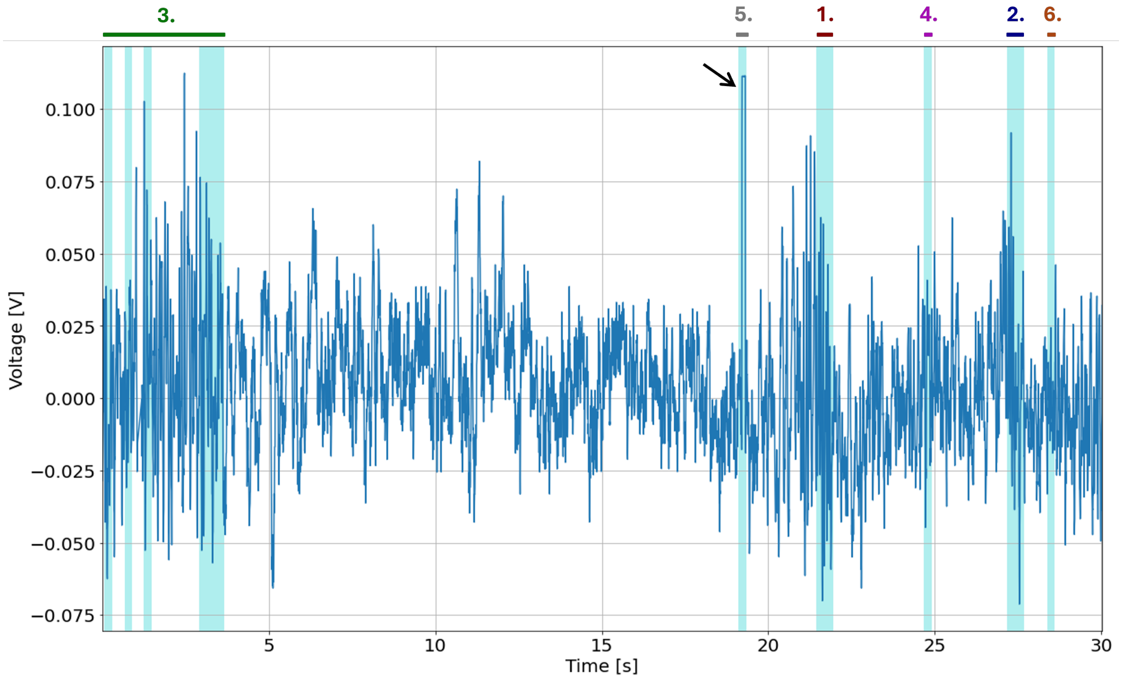


Figure 4.8: Reconstruction of the online detections made on real EEG data.

the algorithm.

As evident, visually assessing performance and identifying true spindles becomes significantly more challenging in this context, emphasizing the importance of quantitative evaluation. This was performed on a 300-second-long measurement, and the results are collected in Table 4.4. The data was captured and reconstructed the same way as before. The online detections from the 300-second measurement were analyzed based on the time intervals suggested by Table 4.3. Similar to the synthetic signal, the original durations of each event within one period were compared to the online results. It shows, that for the third spindle, an average of 2.04 seconds was missed during each period. This difference corresponds to many false negative data points, highlighting the detector's weakness when it comes to longer spindles. Additionally, some spindles, such as the second and fourth, exhibited false positive points, as their duration exceeded the original length. The periodicity of the looped signal resulted in each original spindle being repeated 10 times, as reflected in Table 4.4 under the column labeled "Original Count". Based on the known start and end times of the recurring spindles, the number of detections made within these intervals was counted and recorded under the "Online Count" column in Table 4.4. Since multiple detections can overlap with a single spindle, the average number of detections per spindle per period was also calculated. This pointed out that the third spindle was not always identified with the same number of detections, which is a result of the real-time sampling and processing. A "hit" was registered if any part of an original spindle was covered by a detection.

	Original Duration [s]	Online Duration [s]	Original Count	Online Count	Average Detections per Period	Hits
1. Spindle	1.276	0.863	10	10	1.0	10
2. Spindle	0.57	0.849	10	10	1.0	10
3. Spindle	3.911	1.871	10	52	5.2	10
4. Spindle	0.336	0.569	10	10	1.0	10
5. Spindle	-	0.569	-	10	1.0	-
6. Spindle	-	0.574	-	10	1.0	-

Table 4.4: Results of the online detection on signal containing real, expert annotated spindles.

The results indicate that all spindles originally identified by the experts were successfully detected. The false positive event observed in the offline analysis also reappeared in the online detections, demonstrating comparable performance between the two approaches. Additionally, two extra spindles were consistently identified. One of these, located approximately 1 second before the third expert-detected spindle, which might mean it is related to that event. The second extra spindle corresponds to the marked beginning of the signal, as noted earlier, and can be disregarded since it would not occur in real EEG data.

The detections aligned well with expectations derived from the offline analysis. Event recall was perfect, and the precision closely matched the offline results, though there is potential for further optimization. This dataset proved significantly more challenging for the system compared to the synthetic one, as it included longer and more diverse spindle phenotypes. This complexity was also reflected in the stimulation delays.

By using the saved detection timestamps, delays were calculated relative to spindle onset and offset times. The analysis revealed an average delay of 0.45 ± 0.723 seconds from the disappearance of spindles and 0.76 ± 0.861 seconds from their onset. While these values fell within the permissible range (estimated to be 0.5 seconds from the end), they were near the upper limit. Since the goal is to minimize delays, additional statistical assessments were conducted for clarity. Using delays rounded to two decimal places, the mode of delays was found to be 0.09 seconds from the end and 0.30 seconds from the start. These results imply that, in general, detections occur promptly; however, certain types of spindles consistently take longer to be identified, highlighting areas for improvement in the detection algorithm.

Chapter 5

Discussion

This thesis presents my work for developing a device to detect sleep spindle oscillations online and use these detections to trigger a neural stimulator. The developed device has proven effective in detecting sleep spindles in rodents' EEG data. However, certain limitations are inherent in the current system. Addressing these constraints is key to enhancing its performance. In this chapter, I will discuss potential theoretical solutions that could be implemented in future iterations of the system.

5.1. Limitations and Future Development

5.1.1 Software

Detection Algorithm

Despite the promising performance of My Detector in both offline and online environments, it remains one of the simplest algorithms applied to this problem. It is important to acknowledge that even a rudimentary neural network used during evaluation often outperformed the detector, despite not being trained on domain-specific data. This suggests that higher F1 scores and overall accuracy may be achievable once the network is properly trained on rodent-specific datasets. Such results could potentially match those reported in the original study [19]. Moreover, other machine learning-based algorithms have been described in the literature, showing compelling results in similar applications [25][24]. Incorporating such advanced techniques could significantly reduce uncertainties and enhance the overall system. In particular, adopting machine learning models specifically designed for real-time systems [24] could provide substantial improvements.

Graphical User Interface

The real-time visualization function efficiently displays the captured signal alongside the detected spindles. For a more advanced solution, this functionality could be integrated into a graphical user interface (GUI), allowing users to specify input variables (such as output file names) and log detections seamlessly. Incorporating start and stop buttons would further enhance user-friendliness. Implementing such features would necessitate a more sophisticated data transfer mechanism to enable seamless communication between the processes and the GUI.

Storage Function

Extensive experimentation was conducted to optimize the storage function across multiple criteria. Despite these efforts, the current implementation remains suboptimal as post-processing is still required. Ideally, the storage system should ensure that time and voltage values are in strictly ascending order, with corresponding Boolean values clearly marking spindle occurrences. With the current CSV format, retroactive writing or modifying specific entries directly is infeasible. A potential solution, rewriting the entire file, would impose a significant computational burden. Alternative formats like TXT or binary were dismissed due to their unsuitability for managing large datasets, particularly for long-duration experiments. HDF5 emerged as a promising option for its ability to handle large datasets efficiently. However, achieving an accurate and fully functional implementation proved challenging, leaving room for development in this area. The precise timing of stimulations should also be recorded for future analysis. Currently, these times are displayed in the command window and manually copied into a file at the end of the process. Utilizing the HDF5 file format could address this issue by enabling the storage of stimulation timestamps alongside other data, enhancing both organization and accessibility.

5.1.2 Hardware

Amplifier

The current hardware setup is compact and performs reliably, but there is room for improvement, especially in the amplifier. The amplifier is significantly larger than other components, limiting the system's portability and usability. Smaller, commercially available amplifiers, such as the CGX CAMP Compact Amplifier [26], are affordable and could improve the solution by offering a broader range of gain settings. This would enhance resolution and overall system robustness. An even more effective solution could involve designing a custom amplifier circuit integrated with the ADC module. Such a design would reduce the device's size and complexity, further optimizing its performance.

Connections

Connections are currently made using a breadboard and jumper cables, which is neither stable nor practical, as it requires reassembly before each measurement. Implementing a custom circuit could seamlessly address the connections between the amplifier and ADC. Additionally, this solution would provide an opportunity to design a more robust interface for connecting to the Raspberry Pi and/or the transducer.

5.1.3 Headgear

The current headgear design is unsuitable for extended behavioral studies due to its reliance on glue for attachment, which limits reproducibility and renders it impractical for repeated use. A redesigned headgear incorporating an adjustable arm and customizable drop length, as suggested by Lee et al. [22], would enhance versatility and precision. This design would allow researchers to fine-tune the stimulation angle, improving both usability and experimental consistency. Additionally, the transducer could be housed in an adaptable holder to accommodate various sizes, with a cone structure facilitating transmission. A specialized material currently under development could revolutionize this approach, offering improved efficiency. Alternatively, integrating ring transducers, as demonstrated by Hyunggug Kim et al. [27], could reduce the weight carried by the animal and enhance focusing precision, further optimizing experimental outcomes.

5.1.4 Experimental Environment

Despite our best efforts to simulate analog signals using the oscilloscope, it remains an approximation. The next step involves measuring EEG signals from surgically implanted animals. However, animal experiments introduce significant uncertainty, with many factors potentially affecting performance outcomes. These include the quality of the surgery or screws, the animal's characteristics, the stability of the physical system, electrical noise of the environment, and numerous other unpredictable influences on the final results. The further research should be conducted by considering these aspects.

Chapter 6

Summary

To summarize, a reliable real-time sleep spindle detector was developed. Various algorithms were evaluated based on their offline performance, helping us to select an optimal one for online detection. This detector was then implemented on hardware and modified to meet the demands of real-time applications. The microcomputer running the processing code was equipped with additional components to create a system capable of analog signal processing, which included the integration and testing of an ADC and amplifier. An ultrasonic transducer was added for stimulation purposes, and testing these components required building a circuit with an oscilloscope and LED.

The online detector was tested on two types of signals. First, synthetic spindles were generated over real EEG backgrounds, allowing for visual performance validation. Next, snippets of real EEG signals were used, and statistical parameters were calculated for both offline and online detections to compare results, showing a stable and accurate performance. Finally, a 3D-printed headgear was designed to hold the transducer on the head of the animal during awake experiments.

6.1. Future Plans

In the future, we aim to use this system in actual animal experiments. By selecting a behavioral test for the rodents, their performance can be assessed with and without tFUS intervention. The device would be active during the animals' sleep, with the headgear holding the transducer in place over the animal's head and EEG screws connected to the ADC input channel. EEG data would be processed in real-time on the Raspberry Pi, triggering the transducer upon spindle detection. This approach could significantly advance ultrasound stimulation and sleep research.

Another potential application of this system lies in its adaptability to other signal processing tasks. The detection algorithm can be easily modified to identify different phenomena, making it versatile for various research areas.

Bibliography

[1] R. Polanía, M. A. Nitsche, and C. C. Ruff, "Studying and modifying brain function with non-invasive brain stimulation," *Nature Neuroscience*, vol. 21, no. 2, pp. 174–187, 2018. DOI: `10.1038/s41593-017-0054-4`. [Online]. Available: `https://doi.org/10.1038/s41593-017-0054-4`.

[2] M. Meneghetti, F. Gudmundsen, N. S. Jessen, *et al.*, "Mapping whole brain effects of infrared neural stimulation with positron emission tomography," *Imaging Neuroscience*, vol. 1, pp. 1–17, Dec. 2023, ISSN: 2837-6056. DOI: `10.1162/imag_a_00052`. eprint: `https://direct.mit.edu/imag/article-pdf/doi/10.1162/imag\_a\_00052/2200611/imag\_a\_00052.pdf`. [Online]. Available: `https://doi.org/10.1162/imag%5C_a%5C_00052`.

[3] E. N. Harvey and A. L. Loomis, "High frequency sound waves of small intensity and their biological effects," *Nature*, vol. 121, no. 3051, pp. 622–624, 1928. DOI: `10.1038/121622a0`. [Online]. Available: `https://doi.org/10.1038/121622a0`.

[4] G. Darmani, T. Bergmann, K. Butts Pauly, *et al.*, "Non-invasive transcranial ultrasound stimulation for neuromodulation," *Clinical Neurophysiology*, vol. 135, pp. 51–73, 2022, ISSN: 1388-2457. DOI: `https://doi.org/10.1016/j.clinph.2021.12.010`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S1388245721008920`.

[5] A. Sirota, J. Csicsvari, D. Buhl, and G. Buzsáki, "Communication between neocortex and hippocampus during sleep in rodents," en, *Proceedings of the National Academy of Sciences*, vol. 100, no. 4, pp. 2065–2069, Feb. 2003, ISSN: 0027-8424, 1091-6490. DOI: `10.1073/pnas.0437938100`. [Online]. Available: `https://pnas.org/doi/full/10.1073/pnas.0437938100` (visited on 10/01/2024).

[6] S. Astori, R. D. Wimmer, and A. Lüthi, "Manipulating sleep spindles – expanding views on sleep, memory, and disease," *Trends in Neurosciences*, vol. 36, no. 12, pp. 738–748, 2013. DOI: `10.1016/j.tins.2013.10.001`.

[7] S. Dong, Z. Xie, and Y. Yuan, "Transcranial ultrasound stimulation modulates neural activities during nrem and rem depending on the stimulation phase of slow oscillations and theta waves in the hippocampus," *Cerebral Cortex*, vol. 33, no. 14, pp. 8956–8966, May 2023, ISSN: 1047-3211. DOI: `10.1093/cercor/bhad174`. eprint: `https://academic.oup.com/cercor/article-pdf/33/14/8956/50822691/bhad174.pdf`. [Online]. Available: `https://doi.org/10.1093/cercor/bhad174`.

[8] Schmeichel Lab, *Research - schmeichel lab*, Accessed: 2024-11-10. [Online]. Available: `https://schmeichel-lab.weebly.com/research.html`.

[9] A. L. Loomis, E. N. Harvey, and G. Hobart, "Potential rhythms of the cerebral cortex during sleep," *Science*, vol. 81, no. 2111, pp. 597–598, 1935. DOI: `10.1126/science.81.2111.597`. eprint: `https://www.science.org/doi/pdf/10.1126/science.81.2111.597`. [Online]. Available: `https://www.science.org/doi/abs/10.1126/science.81.2111.597`.

[10] M. Steriade, D. A. McCormick, and T. J. Sejnowski, "Thalamocortical oscillations in the sleeping and aroused brain," *Science*, vol. 262, no. 5134, pp. 679–85, Oct. 1993, ISSN: 0036-8075 (Print) 0036-8075 (Linking). DOI: `10.1126/science.8235588`.

[11] O. Eschenko, M. Mölle, J. Born, and S. J. Sara, "Elevated Sleep Spindle Density after Learning or after Retrieval in Rats," en, *The Journal of Neuroscience*, vol. 26, no. 50, pp. 12 914–12 920, Dec. 2006, ISSN: 0270-6474, 1529-2401. DOI: `10.1523/JNEUROSCI.3175-06.2006`. [Online]. Available: `https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.3175-06.2006` (visited on 02/23/2024).

[12] S. Fogel and J. Jacob, "The role of sleep spindles in simple motor procedural learning," *Sleep*, vol. 25, A279–A280, Jan. 2002.

[13] Z. Clemens, D. Fabó, and P. Halász, "Overnight verbal memory retention correlates with the number of sleep spindles," eng, *Neuroscience*, vol. 132, no. 2, pp. 529–535, 2005, ISSN: 0306-4522. DOI: `10.1016/j.neuroscience.2005.01.011`.

[14] E. J. Wamsley, M. A. Tucker, A. K. Shinn, *et al.*, “Reduced sleep spindles and spindle coherence in schizophrenia: Mechanisms of impaired memory consolidation?” *Biological Psychiatry*, vol. 71, no. 2, pp. 154–161, 2012, Functional Consequences of Altered Cortical Development in Schizophrenia, ISSN: 0006-3223. DOI: `https://doi.org/10.1016/j.biopsych.2011.08.008`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0006322311008146`.

[15] S. C. Warby, S. L. Wendt, P. Welinder, *et al.*, “Sleep spindle detection: Crowdsourcing and evaluating performance of experts, non-experts, and automated methods,” *Nature Methods*, vol. 11, no. 4, pp. 385–392, 2014. DOI: `10.1038/nmeth.2855`.

[16] Emese Pálfi, György Lévay, András Czurkó, Balázs Lendvai, Tamás Kiss, “Acute blockade of nr2c/d subunit-containing n-methyl-d-aspartate receptors modifies sleep and neural oscillations in mice,” *Journal of sleep research*, 2020. DOI: `10.1111/jsr.13257`.

[17] Karine Lacoursea , Jacques Delfratea , Julien Beaudrya , Paul Peppardb , Simon C. Warby, “A sleep spindle detection algorithm that emulates human expert spindle scoring,” *Journal of Neuroscience Methods*, 2019. DOI: `10.1016/j.jneumeth.2018.08.014`.

[18] L. K. et al., *Implementation of the sumo (slim u-net trained on moda) model*, Accessed: 2024-11-11, 2022. [Online]. Available: `https://github.com/dslaborg/sumo`.

[19] Lars Kaulen , Justus T. C. Schwabedal , Jules Schneider , Philipp Ritter , Stephan Bialonski, “Advanced sleep spindle identifcation with neural networks,” *Nature: Scientific Reports*, 2022. DOI: `10.1038/s41598-022-11210-y`.

[20] Raspberry Pi Foundation, *Raspberry pi documentation - processors: Bcm2711*, Accessed: 2024-10-25, 2024. [Online]. Available: `https://www.raspberrypi.com/documentation/computers/processors.html#bcm2711`.

[21] Hestore.hu, *Ads1115 adc module - Hestore.hu*, `https://www.hestore.hu/prod_10041798.html`, Accessed: 2024-10-30.

[22] W. Lee, P. Croce, R. Margolin, *et al.*, “Transcranial focused ultrasound stimulation of motor cortical areas in freely-moving awake rats,” *BMC Neuroscience*, vol. 19, p. 57, 2018. DOI: `10.1186/s12868-018-0459-3`. [Online]. Available: `https://doi.org/10.1186/s12868-018-0459-3`.

[23] RISD Nature Lab, *Rat skull (muridae)*, Accessed: 2024-10-25, 2021. [Online]. Available: `https://sketchfab.com/3d-models/rat-skull-b7e100f06a284730a95bcd7248c3b8`

[24] P. M. Kulkarni, "A deep learning approach for real-time detection of sleep spindles," *Journal of Neural Engineering*, vol. 16, no. 3, 2019. DOI: `10.1088/1741-2552/ab0933`.

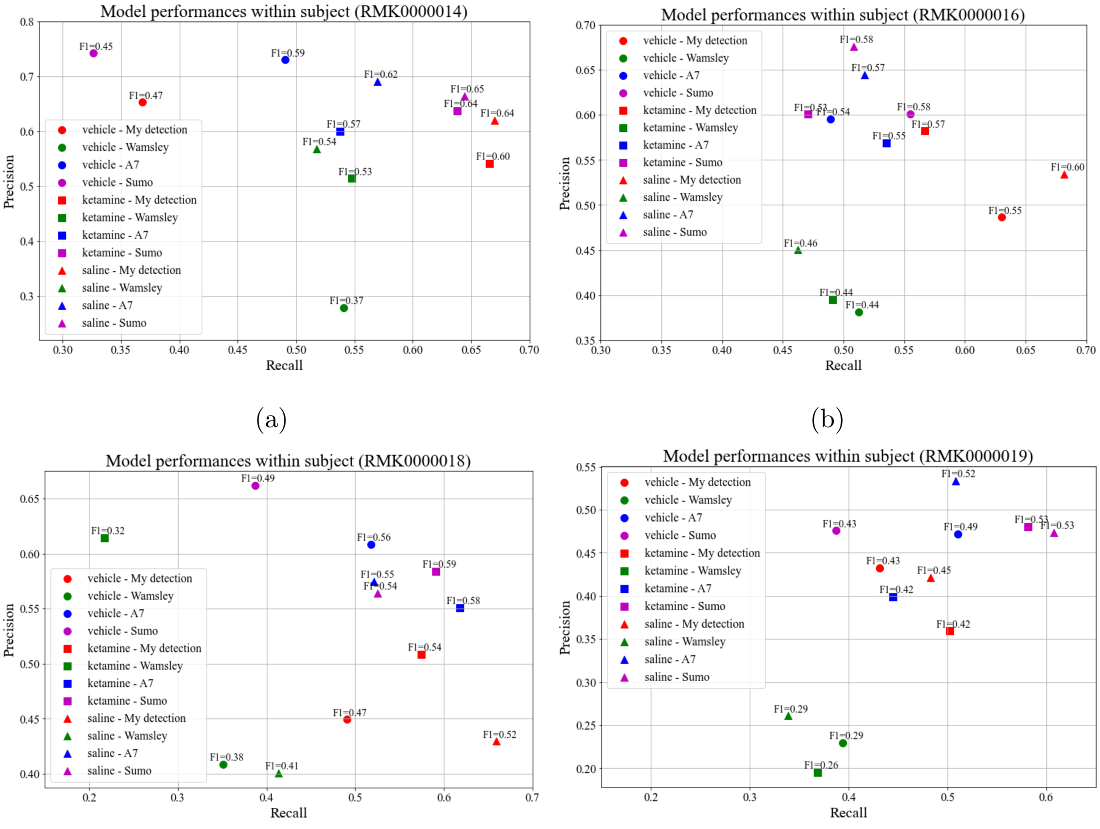
[25] N. I. Tapia-Rivas, "A robust deep learning detector for sleep spindles and k-complexes: Towards population norms," *Nature Scientific Reports*, vol. 14, no. 263, 2024. DOI: `10.1038/s41598-023-50736-7`.

[26] C. Systems, *Cgx camp compact amplifier*, `https://www.cgxsystems.com/camp`, Accessed: 2024-11-16, 2024.

[27] H. Kim, "Miniature ultrasound ring array transducers for transcranial ultrasound neuromodulation of freely-moving small animals," *Brain Stimulation: Basic, Translational, and Clinical Research in Neuromodulation*, vol. 12, no. 2, 2019. DOI: `10.1016/j.brs.2018.11.007`.

Appendix A

Appendix



(a)

(b)

(c)

(d)

Figure A.1: F1 score, precision and recall scores for the different models within subjects. The different animals are represented by their code in the brackets, while the different measurements are indicated by the name of the chemical (eg.: ketamine) used in the experiment.

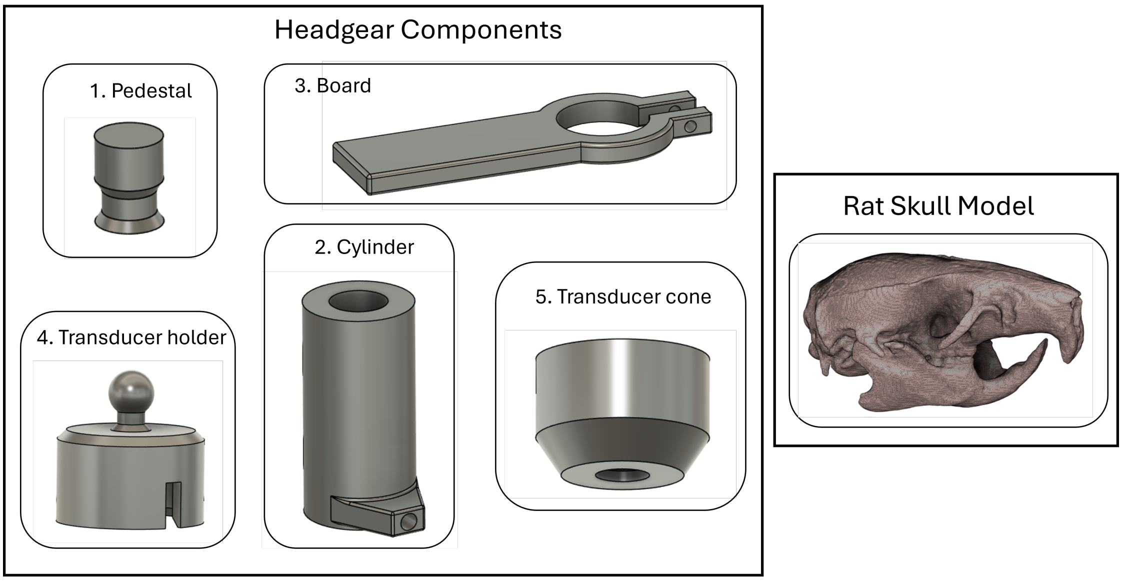


Figure A.2: Individual components of the headgear, displayed next to a rat skull model for design reference. The sizes of the parts are not proportional to the skull for illustrative purposes.

RMK0000014				
	Param1	Param2	Param3	Param4
My Detector	16.588	0.555	-	-
Wamsley et al.	15.330	0.553	3.094	1.053
A7 detector	1.732	0.478	2.665	0.573

RMK0000016				
	Param1	Param2	Param3	Param4
My Detector	19.774	0.697	-	-
Wamsley et al.	13.240	0.378	3.201	2.305
A7 detector	1.918	0.374	2.063	0.589

RMK0000018				
	Param1	Param2	Param3	Param4
My Detector	16.570	0.447	-	-
Wamsley et al.	13.353	0.412	3.249	1.097
A7 detector	1.861	0.401	2.470	0.609

RMK0000019				
	Param1	Param2	Param3	Param4
My Detector	21.358	0.656	-	-
Wamsley et al.	11.477	0.462	3.035	2.345
A7 detector	1.881	0.528	2.279	0.641

Table A.1: Parameter values across different detectors for the mice dataset, where the animals are coded as RMK0000014, RMK0000016, RMK0000018, and RMK0000019.