Computer Methods and Programs in Biomedicine 230 (2023) 107357

Contents lists available at ScienceDirect

Computer Methods and Programs in Biomedicine

journal homepage: www.elsevier.com/locate/cmpb

MEDUSA©: A novel Python-based software ecosystem to accelerate brain-computer interface and cognitive neuroscience research

Eduardo Santamaría-Vázquez [a,b,*], Víctor Martínez-Cagigal [a,b], Diego Marcos-Martínez [a], Víctor Rodríguez-González [a,b], Sergio Pérez-Velasco [a], Selene Moreno-Calderón [a], Roberto Hornero [a,b]

[a] Biomedical Engineering Group (GIB), E.T.S Ingenieros de Telecomunicación, University of Valladolid, Paseo de Belén 15, Valladolid, 47011, Spain
[b] Centro de Investigación Biomédica en Red en Bioingeniería, Biomateriales y Nanomedicina, (CIBER-BBN), Spain

ARTICLE INFO

Article history:
Received 22 July 2022
Revised 14 December 2022
Accepted 15 January 2023

Keywords:
Brain-computer interfaces
Neurotechnology
Neuroscience
Electroencephalography

ABSTRACT

Background and objective: Neurotechnologies have great potential to transform our society in ways that are yet to be uncovered. The rate of development in this field has increased significantly in recent years, but there are still barriers that need to be overcome before bringing neurotechnologies to the general public. One of these barriers is the difficulty of performing experiments that require complex software, such as brain-computer interfaces (BCI) or cognitive neuroscience experiments. Current platforms have limitations in terms of functionality and flexibility to meet the needs of researchers, who often need to implement new experimentation settings. This work was aimed to propose a novel software ecosystem, called MEDUSA©, to overcome these limitations.

Methods: We followed strict development practices to optimize MEDUSA© for research in BCI and cognitive neuroscience, making special emphasis in the modularity, flexibility and scalability of our solution. Moreover, it was implemented in Python, an open-source programming language that reduces the development cost by taking advantage from its high-level syntax and large number of community packages.

Results: MEDUSA© provides a complete suite of signal processing functions, including several deep learning architectures or connectivity analysis, and ready-to-use BCI and neuroscience experiments, making it one of the most complete solutions nowadays. We also put special effort in providing tools to facilitate the development of custom experiments, which can be easily shared with the community through an app market available in our website to promote reproducibility.

Conclusions: MEDUSA© is a novel software ecosystem for modern BCI and neurotechnology experimentation that provides state-of-the-art tools and encourages the participation of the community to make a difference for the progress of these fields. Visit the official website at https://www.medusabci.com/ to know more about this project.

1. Introduction

Neuroscience is a multidisciplinary field devoted to understanding the brain, one of the most complex research endeavors nowadays. Since the late nineteenth and early twentieth centuries, when Santiago Ramón y Cajal and other pioneers established the first steps of modern neuroanatomy, this field has witnessed impressive advances and our understanding of the central nervous system is rapidly growing [1]. This development is directly linked to the technical advances in neuroimaging, which now allow to characterize the anatomical structure and functional activity of the brain with unthinkable precision just a few decades ago [2]. Still, we are just scratching the surface, and there are countless questions that remain to be answered. Due to the complexity of brain research, neuroscientists break down the problem to analyze it from multiple perspectives, ranging from structural details at the molecular level to high level functions such as memory,

* Corresponding author at: Biomedical Engineering Group (GIB), E.T.S Ingenieros de Telecomunicación, University of Valladolid, Paseo de Belén 15, Valladolid 47011, Spain.
E-mail addresses: eduardo.santamaria@gib.tel.uva.es (E. Santamaría-Vázquez), victor.martinez@gib.tel.uva.es (V. Martínez-Cagigal), diego.marcos@gib.tel.uva.es (D. Marcos-Martínez), victor.rodriguez@gib.tel.uva.es (V. Rodríguez-González), sergio.perez@gib.tel.uva.es (S. Pérez-Velasco), selene.moreno@gib.tel.uva.es (S. Moreno-Calderón), robhor@tel.uva.es (R. Hornero).

https://doi.org/10.1016/j.cmpb.2023.107357

language or conscience [2]. Moreover, this science has benefited in recent years from the contributions of physicians, psychologists, philosophers, mathematicians or engineers to build one of the most fruitful areas of multidisciplinary research in the history of science [2].

In order to gain new insights into brain function, researchers often need to conduct sophisticated experiments while recording images or signals from the brain. In general, these experiments can be implemented using open-loop or closed-loop systems. Open-loop systems are those where the subject performs a task while at least one signal or image is being recorded, but these recordings do not affect the outcome of the experiment [3]. An illustrative example could be the recording of functional magnetic resonance imaging (fMRI) from a subject performing some mathematical calculations to study the regions implicated in this mental task. On the other hand, closed-loop experiments establish a causal relationship between the user's brain activity and the outcome of the experiment through the analysis in real time of the recorded data to extract meaningful information. Closed-loop systems are implemented using brain-computer interfaces (BCI), a novel technology that provides new ways of interaction between our brain and the environment with different applications, such as device control to increase the quality of life and independence of severely disabled people, neurorehabilitation or cognitive training, among others [3,4]. Therapies based on neurofeedback are examples of closed-loop systems that are being intensively investigated as promising non-pharmacological interventions for multiple neuropsychiatric disorders [5].

The implementation of such experiments requires specific software to handle the signal acquisition, data processing, task presentation, and feedback to the user [3]. The design and development of these tools is difficult and time consuming, requiring extensive technical knowledge of electronics or programming. Furthermore, a great flexibility is often required in this research environment to rapidly adjust the applications to particular studies or projects. Unfortunately, this expertise is often out of reach for most neuroscience research groups, who stick to the available software or rely on external entities to develop the required programs for their research, leading to delays and cost increases. This is particularly relevant in closed-loop experiments implemented with BCIs due to their technical complexity [6,7].

In this context, software tools and applications specifically designed to facilitate the implementation of neuroscience experiments are of great importance for the progress of brain research and neurotechnology. In fact, the impact that this kind of tools could have in particular fields should not be underestimated, given their ability to speed up experimentation, reduce project costs, and enable the participation of researchers without technical knowledge [8]. While there is a wide range of signal/image processing toolboxes with state-of-the-art methods (e.g., EEGLAB, Brainstorm, MNE), there are limited options for experimental design and implementation. Most of the current tools have few or none customization options and they can only be applied for specific tasks. Moreover, these programs are often distributed under proprietary terms with no compatibility between biomedical recording equipment from different manufacturers, which limits the opportunity to take advantage from all the resources available in a research laboratory.

There have been some attempts to overcome these limitations. Two examples of open-source platforms for neuroscience research with success within the BCI community are BCI2000 (https://www.bci2000.org) [9] and OpenVibe (http://openvibe.inria.fr) [8]. Although these platforms have been widely used in BCI studies for years, they have drawbacks that should be considered. Their signal acquisition module is not prepared to handle multiple input signals, which limits their application in a wide range of experiments (e.g., collaborative and competitive BCIs). Another important aspect is that their implementation in C++, a complex programming language, is not convenient to keep up with the latest developments in the BCI field. For instance, BCI2000 and OpenVibe do not offer some state-of-the-art BCI paradigms, such as those based on code-modulated visual evoked potentials (c-VEP) [10], or signal processing algorithms, such as deep neural networks [11]. Moreover, these platforms lack specific tools to create and share new applications and experiments, which are important functionalities in research environments. As a result, despite the useful help that they provided in the past decades, the community has little opportunity to contribute to their development. These limitations also apply to less-known projects that, in some cases, are barely (or no longer) maintained: BF++ (http://www.brainterface.com) [12], xBCI (http://xbci.sourceforge.net) [13] or Pyff (https://bbci.de/pyff/index.html) [14]. In this context, novel platforms with the power and flexibility to address the complexity of these challenges could drive further advances in the field.

In this article, we present MEDUSA©, a novel Python-based software ecosystem to implement BCI and neuroscience experiments that aims to overcome the previous limitations. The highlights of this open-source solution are: (1) compatibility with virtually any signal acquisition system supported by high-level functionalities built on top of the lab-streaming layer (LSL) protocol, including the possibility of recording multiple signals at the same time; (2) complete suite of BCI paradigms and cognitive neuroscience experiments, such as event-related potential (ERP) spellers based on c-VEP and P300, motor imagery (MI), neurofeedback and neuropsychological tasks (Dual *N*-back, Stroop task, Digit Span test, Corsi Block Tapping test and Go/No-go task); (3) state-of-the-art signal processing methods and models for offline and online processing, including deep neural networks and connectivity analysis; (4) developer tools to simplify the implementation of custom open-loop and closed-loop experiments; and (5) specific functionalities to share experiments and to promote open science, reproducibility, and collaboration within the community, including an app market and several discussion spaces in our website.

2. Overview of MEDUSA©

2.1. Components

MEDUSA© is a software ecosystem for the development of BCI and neuroscience experiments. It has two independent components with different goals: MEDUSA© Kernel and MEDUSA© Platform.

MEDUSA© Kernel is a Python package that contains ready-to-use methods for analyzing brain signals, including advanced signal processing, machine learning, deep learning, and miscellaneous high-level analyses. It also includes logical functions and classes to handle different biosignals, such as electroencephalography (EEG) and magnetoencephalography (MEG), save experimental data or implement standalone processing pipelines.

MEDUSA© Platform is a desktop application, also programmed in Python, that implements high level functionalities to perform experiments. It includes a modern graphic user interface (GUI) supported by advanced signal acquisition functions and real time charts. One of the most critical features is the possibility to install and create apps, which are implementations of neuroscience and BCI experiments or paradigms. Noteworthy, all these functionalities rely on MEDUSA© Kernel to perform the necessary real-time signal processing. In the following sections these characteristics are explained in detail.

2.2. Design principles

MEDUSA© has been designed and developed following three principles:

- Modularity: MEDUSA© is made of autonomous structures organized in different levels of abstraction that are connected through simple communication protocols. This allows to quickly fix or upgrade functionalities without interfering with the rest of the components. For instance, MEDUSA© Kernel provides different interfaces for low-level and high-level functions with specific implementations that facilitate independence. In MEDUSA© Platform, the design philosophy is similar. Its architecture allows the creation of new experimental protocols on demand using specific components, called apps, which are independent of real-time acquisition and visualization stages.
- Flexibility: MEDUSA© has been specifically designed as a research tool by means of an architecture that allows to perform quick experiments with new signal processing methods and feedback paradigms. In addition, we put special emphasis on the documentation and code comments, including examples and tutorials that illustrate the operation of the platform and how to develop new apps.
- Scalability: MEDUSA© is designed to update its capabilities over time without modifying unrelated parts of code thanks to the use of standardized meta-classes, which is particularly useful in a research context. This design allows the software to keep up with latest developments in the BCI field, which may include new signal processing algorithms or BCI paradigms.

2.3. Implemented in Python

MEDUSA© has been developed in Python. Currently, this high-level, open-source programming language is one of the most widely used in both research and industry due to its simplicity and open-source philosophy [15]. In comparison with other languages, such as C, C++ or Java, Python simplifies the development of complex programs at the expense of an affordable reduction in performance [16]. This is especially important in research environments, where flexibility is a key feature as new methods and experiments are constantly developed. In addition, it has a large community that develops a wide range of specific tools and libraries. MEDUSA© exploits the power of packages such as SciPy, Numpy, Scikit-learn or Tensorflow, which are the result of a joint effort to implement state-of-the-art data processing, machine learning and deep learning [15,16]. This gives MEDUSA© an important advantage over other neurotechnology platforms (e.g., BCI2000, OpenVibe), as it allows us to incorporate the latest developments in these areas directly in the software workflow.

2.4. Distribution

MEDUSA© is an open-source suite distributed under Creative Commons Attribution-NonCommercial-NoDerivs 2.0 license. It can be freely downloaded from the official website: www.medusabci.com. Additionally, the website provides updated information, documentation, tutorials and a discussion forum for the community. The official repositories of all our developments, including MEDUSA© Platform and MEDUSA© Kernel, can be found at www.github.com/medusabci.

3. MEDUSA© kernel

MEDUSA© Kernel is a Python library, available in the Python Package Index (PyPI) repository, with a complete suite of functions for signal processing. The included functions can be categorized according to their different levels of abstraction. The first level is composed of low-level functions, which are generic methods that can be used to process signals in many scenarios, including the following:

- Temporal filters: configurable infinite impulse response (IIR) and finite impulse response (FIR) filters.
- Spatial filters: common average reference (CAR), laplacian filter, multi-class common spatial patterns (CSP) and canonical correlation analysis (CCA).
- Local activation metrics: including spectral metrics, such as band power, median frequency or Shannon entropy; and non-linear features, such as central tendency measure, sample entropy, multiscale entropy, Lempel-Ziv's complexity and Multiscale Lempel-Ziv's complexity.
- Connectivity metrics: amplitude-based metrics, such as amplitude correlation envelope (AEC) and instantaneous amplitude correlation (IAC), and phase metrics, such as phase locking value (PLV), phase-based lag index (PLI) and weighted PLI (wPLI).

In a higher level of abstraction, there are functions that apply a processing pipeline to the input data to analyze certain features. MEDUSA© Kernel does not assume the nature of the input data in low-level functions, but most of the high-level analysis that are currently implemented are designed to work with EEG and MEG recordings. In short, high-level functions use the low-level methods to implement specific use-cases. These functions include signal processing algorithms for BCIs based on:

- P300 potentials: complete classification pipelines including regularized linear discriminant analysis (rLDA) [17], EEGNet [18] and EEG-Inception [19] that can be applied in offline and online experiments; P300 analysis and charts; and specialized data structures and functions for command decoding.
- Motor imagery: complete classification pipelines including CSP combined with rLDA [20], EEGNet [18], EEG-Inception [21] and EEGSym [21] that can be applied in offline and online modes; MI analysis charts; and specialized data structures for MI decoding.
- c-VEPs: offline and online circular-shifting reference pipeline based on CCA [10]; c-VEP analysis and charts; raster latencies correction; filter banks; and maximal length sequences (i.e., m-sequences) generation through linear feedback shift registers (LSFR) for binary and p-ary bases.
- Neurofeedback (NF): battery of high-level models based on spectral and connectivity metrics ready to be applied in online and offline apps [22].

This modular architecture, organized in levels of abstraction, guarantees independence between the different components of the library. Additionally, the package includes classes and functions to import data from other toolboxes (e.g., MATLAB, MNE), define the data format of signals and experiments, save recordings to several file types (e.g., bson, json, mat) and implement custom real-time signal processing pipelines. Furthermore, some of the functions, including the BCI models, can be applied in both online and offline experiments. Therefore, MEDUSA© Kernel can be used for offline analysis of previously recorded data, such as public databases, or in real-time tasks. In fact, MEDUSA© Platform relies on this package for signal processing. This is an interesting feature that allows reproducing the exact same results achieved in an online experiment during subsequent offline analyses, thus facilitating experimental reproducibility. Finally, it is worth mentioning that community contributions to this package are welcome. For more information, check the official documentation page at https://docs.medusabci.com/kernel. There, you will find the API reference, hands-on tutorials, and the contributor's guide.

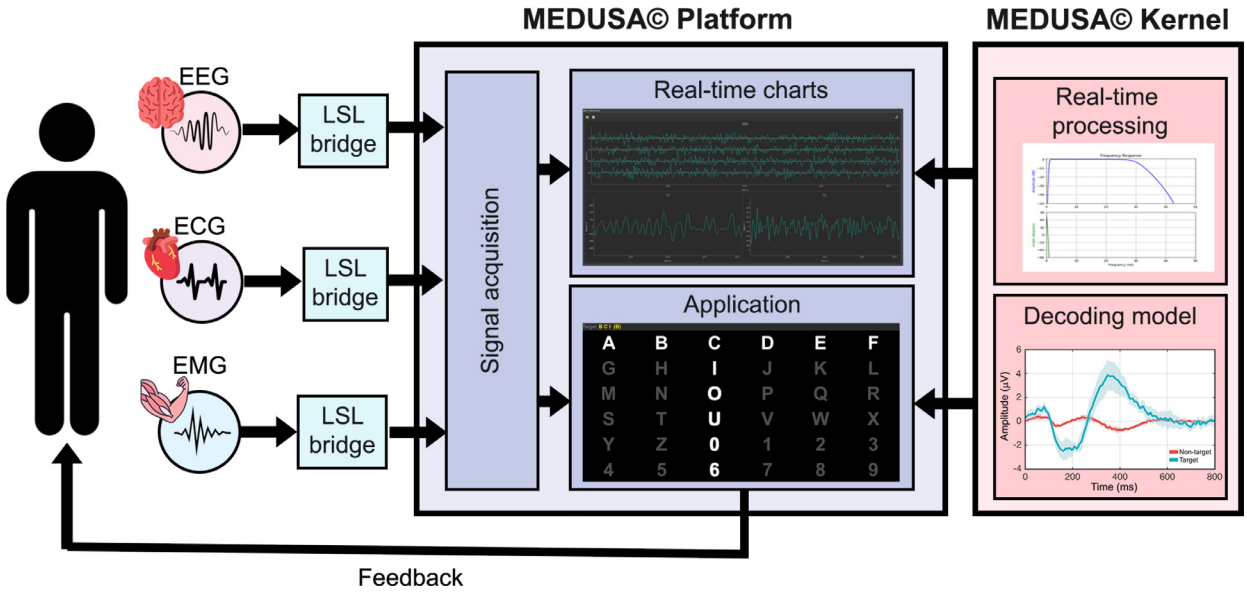

Fig. 1. Schematic overview of MEDUSA©. EEG: electroencephalography; ECG: electrocardiography; EMG: electromyography; LSL: lab-streaming protocol. An arbitrary number of input signals can be received in the platform through the LSL. These signals are available for real-time charts and apps, which implement open-loop or closed-loop BCI and neuroscience experiments. In this case, we represent the row-column paradigm (RCP) app. Functions from the Kernel may be used for signal processing in real-time charts and apps. In this example, the model detects the event-related potentials (ERP) in the EEG.

4. MEDUSA© platform

The architecture of MEDUSA© Platform follows a generic and adaptable structure that can be applied to most BCI and neuroscience experimental designs. This architecture can be divided in three main modules: signal acquisition, real-time charts, and apps. Each of these modules plays an essential role during the experimental process. The functionalities of these modules can be accessed from a modern GUI that provides a quick and intuitive control of the software. The Fig. 1 shows a schematic overview of MEDUSA©, including all the components that will be introduced in this section. Additionally, the Fig. 2 shows the main window of MEDUSA© Platform. The official documentation is hosted at https://docs.medusabci.com/platform. There, you will find hands-on tutorials, and the contributor's guide.

4.1. Signal acquisition

The signal acquisition module provides advanced functions for managing physiological measurement equipment in real time via LSL. This open-source protocol enables standardized time series recording by handling the networking, time-synchronization and real-time propagation of data. MEDUSA© Platform wraps LSL with high-level functionalities to offer more options than the original implementation, making our solution independent of the data recording hardware. This feature is especially important in research environments where a wide range of commercial and non-commercial devices are used to perform specific experiments.

In order to connect a device with MEDUSA© Platform, a LSL bridge (i.e., independent script or program) is needed to receive data using the device-specific application programming interface (API) and send it over LSL. Then, MEDUSA© Platform can be configured to receive the stream, selecting the type of signal and channel information. Once the stream has been configured, it will be available for real-time visualization and apps. Noteworthy, MEDUSA© Platform can manage several LSL streams at the same time, which is useful to synchronously record different signals or implement collaborative/competitive BCIs.

In recent years, LSL has been adopted by the research community as a *facto* standard to record biological signals, such as EEG, electrocardiography (ECG), or electromyography (EMG), in many research laboratories around the world. This open source protocol has a large community developing LSL bridges for most commercial EEG recording devices (e.g., g.tec, Brain Products, Neuroscan, etc.). Moreover, in those cases where LSL support is not yet available for a specific device, the user could write a LSL bridge in order to connect the hardware to LSL and stream the data. Check our tutorial on how to create LSL bridges, which is available in the documentation, for more information. These features make LSL a suitable technology to support the signal acquisition module of MEDUSA© Platform in order to take its functionalities one step ahead of other available solutions. For more information about LSL, visit the official website https://labstreaminglayer.org.

4.2. Real-time charts

MEDUSA© Platform implements several types of charts to allow the visualization of LSL streams in real time. The panel can be fully customized to fit different charts that can be adapted depending on each situation. This module is implemented using PyQtGraph (https://www.pyqtgraph.org), an open source Python package optimized for real-time representations with minimum resource consumption. Currently, both time and frequency representations are supported.

The temporal charts show the signal time courses in real time. Single and multi-channel charts are supported, with full control over the pre-processing of the signal before its representation, display time, decimation factor to reduce computational cost, scaling, and GUI characteristics (e.g., line width, colors, etc.).

The frequency charts estimate the power spectral density (PSD) using the Welch method. The lightweight implementation of this algorithm allows to transform the signal in real time, achieving a smooth temporal visualization of single or multi-channel signals. The chart also includes the possibility to configure several parameters, such as the spectral resolution, overlap or the segment length. These charts allow an alternative representation of the LSL streams

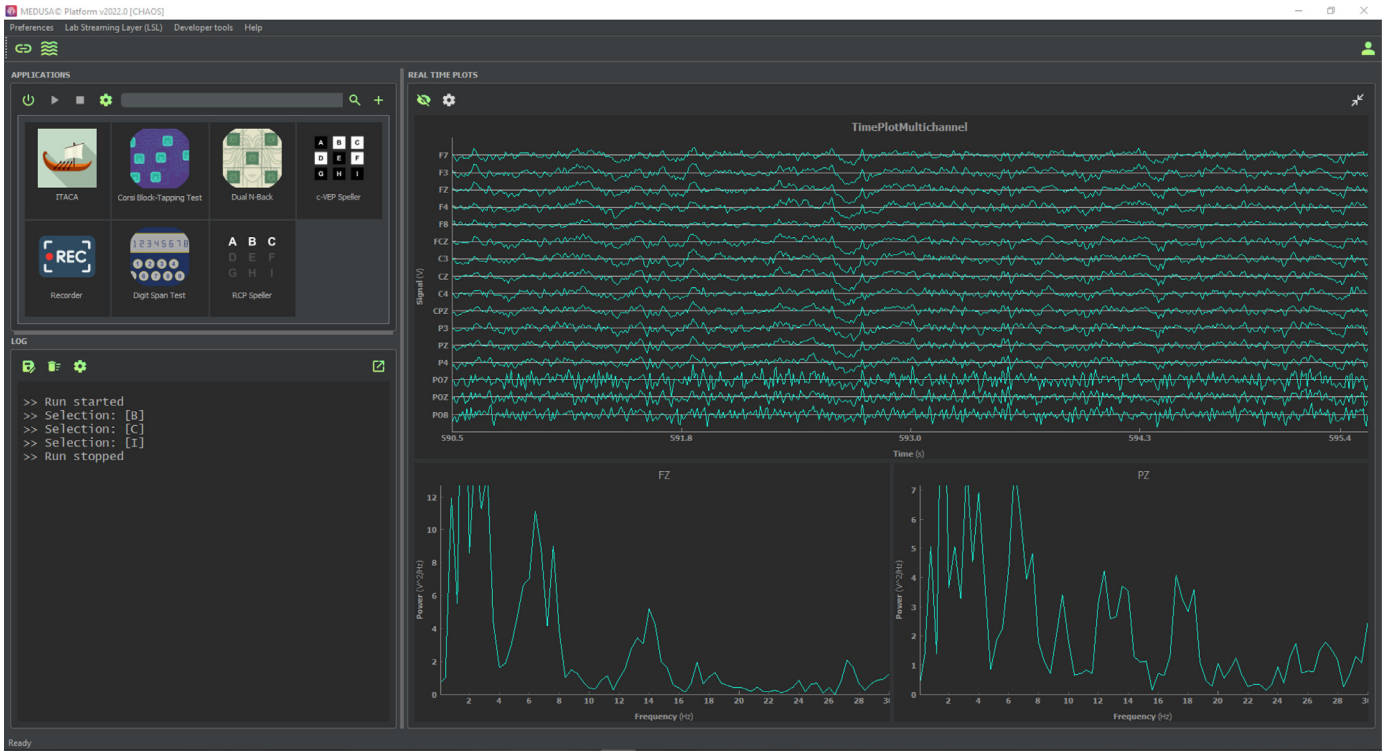

Fig. 2. Main window of MEDUSA© Platform. The view is divided in panels to control the different functionalities. These panels are: apps (up-left), log (bottom-left) and real-time plots (right). Additionally, more controls and configurations are available in the task bars.

to identify specific signal features, noisy components and sensor malfunctioning.

4.3. Apps

The apps are the central components of MEDUSA© Platform. These programs implement tasks or stimulation paradigms and provide real-time feedback while monitoring one or more signals. The architecture of the platform has been designed to provide independence between the signal acquisition and real-time visualization modules and the apps. Moreover, the software provides generic workflows that are applicable for most BCI and neuroscience experiments, which represents a powerful feature of MEDUSA©. This design increases the scalability of the solution, allowing to develop new apps on demand without modifying the base code of the platform.

Apps based on Qt (https://www.qt.io) and Unity (https://unity.com) are currently supported. In both cases, the workflow of the application is implemented in Python within MEDUSA©, but the stimulus and feedback presentation is performed by these frameworks. Qt is a widely used library, developed in C++ but with official bindings for Python, for the design and development of GUIs. Qt is powerful and simple, and comes with a large set of predefined widgets and functions, which makes it suitable for developing simple apps in a short time. However, this framework does not provide precise refresh rates and time synchronization, a key requirement in some experiments, e.g. BCIs based on steady-state visual evoked potentials (SSVEPs) or c-VEPs. On the other hand, Unity is a powerful graphic engine that implements advanced options for GUI control, 3D modeling and animations. Although the development of visual applications and games is more complex in this framework, it provides precise time control over refresh rate and stimulus presentation, making it suitable for advanced BCI applications. Of note, programming a Unity application implies coding in C#, its native programming language. The communication between Unity and MEDUSA© is performed through a multi-client asynchronous TCP/IP-based protocol.

Six ready-to-use apps are currently available for MEDUSA© Platform. However, the possibilities do not end here, because our software is especially designed to facilitate the implementation of custom experiments. To this end, Qt and Unity-based templates are provided in order to simplify the design and development of new apps, providing a generic workflow that should be applicable in most experiments. Additionally, we provide a complete guide, which can be found in the official documentation, that explains in detail how to design and develop apps for MEDUSA© Platform, giving real examples. Afterwards, these apps can be shared through the official app market at https://www.medusabci.com/market. In the following subsections, we present the apps that are currently available.

4.3.1. Recorder

This app is designed as a generic open-loop system that makes the recording of bio-signals easier. Custom conditions (e.g., eyes closed, eyes open) and events (e.g., movement, blink) can be defined. An interesting feature is the possibility of creating a recording plan with different conditions to maximize automatization. The time for each condition can be defined in advance, with sounds and messages being emitted when the time of each stage is up (e.g., 5 min eyes open, 5 min eyes closed, 2 min reading). Conditions and events can also be marked by pushing a button. Naturally, the timestamps and labels of the conditions and events are registered for offline analysis purposes.

4.3.2. RCP speller

This app implements the classical P300 speller based on the row-column paradigm (RCP) proposed by Farwell and Donchin [23]. The RCP displays a matrix of commands whose rows and columns are highlighted in a random order. For each selection (i.e.,

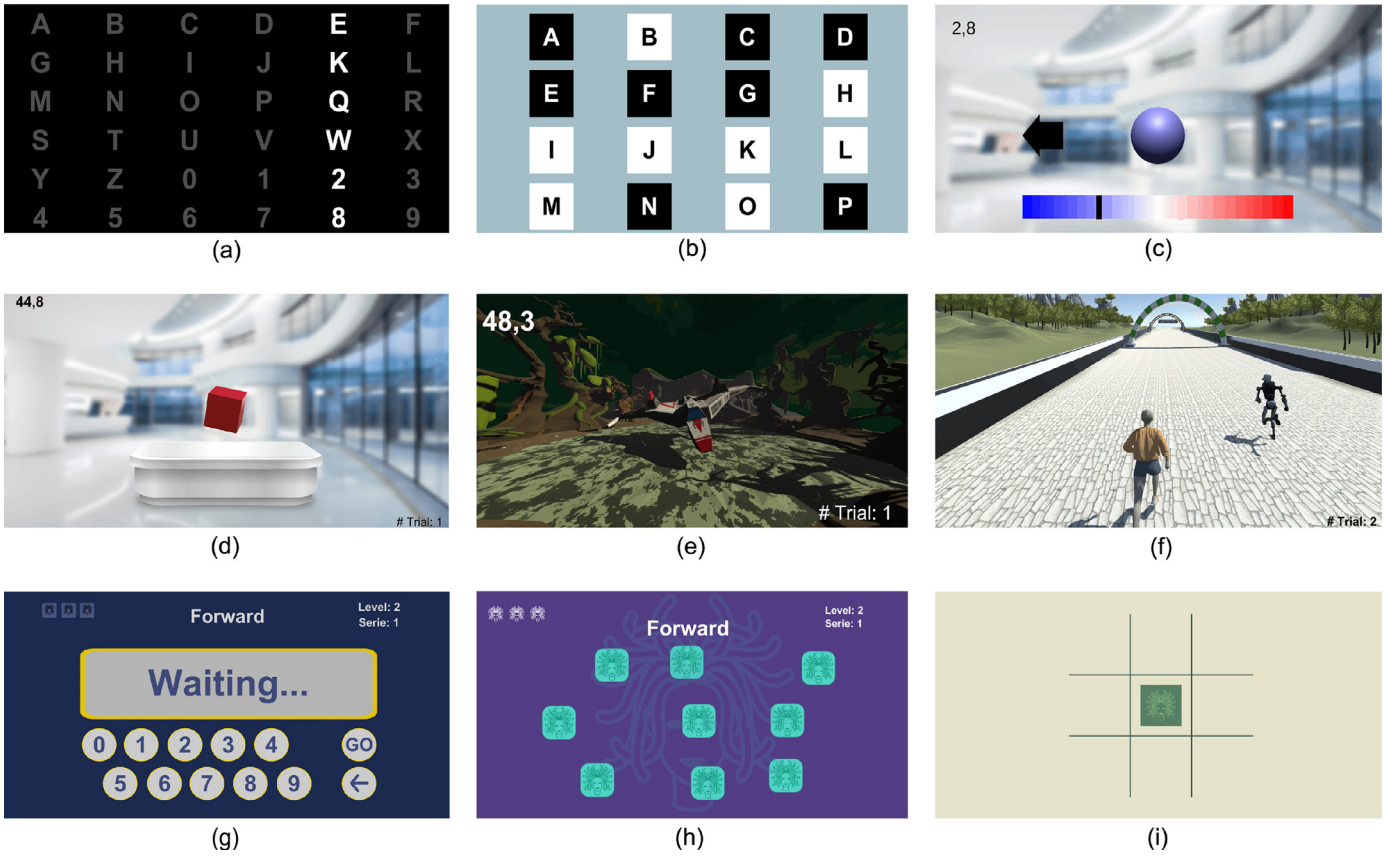

Fig. 3. Screenshots of some MEDUSA© apps. (a) row-column paradigm (RCP) speller, (b) code-modulated visual evoked potentials (c-VEP) speller, (c) Motor imagery paradigm, (d) Jedi cube scenario from the neurofeedback app, (e) Luke's spaceship scenario from the neurofeedback app, (f) Neurorunner scenario from the neurofeedback app, (g) Digit Span test from the Neuropsychological evaluation app (h) Corsi Block-Tapping test from the Neuropsychological evaluation app, (i) Dual *N*-Back test from the Neuropsychological evaluation app.

trial), the user has to stare at the target command while ignoring the other stimuli. When the trial ends, the application finds out the command by detecting the P300 elicited in the EEG of the user just after each target stimulus. The original purpose of the RCP speller was to improve the independence and quality of life of severely disabled people by providing a new channel of communication between the brain and the environment [3]. Despite the fact that this paradigm has been surpassed in recent years in terms of precision and selection speed by more advanced options, such as SSVEPs and c-VEPs, it is still widely used and it can be applied to investigate attention, visual information processing and cognitive responses within the brain [24,25].

The RCP speller app for MEDUSA© Platform provides advanced options. First, it allows to configure all the important parameters: stimulus duration, inter-stimulus interval, text or icon commands, flashing colors or command functions. Moreover, the available models for P300 detection include widely used options, such as rLDA, or advanced deep convolutional neural networks, such as EEGNet or EEG-Inception to improve performance [19]. It is also worth mentioning that it allows to configure the number of commands, including nested matrices to design complex menus for practical applications. The current implementation has been used in several studies to date, delivering state-of-the-art results [17,19,26,27]. The Fig. 3(a) shows a screenshot of this application.

4.3.3. c-VEP speller

This app provides a BCI speller based on c-VEPs under the circular-shifting paradigm [10]. The use of c-VEP as control signals is a recent but promising alternative to achieve reliable, high-speed BCIs for communication and control with very short calibration times [10]. Here, the commands are encoded using shifted versions of a pseudorandom sequence that exhibit low auto-correlation values for non-zero circular shifts. Under the assumption that the EEG response to the encoded stimuli shares some of the correlation properties of the original sequence, EEG responses elicited by looking at the shifted-versions of the sequence will allow users' intentions to be decoded through a correlation analysis [10].

To the best of our knowledge, MEDUSA© Platform is the only software that includes a c-VEP-based BCI speller nowadays. The application allows to customize a wide variety options, such as the number of commands, sequence rate (i.e., the rate at which the code is displayed), sequence length, color encoding and signal processing. Four different binary *m*-sequences with different lengths (31, 63, 127, and 255 bits) are available to allow users to include a variable number of commands. Then, encoding is automatically applied using circular-shifting seeking for maximal spacing between lags to benefit the subsequent decoding. By default, the application uses the "reference method" for circular-shifting as signal processing pipeline; i.e., epoch averaging and CCA to compute command templates and a correlation-based classifier for decoding (for more information, see Martínez-Cagigal et al. [10]). To improve accuracy for some users, the application also provides the ability to include a filter bank. Figure 3(b) shows a screenshot of this app.

4.3.4. Motor imagery

This app implements a classical two-class MI experiment. MI is detected by decoding the oscillations in the electric field in the sensorimotor cortex of the brain known as sensorimotor rhythms [3], and other components of movement-related cortical potentials [3]. The use of MI-based BCIs is of great interest due to its

great potential for rehabilitation by inducing neural plasticity in the brain [28].

The app shows a color changing sphere and a sliding bar, in which color and position are continuously updated depending on the detected class. Additionally, it allows to configure important parameters, such as preparation, trial or rest duration, or the classification model (CSP + rLDA [20] or EEGSym [21]). Typically, a previous calibration run of around 60 to 100 trials of each class would be needed to use the traditional CSP + rLDA classifier [20]. Alternatively, this app includes an initialized version of EEGSym trained with data from 280 subjects to start an experiment without calibration with an expected accuracy above 80% (for more information regarding the pre-training of the network, see Pérez-Velasco et al. [21]). The MI application also integrates a simple visualization tool of the event related desynchronization / synchronization associated with a MI event. Figure 3(c) shows a screenshot of this app.

4.3.5. Neurofeedback

This app includes the necessary tools to perform neurofeedback studies. Neurofeedback is a technique that presents real-time feedback to inform the user about specific features of their brain activity [5]. The aim is to facilitate self-regulation of brain activity using operant conditioning. Through neuro-plasticity mechanisms, this technology makes it possible to induce changes in brain activity that mediate cognitive functions, such as working memory [29], sustained attention [30], and episodic memory [31].

The implementation on MEDUSA© Platform has three different scenarios to facilitate the design of neurofeedback protocols with progressive difficulty. Each scenario has a different gamified design in order to keep the user motivated and engaged throughout the study [32]. These scenarios are: (1) a rising cube whose position and rotation speed depends on the user's brain activity; (2) a futuristic scene were the avatar has to rise his spaceship, again with the position controlled by the feedback metric; and (3) a foot race against an opponent, where the user can increase the speed of the avatar using the BCI. This application allows to easily configure all the important parameters of a neurofeedback study: the feedback rate, the brain activity to be trained (metrics based on band powers or connectivity between different regions), the channels used to provide feedback or the difficulty of the objectives to be achieved in each scenario. Figure 3(d)–(f) show screenshots of the three training scenarios contained in this app.

4.3.6. Neuropsychological evaluation tasks

This set of apps implement the computerized version of neuropsychological assessment tests widely used in the study of executive functions, such as attention or working memory. Specifically, MEDUSA© Platform includes the following: Dual *N*-Back [33], Stroop task [34], Digit Span test [35], Corsi Block-Tapping test [36] and Go/No-go task [37].

The implementation of these scenarios makes it possible to use them in conjunction with biosignal recording equipment. This enables analysis of changes in the user's brain activity in response to the cognitive demand generated during the use of the app. These experiments can be used both in cognitive assessment for neurofeedback and cognitive psychology studies. Figure 3(g)–(i) show screenshots of three of these test.

5. Discussion

There are important aspects that are worth discussing regarding the implementation and functionalities of MEDUSA©. In the following paragraphs, we analyze these aspects, providing at the same time an in-depth comparison with other general-purpose BCI platforms, such as BCI2000 and OpenVibe. For this comparison, we only considered software that includes all three stages of a BCI system: signal acquisition, signal processing, and feedback presentation. Therefore, the comparison leaves out signal processing toolboxes such as MNE, BioSig or Gumpy. The Table 1 provides a summarized comparison between the different BCI platforms available nowadays.

The first aspect to consider is the programming language, which has a direct impact in other characteristics as well. Among the previous BCI platforms, BCI2000, OpenVibe, xBCI and BF++ were developed in C++ and Pyff in Python [8,9,12–14,38]. As can be seen, most of them are based in C++, a general-purpose standard for its efficiency and high performance. Thanks to its low abstraction level, an experienced developer can optimize programs to an extent that is not possible with other programming languages. Nevertheless, C++ has a steep learning curve and the development is time consuming and complex, making it difficult to update and maintain the software. On the other hand, Python is a general-purpose language that reduces development costs thanks to its high abstraction level and community packages, flattening the learning curve and requiring less lines of code to implement the same function. Moreover, unlike in C++, there is a wide range of community packages (e.g., Numpy, Scipy, Scikit-learn, Tensorflow) that are specifically designed for signal processing, machine learning and deep learning, which simplifies the implementation of these functionalities. This increases the flexibility of Python-based software, making this language specially suitable for BCI research, where the possibility of including new features and testing hypothesis quickly is of great importance. The main disadvantage of Python is its reduced performance in comparison with other options, such as C++. However, the speed and memory requirements for most real-time applications can be met with careful implementations using concurrent processing to take advantage of modern multi-core processors. Moreover, critical parts of the software can be implemented in more efficient languages (e.g., C, C++ or C#) and have them called from Python. Until now, Pyff was the only BCI software implemented in this language, but the project has been discontinued for more than 7 years. This leaves MEDUSA© as the main alternative in this programming language.

Another critical aspect is the software maintenance. As can be seen in the Table 1, most of the projects are discontinued after their publication. Of the analyzed platforms, only OpenVibe is under active development to keep up with the latest BCI advances. For its part, BCI2000 receives critical updates, but the project does not seem to be adding new functionalities. The rest of the projects have been discontinued for several years or they are only used by the groups that developed them [3]. In this regard, it has to be taken into account that developing these platforms require advanced knowledge in software development and operations (DevOps) methodologies, signal processing, machine learning, graphical user interfaces or game engines. These special characteristics make BCI software updates and maintenance true challenges, especially considering the fast rate of developments in the field. As a result, the majority of the projects are abandoned few years, or even months, after their official release. Several strategies can be applied to mitigate this problem: (1) modular architecture to allow updates of individual components without disrupting the rest of the platform; (2) flexible implementation in high-level programming languages, such as Python, with detailed documentation and examples; (3) scalable design, with structures prepared to add experiments and functionalities on demand; and (4) provide specific tools to facilitate the community contributions. MEDUSA© has been developed under these principles, which simplifies the maintenance process to meet our long-term view of the project.

Regarding the available functionalities, the current implementation of MEDUSA© already includes one of the most complete suite of signal processing methods and BCI experiments among the re-

Table 1
Comparative between different platforms for BCI experiments.

Platform	Language	Last update	Analysis toolbox	Deep learning	Connectivity analysis	LSL support	Open app development	App market	Novel BCI paradigms	Modern GUI
xBCI [13]	C+	Dec 2008	✓							
Pyff [14]	Python	Feb 2016								✓
BF+ [12]	C+	May 2016	✓							
BCI2000 [9]	C+	Aug 2020	✓							
OpenVibe [8]	C+	Dec 2022	✓		✓	✓	✓			✓
MEDUSA©	Python	Dec 2022	✓	✓	✓	✓	✓	✓	✓	✓

Analysis toolbox: integrated signal processing funtions suitable for offline and online analysis; Open app development: specific tools and guides to develop custom experiments; App market: community space dedicated to share apps or experiments; Novel BCI paradigms: refers to most recent BCI paradigms, such as c-VEPs or connectivity-based neurofeedback.

viewed projects (see the Table 1). For instance, MEDUSA© is the only software that allows to use both deep learning and connectivity metrics for online and offline experiments. It also includes six different apps: recorder, RCP speller, c-VEP speller, motor imagery, neurofeedback and five neuropsychological evaluation tasks. Therefore, it provides a ready-to-use environment for investigating important aspects in these fields, such as the influence of stimulus and feedback characteristics in different BCI paradigms, offline and online comparison of classification approaches or the characterization of bio-signals (EEG, ECG, etc.) while the subject is performing tasks. Additionally, it is worth mentioning that, to the best of our knowledge, our implementation of the c-VEP speller is the only one publicly available at this time, allowing more investigators to use this BCI paradigm [10]. In comparison, among the other platforms, only OpenVibe offers a similar range of applications with different MI and neurofeedback experiments, a RCP speller and an SSVEP speller. Finally, it has to be noted that the viability of these functionalities has been tested in several research studies over the past few years, which provides evidence of the scientific validation of our solution [17,19,21,22,26,27,39].

Another highlight is that MEDUSA© provides specific tools to develop and share custom apps with little effort thanks to its modular, flexible and scalable implementation in Python. These tools include: (1) easy-to-use GUI functions to install and uninstall apps on demand; (2) templates and meta-classes with the necessary functions and communication protocols for an easy development of custom apps with Qt and Unity; (3) extensive documentation with specific tutorials to implement new experiments; and (4) the official app market official app market in our website to share apps with the community. Together, these features provide the right environment to grow a rich variety of community-developed apps for MEDUSA©. We expect that these tools, complemented by other interesting features (e.g., discussion forums, a modern website, etc.), will encourage the active participation of a multidisciplinary community, which is key to the success of the project. In comparison, none of the previous platforms provided specific support to create and share custom experiments with the community, as shown in the Table 1. In this regard, simplifying the implementation of BCI experiments and increasing their reproducibility could drive more developers and other professionals into this field, creating an suitable environment to promote the growth of neurotechnologies. MEDUSA© is the result of our efforts to address these limitations and bring together a united community to further advance this field.

Despite the advantages of MEDUSA©, our work is not without limitations, which will be addressed in the future. Like any new software under active development, there will be bugs to work out. In this regard, we ask our users to report issues and suggestions through the official communication channels: the forum (https://forum.medusabci.com) and the GitHub repositories (https://github.com/medusabci). MEDUSA© has been developed and tested on Windows environments and it does not support Linux or iOS operating systems for now. We plan to solve this limitation shortly to improve the coverage of our software. Additionally, the current implementation of MEDUSA© lacks some functionalities that some BCI researchers might miss. For instance, a wider variety of real-time charts (e.g., real-time spectrogram or M/EEG topographies), either based on PyQtGraph or Matplotlib, would be desirable. We also acknowledge the lack of some BCI applications, including tactile and auditory paradigms, an SSVEP speller or a multiclass MI app. However, we are committed to the maintenance and continuous development of MEDUSA© to make it the software of choice for BCI research. Thus, future versions of the software will include these and other functionalities, which will be made possible by the platform's design advantages. The website will also be completed with more documentation, tutorials and features to encourage the participation of the community.

6. Conclusion

In this paper, we presented MEDUSA©, a novel open-source software ecosystem for BCI and neuroscience experimentation. It includes a complete suite of signal processing methods for offline and online applications, such as spectral and non-linear metrics, connectivity analysis or deep-learning models for EEG processing. Moreover, different ready-to-use experiments are available, highlighting a c-VEP speller, a RCP speller, motor imagery, neurofeedback and five neuropsychological evaluation tasks. These features make MEDUSA© one of the most comprehensive BCI platforms nowadays. Additionally, we are committed to the maintenance and continuous development of the software, which is facilitated by its flexible, modular and scalable design. In this task, the community plays a key role. We made special efforts to promote the participation of the BCI community by providing specific tools to develop and share new methods and experiments. In our opinion, these features could make a difference for the progress of neurotechnologies in the coming years, especially if the community gets involved in the project.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research has been developed under the grants PID2020-115468RB-I00 and RTC2019-007350-1 funded by 'Ministerio de Ciencia e Innovación/Agencia Estatal de Investigación/10.13039/501100011033/' and European Regional Development Fund (ERDF) A way of making Europe; and by 'Centro de Investigación Biomédica en Red en Bioingeniería, Biomateriales y Nanomedicina (CIBER-BBN)' through 'Instituto de Salud Carlos III'

co-funded with ERDF funds. E. Santamaría-Vázquez, D. Marcos-Martínez and S. Pérez-Velasco were in a receipt of a PIF grant from the 'Consejería de Educación de la Junta de Castilla y León', and the European Social Fund; V. Rodríguez-González was in receipt of a PIF grant from the 'University of Valladolid'.

References

[1] A. Fornito, A. Zalesky, E.T. Bullmore, Fundamentals of Brain Network Analysis, Academic Press, 2016.
[2] C. Weiss, J.F. Disterhoft, Cognitive Neuroscience, in: Encyclopedia of Social Measurement, Cambridge University Press, 2004, pp. 341–349.
[3] J. Wolpaw, E.W. Wolpaw, Brain-Computer Interfaces: Principles and Practice, OUP USA, 2012.
[4] Y. Yu, Y. Liu, E. Yin, J. Jiang, Z. Zhou, D. Hu, An asynchronous hybrid spelling approach based on EEG-EOG signals for Chinese character input, IEEE Trans. Neural Syst. Rehabil. Eng. 27 (6) (2019) 1292–1302.
[5] R. Sitaram, T. Ros, L. Stoeckel, S. Haller, F. Scharnowski, J. Lewis-Peacock, N. Weiskopf, M.L. Blefari, M. Rana, E. Oblak, N. Birbaumer, J. Sulzer, Closed-loop brain training: the science of neurofeedback, Nat. Rev. Neurosci. 18 (2) (2017) 86–100.
[6] G. Lopes, P. Monteiro, New open-source tools: using bonsai for behavioral tracking and closed-loop experiments, Front. Behav. Neurosci. 15 (1) (2021) 1–9.
[7] D. Ciliberti, F. Kloosterman, Falcon: a highly flexible open-source software for closed-loop neuroscience, J. Neural Eng. 14 (4) (2017) 1–16.
[8] Y. Renard, F. Lotte, G. Gibert, M. Congedo, E. Maby, V. Delannoy, O. Bertrand, A. Lécuyer, OpenViBE : an open-source software platform to design, test, and use brain - Computer interfaces in real and virtual environments, Presence 19 (1) (2010) 35–53.
[9] G. Schalk, D.J. McFarland, T. Hinterberger, N. Birbaumer, J.R. Wolpaw, BCI2000: a general-purpose brain-computer interface (BCI) system, IEEE Trans. Biomed. Eng. 51 (6) (2004) 1034–1043.
[10] V. Martínez-Cagigal, J. Thielen, E. Santamaría-Vázquez, S. Pérez-Velasco, P. Desain, R. Hornero, Brain-computer interfaces based on code-modulated visual evoked potentials (c-VEP): a literature review, J. Neural Eng. 18 (6) (2021) 1–21.
[11] A. Craik, Y. He, J.L. Contreras-Vidal, Deep learning for electroencephalogram (EEG) classification tasks: a review, J. Neural Eng. 16 (3) (2019) 1–28.
[12] L. Bianchi, F. Babiloni, F. Cincotti, S. Salinari, M.G. Marcian, Introducing BF++: A C++ framework for cognitive bio-feedback systems design, Methods Inf. Med. 42 (01) (2003) 104–110.
[13] I.P. Susila, S. Kanoh, K.I. Miyamoto, T. Yoshinobu, xBCI: a generic platform for development of an online BCI system, IEEJ Trans. Electr. Electron. Eng. 5 (4) (2010) 467–473.
[14] B. Venthur, S. Scholler, J. Williamson, S. Dähne, M.S. Treder, M.T. Kramarek, K.-R. Müller, B. Blankertz, Pyff - a pythonic framework for feedback applications and stimulus presentation in neuroscience, Front. Neurosci. 4 (1) (2010) 1–17.
[15] C.R. Harris, K.J. Millman, S.J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N.J. Smith, R. Kern, M. Picus, S. Hoyer, M.H. van Kerkwijk, M. Brett, A. Haldane, J.F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, T.E. Oliphant, Array programming with NumPy, Nature 585 (7825) (2020) 357–362.
[16] P. Virtanen, R. Gommers, T. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Al, SciPy 1.0: fundamental algorithms for scientific computing in Python, Nat. Methods 17 (3) (2020) 261–272. Nature.Com
[17] E. Santamaría-Vázquez, V. Martínez-Cagigal, J. Gomez-Pilar, R. Hornero, Asynchronous control of ERP-based BCI spellers using steady-state visual evoked potentials elicited by peripheral stimuli, IEEE Trans. Neural Syst. Rehabil. Eng. 27 (9) (2019) 1883–1892.
[18] V.J. Lawhern, A.J. Solon, N.R. Waytowich, S.M. Gordon, C.P. Hung, B.J. Lance, EEGNet: A compact convolutional neural network for EEG-based brain-computer interfaces, J. Neural Eng. 15 (056013) (2018) 1–17.
[19] E. Santamaría-Vázquez, V. Martinez-Cagigal, F. Vaquerizo-Villar, R. Hornero, EEG-inception: a novel deep convolutional neural network for assistive ERP-based brain-computer interfaces, IEEE Trans. Neural Syst. Rehabil. Eng. 28 (12) (2020) 2773–2782.
[20] R. Fu, Y. Tian, T. Bao, Z. Meng, P. Shi, Improvement motor imagery EEG classification based on regularized linear discriminant analysis, J. Med. Syst. 43 (6) (2019) 1–13.
[21] S. Pérez-Velasco, E. Santamaría-Vázquez, V. Martínez-Cagigal, D. Marcos–Martínez, R. Hornero, EEGSym: overcoming inter-subject variability inMotor imagery based BCIs with deep learning, IEEE Trans. Neural Syst. Rehabil. Eng. 30 (2022) 1766–1775.
[22] D. Marcos-Martínez, V. Martínez-Cagigal, E. Santamaría-Vázquez, S. Pérez-Velasco, R. Hornero, Neurofeedback training based on motor imagery strategies increases EEG complexity in elderly population, Entropy 23 (12) (2021) 1–19.
[23] L.A. Farwell, E. Donchin, Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials, Electroencephalogr. Clin. Neurophysiol. 70 (6) (1988) 510–523.
[24] M. Arvaneh, I.H. Robertson, T.E. Ward, A P300-based brain-computer interface for improving attention, Front. Hum. Neurosci. 12 (January) (2019) 1–14.
[25] R. Zhong, M. Li, Q. Chen, J. Li, G. Li, W. Lin, The P300 event-related potential component and cognitive impairment in epilepsy: a systematic review and meta-analysis, Front. Neurol. 10 (AUG) (2019) 1–8.
[26] V. Martínez-Cagigal, E. Santamaría-Vázquez, R. Hornero, Asynchronous control of P300-based brain-computer interfaces using sample entropy, Entropy 21 (3) (2019) 1–14.
[27] E. Santamaría-Vázquez, V. Martínez-Cagigal, S. Pérez-Velasco, D. Marcos–Martínez, R. Hornero, Robust asynchronous control of ERP-based brain-computer interfaces using deep learning, Comput. Methods Programs Biomed. 215 (1) (2022) 1–10.
[28] T. Ono, K. Shindo, K. Kawashima, N. Ota, M. Ito, T. Ota, M. Mukaino, T. Fujiwara, A. Kimura, M. Liu, J. Ushiba, Brain-computer interface with somatosensory feedback improves functional recovery from severe hemiplegia due to chronic stroke, Front. Neuroeng. 7 (1) (2014) 1–8.
[29] S. Enriquez-Geppert, R.J. Huster, C. Figge, C.S. Herrmann, Self-regulation of frontal-midline theta facilitates memory updating and mental set shifting, Front. Behav. Neurosci. 8 (2014) 1–13.
[30] J.R. Wang, S. Hsieh, Neurofeedback training improves attention and working memory performance, Clin. Neurophysiol. 124 (12) (2013) 2406–2420.
[31] K.C.J. Eschmann, R. Bader, A. Mecklinger, Improving episodic memory: frontal-midline theta neurofeedback training increases source memory performance, NeuroImage 222 (117219) (2020) 1–11.
[32] A. Roc, L. Pillette, J. Mladenovic, C. Benaroch, B. N'Kaoua, C. Jeunet, F. Lotte, A review of user training methods in brain computer interfaces based on mental tasks, J. Neural Eng. 18 (1) (2021) 1–35.
[33] S.M. Jaeggi, M. Buschkuehl, W.J. Perrig, B. Meier, The concurrent validity of the *N*-back task as a working memory measure, Memory 18 (4) (2010) 394–412.
[34] C.M. MacLeod, Half a century of research on the stroop effect: an integrative review, Psychol. Bull. 109 (2) (1991) 163–203.
[35] A. Ockelford, The magical number two, plus or minus one: some limits on our capacity for processing musical information, Musicae Sci. 6 (2) (2002) 185–219.
[36] R.P. Kessels, M.J. Van Zandvoort, A. Postma, L.J. Kappelle, E.H. De Haan, The corsi block-tapping task: standardization and normative data, Appl. Neuropsychol. 7 (4) (2000) 252–258.
[37] F. Verbruggen, G.D. Logan, Automatic and controlled response inhibition: associative learning in the Go/No-go and stop-signal paradigms, J. Exp. Psychol. 137 (4) (2008) 649–672.
[38] C.A. Kothe, S. Makeig, BCILAB: a platform for brain-computer interface development, J. Neural Eng. 10 (5) (2013) 1–17.
[39] E. Santamaría-Vázquez, V. Martínez-Cagigal, D. Rodríguez, J. Finat, R. Hornero, Preventing cognitive decline in elderly population through neurofeedback training: a pilot study, in: International Conference on NeuroRehabilitation, Springer, 2020, pp. 407–411.