Adding Human Learning in Brain–Computer Interfaces (BCIs): Towards a Practical Control Modality

NATALIYA KOSMYNA, Grenoble INP
FRANCK TARPIN-BERNARD, Univ. Grenoble Alpes
BERTRAND RIVET, Grenoble INP

In this article, we introduce CLBCI (Co-Learning for Brain–Computer Interfaces), a BCI architecture based on co-learning in which users can give explicit feedback to the system rather than just receiving feedback. CLBCI is based on minimum distance classification with Independent Component Analysis (ICA) and allows for shorter training times compared to classical BCIs, as well as faster learning in users and a good performance progression. We further propose a new scheme for real-time two-dimensional visualization of classification outcomes using Wachspress coordinate interpolation. It allows us to represent classification outcomes for n classes in simple regular polygons. Our objective is to devise a BCI system that constitutes a practical interaction modality that can be deployed rapidly and used on a regular basis. We apply our system to an event-based control task in the form of a simple shooter game in which we evaluate the learning effect induced by our architecture compared to a classical approach. We also evaluate how much user feedback and our visualization method contribute to the performance of the system.

Categories and Subject Descriptors: I.5.5 [**Implementation**]: Interface System; H.5.2 [**User Interfaces**]: Input Devices and Strategies

General Terms: Co-learning, Brain–Computer Interfaces, Visualization

Additional Key Words and Phrases: Wachspress coordinates, interactive machine learning

ACM Reference Format:
Nataliya Kosmyna, Franck Tarpin-Bernard, and Bertrand Rivet. 2015. Adding human learning in brain–computer interfaces (BCIs): Towards a practical control modality. ACM Trans. Comput.-Hum. Interact. 22, 3, Article 12 (May 2015), 37 pages.
DOI: http://dx.doi.org/10.1145/2723162

1. INTRODUCTION

Brain–Computer Interfaces (BCIs) allow capturing the brain activity of users by processing and classifying their brain signals with the purpose of generating commands for any computer system. There are many applications of BCIs [van Erp et al. 2012], traditionally in medicine [Wolpaw et al. 2002] (interaction with locked-in patients, typing for quadriplegics) but also increasingly out-of-the lab applications such as controlling devices (e.g., car, robotic arm – Chapter 6.2 in Tan and Nijholt [2010]), user-state monitoring (e.g., workload [Pike et al. 2014]) for interface and task adaptation, as well as gaming [Nijholt et al. 2009a].

Authors' addresses: N. Kosmyna, LIG-IIHM, BP 53, Grenoble Cedex 9, France; email: natalie@kosmina.eu; F. Tarpin-Bernard, Scientific Brain Training, Bat. CEI, 66 Bd. Niels Bohr CS 52132 69603 Villeurbanne Cedex, France; email: f.tarpin@sbt.fr; B. Rivet, Gipsa-lab/Grenoble INP, 11 Rue des Mathématiques, 38400 Saint Martin d'Hères, France; email: bertrand.rivet@gipsa-lab.grenoble-inp.fr.

DOI: http://dx.doi.org/10.1145/2723162

Mastering the production of stable brain-signal patterns [van Erp et al. 2012] is a crucial component of all existing BCI systems. Learning this skill can be challenging, requiring hours of training and repetitive practice, especially for continuous control. Users must train to adapt their brain-signal patterns to make them easy to recognize by a computer [Lotte et al. 2013]. For the first BCI applications, the training technique used was Operant Conditioning (OC). OC produces good results; however, it requires extensive and painstaking training (up to several months): all the effort in learning to use the BCI is incumbent on the user. Subsequently, with the appearance of machine-learning techniques, it became common to train classifiers. This allowed reduction of the training time to hours (at most), instead of days or months, by shifting the focus from the user to the system. Instead of training the user to adapt, we train a classifier to adapt to each user. However, even then, users do not have an easy way of knowing whether they imagine the action in a consistent or correct manner so that the system can recognize their brain-signal patterns. One of the solutions to alleviate the severity of this issue is the use of neurofeedback [Gruzelier et al. 2006], which consists of displaying components or features of the user's brain signals in real time (e.g., the blue bar in Figure 2(a)). This allows users to get a sense of how to modulate their signals during the training of a classifier, which contributes to making the job of the classifier easier. Yet, even today, feedback is designed at a low level in an often unappealing form and is thus difficult for users to understand and interpret. Currently, the improvement of feedback in BCIs is an essential step towards improving BCI performance and usability [Lotte et al. 2013].

While efforts towards improving feedback are paramount, there is a limitation in that the feedback remains unidirectional. The system gives feedback to the user about the training or calibration, while the user is just a spectator. Lotte et al. [2013], basing themselves on a parallel with formative feedback, have argued that BCI feedback should also allow training of the user. The dynamic between a BCI and the user should be closer to that of a teacher and his or her student. Feedback must go both ways if the educational objective is to be met in the best possible manner. This is called BCI "co-learning" [Kosmyna and Tarpin-Bernard 2013]. However, there have been few approaches involving feedback from the user to the system. Most involve implicit feedback, in which feedback is captured without any conscious action from the user through the capture of physiological signals, for example. One common approach for implicit feedback is the detection of Error Potentials (ErrP) in the users' brain signals indicating that the user perceived an error [Blankertz and Schäfer 2002; Iturrate et al. 2010, 2012; Vi and Subramanian 2012]. Such feedback can be used to make adaptable BCI systems and classifiers in order to reduce the amount of errors.

In order to make BCIs more suitable for real-world applications, users must be allowed to interact *asynchronously (in "real time")*. Another limitation is that the standard experimental setting for BCIs is *synchronous*: the BCI can only be used during limited time periods at regular intervals. For example, the system displays a cue; 2 seconds later, the BCI is usable during 5 seconds and then 10 more seconds pass before the cue is shown again. Feedback is only displayed during this usable period in a continuous manner [Wolpaw et al. 2002]. In a synchronous setting, the performance of BCIs is easy to evaluate, making the setting desirable for experiments. For real-world use, however, an asynchronous system is more adequate: classification occurs continuously up to several times per second.

Given the constraints for co-learning BCIs imposed by asynchronous (*real-time*) classification, visualization, and feedback to and from users, we need to make the classification process interactive. To that end, we turn to the field of Interactive Machine Learning (IML), which involves a combination of classifier visualization techniques and interaction techniques to enable users or experts to guide and adapt the training process of a classifier. Our specific interest in IML pertains to the real-time interaction with supervised classifiers as characterized by Fiebrink et al. [2011], along with visualization techniques compatible with real-time classification and nonexpert friendly visual feedback.

Based on the BCI design guideline of Lotte et al. [2013], the ideas of co-learning BCIs [Kosmyna et al. 2013] as well as IML techniques, we propose a new architecture for co-learning BCI named CLBCI (Co-learning for Brain Computer Interfaces) coupled with WDV (Wachspress Distance Visualization), a simple visualization method for real-time feedback. WDV consists of projecting distance features and the class distribution of the classification outcome in a 2D regular polygon (Wachspress coordinate interpolation [Lidberg 2011] is a generalization of Barycentric coordinates).

We apply our system to an event-based control task in the form of a simple shooter game for which we evaluate the learning effect induced by our architecture compared to a baseline approach from the state of the art. We consider two conditions: the presence or absence of WDV visualization and the presence or absence of user feedback, studied over three sessions. This way, we can study the influence of each condition on user learning in the context of BCIs. We make the following hypotheses:

- (H1) The combination of CLBCI, user feedback, and the WDV visualization scheme will help users modulate their brain signals better and induce a learning effect greater than in a standard state-of-the-art BCI.
- (H2) Both visualization and user feedback are complementary: visualization alone is not sufficient.

In the remainder of the article, we first present relevant related work and background information. Thus, the system we propose lies at the intersection of three fields that needs to be covered before we can present the system. We divide them as follows:

- Section 2: General information about BCIs and related work on signal processing, classification, and existing BCI paradigms
- Section 3: General presentation of BCI feedback and related work on feedback techniques geared towards user learning
- Section 4: A review of related work in Interactive Machine Learning with a special focus on real-time visualization techniques and their limitations for the application to BCI co-learning

In light of the application and of the evaluation of our system, we briefly introduce BCIs in the context of HCI applications (Section 5), focusing on existing control applications and then more specifically games, which is our application area of choice. Subsequently, we present our CLBCI system and the WDV visualization method (Section 6). Section 7 contains the evaluation of our system and an analysis and discussion about the results. We provide a discussion of the limitations of our system in Section 8 and potential applications besides BCIs in Section 9. Concluding remarks and propositions for future work and ongoing projects are presented in Section 10.

2. BRAIN COMPUTER INTERFACES (BCI)

We now review and introduce the main BCI concepts that are necessary for the understanding of this article[1]. We will follow the steps of the “BCI loop” (Figure 1) that symbolizes the typical BCI experience.

[1]This section is not intended to be an exhaustive presentation of BCIs. For general matters on the processes associated with the BCI loop, we refer the reader to the state-of-the-art presented by Nicolas-Alonso and Gomez-Gil [2012].

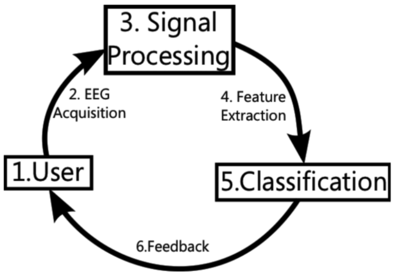


Fig. 1. The common BCI loop.

A BCI is a closed loop between a user and the system. Generally, the user interacts with the system and the system gives feedback about its decision state after the interaction. The BCI loop is composed of (in order): the signal acquisition process; the signal processing step, in which signals are processed and prepared; the feature extraction phase, in which we identify and extract salient features from the signal; and the classification stage, in which the features are matched to the classifier model to identify one of the phenomena it was trained to capture.

More formally, the objective of a BCI system B is to assign a brain signal S_t of a fixed duration (an *epoch;* e.g., 1s) at time t, to a class Cl_i from a set of N classes Cl that correspond to a set of brain activity states BS_i that need to be recognized.

A machine-learning classifier C is trained to recognize the desired states BS_i through a set of *training examples* $T(Cl_i)$ for each class Cl_i. A training example is a signal epoch of the same duration as S_t that was recorded when the user was in the desired state BS_i corresponding to the class Cl_i. This is called the training or calibration phase.

C returns, for each S_t, a classification outcome C_o. C_o is a map that associates to each C_i a confidence weight or posterior probability w_i. Note that not all classifiers give a distribution across outcomes; for example, the output of Linear Discriminant Analysis (LDA) is the label of the selected class. However, we can see this latter case as a particular case in which the probability associated with the selected class is 1 and the probability associated with the other classes is 0. The element of C_o with the highest weight is the most likely assignment for S_t.

The first modern BCI systems based on machine learning were trained and evaluated offline (with prerecorded signals) and were the first step towards the development of better feature extraction and signal processing techniques. Those developments then made possible the appearance of single-trial classification BCI systems that could be used online as well (i.e., used directly for a task). Among those systems, there is a distinction between *synchronous* and *asynchronous* systems.

Synchronous BCIs follow a two-phase protocol. First, signals from users are recorded while they perform a series of imagined actions (e.g., moving the left hand for motor imagery-based BCIs) or while they are subjected to stimuli (e.g., flickering targets or LEDs for SSVEP BCIs). Then, a classifier is trained to recognize the different imagined actions or stimuli. Finally, the BCI system can be used online. In *synchronous* BCIs, online use is not in continuous “real time” (in synchronous systems, feedback is continuous during fixed periods, whereas in asynchronous systems feedback can be continuous all the time). The system gives a cue to users before the stimulus is generated or before they must perform an imagined action. Then, signals are recorded for a fixed period of time (e.g., 5s) to produce a labeled training instance. The system then gives the classification output, after which there is a resting phase (e.g., lasting 10s) before the cycle starts over. In contrast, *asynchronous* BCIs capture and classify brain signals continuously to achieve a fully real-time system [Nooh et al. 2011].

We will now review relevant BCI paradigms. We will then review current signal processing and BCI classification techniques, and finish with current feedback practices for BCIs.

2.1. BCI Paradigms

When the set of brain states to recognize *BSi* are the result of a conscious action of the user, the BCI is called *active*. When the BS_i are the result of an external stimuli, the BCI is called *reactive*. When the BS_i are the result of a passive state of a user, the BCI is called *passive*. In this article, we deal only with active and reactive BCI systems. The most common active and reactive BCIs include: motor imagery, P300, and SSVEP [Nicolas-Alonso and Gomez-Gil 2012]. As for passive BCIs, they often rely on continuous activity in order to detect a general state of mind [Fairclough 2011]. The main BCI paradigms are the following:

- Motor Imagery (MI) – The user imagines moving various body parts, for example, hands, feet, tongue [Blankertz et al. 2008].
- Steady State Visually Evoked Potentials (SSVEP): The user looks at a target flickering at a certain frequency, for example, 15Hz. This causes a rapid succession of action potentials in the visual cortex, some of which are at the same frequency as the stimulation [Kelly et al. 2005].
- P300: P300 is a positive action potential generated 300ms after the user makes a choice (conscious or otherwise). By putting a set of items to choose from in a grid or matrix and by successively flashing the items, it is possible to trigger a P300 when the item the user wants to select flashes [Wolpaw et al. 2002].
- Error potential-based BCIs (ErrP) exploit error-related negativity (ERN) event-related potentials (ERP) that are negative activations generated in the brain 150 ms after the stimulation onset when the user commits an error (even if not consciously aware) or when negative feedback is received [Iturrate et al. 2010]. ERNs are often used to produce adaptive BCI systems that can detect when the user perceives a classification error and adapt the classifier accordingly [Thomas et al. 2013]. They also have important applications in interaction design, for which they can serve to detect desirable properties of interactive processes, without explicit feedback from users [Vi and Subramanian 2012; Vi et al. 2014].

In this work, we focus on SSVEP and MI in particular, presenting them in more detail.

The SSVEPs resulting from looking at a flickering target primarily occur in the occipital region of the visual cortex. The most common type of features used to extract SSVEP activity is Power Spectral Density (PSD) around the frequencies of the targets. PSD is typically computed as the squared modulus of the Fast Fourier Transform (FFT) of the signal. There have also been SSVEP systems using recursive outlier selection as well as template-based approaches (e.g., Minimum Distance Classification) [Amiri et al. 2013]. Typically, SSVEP and other VEP-based paradigms require little to no training. In the case of a machine-learning based system, only minimal training would be required (e.g., in a template-based approach) [Nicolas-Alonso and Gomez-Gil 2012]. For instance, in a standard synchronous setting using PSD features, SSVEP systems can achieve performances of around 83% [Ìscan et al. 2011] in classification accuracy. Asynchronous SSVEP systems in such works as Wei et al. [2013] can achieve anywhere between 67% (for 1s trials) and 91% (for 4s trials).

As for MI, any motor task and motor activity generates action potentials in the mu band (7–13Hz) and in the beta band (13–30Hz) over the motor cortex and the sensory motor cortex. However, actual motor activity is not necessary; merely imagining or mentally rehearsing a motor activity generates similar activations as real movement. This makes paradigms such as MI possible. MI involves two phenomena: an event-related synchronization (ERS) and an event-related desynchronization (ERD). ERDs manifest as a decrease in amplitude and ERSs as an increase in amplitude. For MI, the activation starts with an ERD (mu band) at 2.5s before the onset of motor activity and that reaches its maximal amplitude shortly after the motor activity occurs.

Simultaneously, there is an ERS in the gamma band (36–40Hz) followed by a second ERS over the beta band [Pfurtscheller and Lopes da Silva 1999]. The location of the activations depends on the limb involved in the motor activity or imagined motor activity, given that each limb is mapped to a different location of the sensory motor cortex. For example, right-hand motor activity manifests over the C3 electrode and left-hand motor activity manifest over the C4 electrode (see Figure 11), following the standard 10- to 20-electrode placement system. State-of-the-art MI synchronous systems typically use LDA classification and Common Spatial Pattern (CSP) filters [Lotte et al. 2007]. Such systems achieve high classification accuracies, for example, between 72% to 94% depending on the subjects in one of the first works using CSP [Müller-Gerking et al. 1999] for 2-class MI. More recent work, such as Wang et al. [2004], achieve average performances of 78% to 80% with a protocol involving less training data for a 2-class single-trial MI BCI. Other approaches, such as Hema et al. [2009], achieve performances of up to 96.15% using a recurrent neural network architecture.

As for asynchronous BCI systems based on MI, typical classification performance is around 70% to 76% over 1s or up to 94% over 4s [Nooh et al. 2011]. For example, Kus et al. [2012] obtain a performance of around 74.84%.

2.2. Signal Processing

There are active efforts in signal processing towards improving filtering algorithms to increase the signal-to-noise ratio and improve feature extraction algorithms [Nicolas-Alonso and Gomez-Gil 2012].

The improvement of filtering and feature selection algorithms, especially for asynchronous systems, has been mostly focused on exploiting spatial filtering techniques, for example, CSPs or ICA [Nicolas-Alonso and Gomez-Gil 2012]. ICA can be used for spatial filtering, artifact removal [Fatourechi et al. 2007], or electrode selection. Similarly, with the aim of separating signal from noise, there has been some interest in using distance measures based on differential geometry (e.g., Kullback-Leiber divergence, or Riemannian geodesic distances [Barachant et al. 2012]) that can help in extracting and selecting features by computing a projected distance in a higher-dimensional space. Both approaches show a lot of potential to varying degrees, depending on the specific task and subjects.

2.3. Classification

One criterion for distinguishing BCI systems relates to the classifier they use. *Supervised* BCI systems use supervised classifiers and require to be trained with labeled training data. *Semi-supervised* BCI systems, on the other hand, require only a few labeled training data items for calibration and use adaptation techniques to compensate for the scarceness of training data. Using semi-supervised systems can lead to increased performance as they exploit the additional information potentially provided by extra unlabeled signals. In the context of asynchronous BCIs, semi-supervision is particularly relevant as it can compensate for the lack of training data and the difficulty in precisely locating the activity of interest. Many classifiers have been applied to BCI systems. However, for synchronous systems, the de-facto standards are LDA and Support Vector Machines (SVM). They offer consistent performance and have few parameters to select[2].

[2]Refer to Lotte et al. [2007] for a review of BCI classifiers.

Our own work is based on Minimum Distance Classifiers (MDCs). MDCs are simple classifiers stemming from the pattern analysis community and used very early in BCI research. Contrary to SVMs and LDA, which need a fair amount of training examples to work properly, MDCs can be used in the context of single-trial classification more readily [Barachant et al. 2012]. Averaging signals leads to better classification accuracy, as it smooths noise away; however, it requires several signal captures (trials). To obtain a real-time system, the classifier needs to be able to work with a single trial. However, such classifiers have the disadvantage of being sensitive to noise and to the data used for their initialization [Barachant et al. 2012]. Similarly, the distance measure used is critical and needs to be appropriately selected. MDCs have been applied to asynchronous MI and P300, notably using the Riemannian Geodesic Distance [Barachant et al. 2012]. Their performance with a Riemannian distance was evaluated for asynchronous 4-class MI and compared to the standard CSP + LDA system. In the same conditions, MDCs averaged at 79.71% compared to 80.45% for CSP + LDA. The difference in means is not significant. Although the performance is on par with the standard system, their implementation and functioning is simpler and requires less computational overhead.

In a single trial context, such as when implementing asynchronous systems, it is important to have a form of adaptation to keep references consistent with the current state of the user. One approach is to include a form of feedback for classifier adaptation. This feedback can be given implicitly by using ErrPs or explicitly (by a direct or proxy intervention of the user). Regardless of the type of feedback, it should create minimal interference for the BCI.

An alternative currently being researched by Congedo et al. [2011] is the use of signal databases to offer a strong subject-independent initialization that does not require any calibration or training. The construction of such databases is costly and involving; however, once they are built they can easily be reused by any system.

Once the classification is performed, the results need to be shown to the user in a way dependent on the BCI paradigm used.

3. FEEDBACK

In BCI systems, feedback (from the system to the user) is considered as a fundamental component of the system [Lécuyer et al. 2008] that helps users to modulate their brain activity but also to display the result of the classification. The current design of feedback for BCIs remains limited and rarely considers user satisfaction as a primary concern. As suggested by Lotte et al. [2013], there is a need for a shift in current feedback designs to get closer to practices in instructional design, in which maximizing the learning potential is key. The situation is starting to change, however, as works are appearing that use more varied or multimodal feedback. For example, Vidaurre et al. [2010] and Barbero and Grosse-Wentrup [2010] achieved significantly better BCI performance by providing positively biased feedback to novice users following principles from instructional design.

We will first present a more detailed account of system feedback approaches, then go into possible solutions for user feedback.

3.1. System Feedback

System feedback is mostly concerned with showing the BCI user what the results of the recognition and classification of the brain signals are. However, the visualization of the classification output mostly remains an ad-hoc process. There are usually few classes to choose from and the outcomes are often visualized through the resulting in-task action triggered by the BCI (e.g., when interacting with a virtual reality environment or controlling a robot). However, before real-time use, a training phase is required: a supervised training session, operant conditioning and/or a familiarization session to let the user get acquainted with the nature of BCI control. The training phase needs to be task independent and yet ground the user in the reality of the task. We can see that there is clearly a dichotomy when it comes to system feedback: on the one hand, the *task feedback,* and on the other hand, the *training feedback*.

Lotte et al. [2013] summarize that BCI training feedback is often only evaluative (quantitative measure of how good the performance is) and corrective (whether the task was performed well or not). This does not match findings in instructional design and computerized formative feedback [Shute 2008]. More specifically, a common type of feedback, notably for imagery tasks, is to provide the distance from the separating plane[3]. It is a numerical value that can then be represented by a slider or in other unappealing ways. One of the justifications for such a feedback representation is that there is an overlap with neurofeedback, in which some representations of the signals (features, frequency components) are displayed to the user in real time to help them modulate their brain signals (for a visual example, see Figure 2). However, such representations imply that users must have some knowledge about the various properties of brain signals and in general how the brain and action potentials function. Acquiring such knowledge is not necessarily easy and is certainly not the most desirable requirement for the usability of the system. Furthermore, the semantics of the training phase are different from the semantics of the task itself.

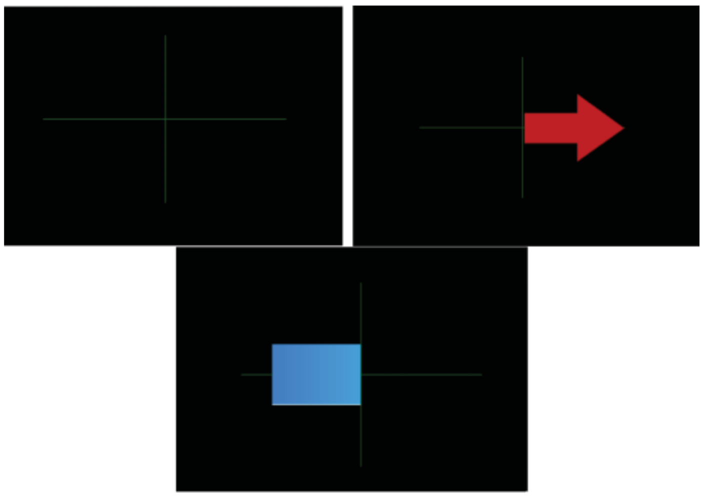
In order to alleviate current limitations, one of the directions is to build adaptive BCI systems that exploit some form of user feedback.

3.2. User Feedback

Aside from system feedback, we have highlighted the importance of incorporating user feedback as a way of further adapting the BCI system to the user and inducing learning. The feedback from the user to the system can either be *implicit* or *explicit*, although the latter has seldom been explored directly. Implicit feedback exploits physiological signals or activity from users to detect errors automatically (e.g., through ERNs). Explicit feedback is somewhat related to the notion of incremental training for BCI classifiers. In a traditional supervised training protocol, signals are first acquired offline and used to train the classifier before users can use it online. In incremental training, the classifier can be used online directly, and training examples can be added while the system is being used, which means that there could be a very short training session. Additional training examples can be captured through feedback from the user, for example. Incremental training has been shown to improve BCI performance. Millán and Mouriño [2003] demonstrated the potential of incremental training using a simple classification technique (Elmann Neural Network). They showed that the resulting classifiers after offline training and online incremental training are statistically similar. Llera et al. [2012] have proposed a theoretical model based on the idea of a “reinforcement signal” that incorporates both the notions of explicit and implicit feedback in a common framework.

Exploiting direct feedback leads to the idea of co-learning, in which users give and receive feedback to/from the system. Thus, the training and learning rate of both the system and the user will mutually increase.

[3]Each average epoch, after processing becomes a feature vector considered as a problem instance that is represented by a point in a (n-dimensional) problem space (each element of the feature vector is a coordinate of the point in one dimension). A linear classifier will learn the equation of a hyper plane (n-dimensional) that separates the instances that belong to one class from the instances of another class. A distance will be a value that tells us how far the current instance is from the separating hyperplane in the problem space.



(a) Classification feedback: an arrow pointing right indicates to the learner to imagine a right-hand movement. The blue bar gives feedback with its direction and length based on the classifier output. Thus it shows how well the mental task is recognized. The bar extends towards the left for an imagined left-hand movement, and toward the right for the imagined right hand movement. (http://OpenViBE.inria.fr).

(b) Brain signals, in which only relevant activity has been preserved.

Fig. 2. Examples of feedback in traditional BCI training protocols (usually for expert users).

4. INTERACTIVE MACHINE LEARNING

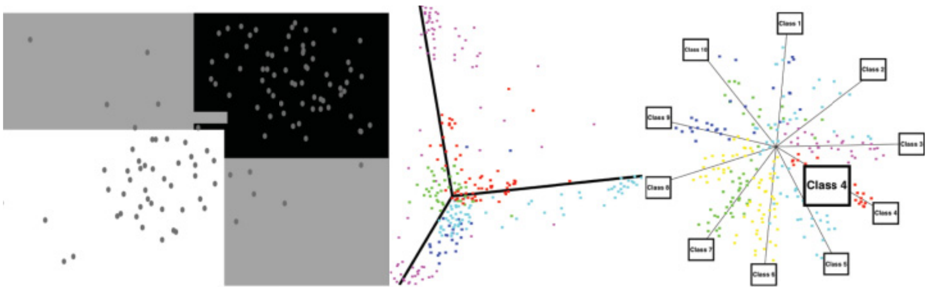
The notion of co-learning in BCIs can be seen as closely related to the notion of IML. The purpose of IML is to perform the interactive adaptation and steering of classifier training in general. This requires a feedback loop between the user and the system: the system gives feedback to the user (classifier visualization of outcomes or features) and the user then gives feedback to the system (new parameters, selection of features or training instances for the classifier) [Fails and Olsen Jr. 2003]. This matches the requirements and the definition of the idea of co-learning BCI by Kosmyna et al. [2013].

There is also a notion of co-training in IML [Zhu et al. 2011; Nigam and Ghani 2000]; however, it corresponds to the co-training of the classifier by two or more users concurrently and is different from what we call co-learning for BCIs.

As far as co-learning BCIs are concerned, we are interested in the particular case of real-time IML with supervised classifiers. We thus review related work about real-time supervised IML in more detail.

4.1. Real-Time Supervised IML

Starting from the work by Fails and Olsen Jr. [2003], who introduced the term of Interactive Machine Learning, and from the work of Talbot et al. [2009] on EnsembleMatrix,



(a) Frank & Hall's visualization (b) MDS visualization (c) Radial class visualization

Fig. 3. Classifier visualization techniques.

one of the first works applying IML to an HCI application, Fiebrink et al. [2011] outline the main-use cases for the application of IML to real-time supervised learning:

- Interactively editing training data
- Supporting the use by nonexperts
- Correspondence to *real user priorities*

In their work, they implement a system that extends the Weka visualization and classifier parameterization interface by allowing users to set parameters, make test runs, and visualize the results in a way that makes it easy to tune classification parameters and to select features.

Our work fits in the framework of their guidelines. For example, the guarantee of a fast training time is an important real-user priority. Similarly, given the real-time nature and potential noise issues, as well as the high probability that certain training trials are artifacts, *a real-user priority* is the ability to interactively edit training trials. The presence of an automatic margin also directly answers the priority of minimizing and smoothing noisy classification outcomes in BCIs.

The interaction of the user with the classifier can be done through GUI controls, as in work by Kapoor et al. [2010], in which an interface is proposed to set classification weights by directly modifying the confusion matrix. However, the classifier visualization aspect is more critical and must be considered carefully, especially with the added constraint of real-time visualization and of allowing the use by nonexperts.

4.2. Classifier Visualization

We are mostly focused on visualizing classification outcomes and classifier-independent features, to allow a consistent and general real-time classification. We will mainly focus on those aspects by assumption that the classifier is a black box that returns a distribution probability over classes. In other words, we are interested in general techniques that work for any classifier rather than specific classifier types. One of the main challenges is to find a way of projecting instance points and decision boundaries meaningfully in a low-dimensional representation [Alsallakh et al. 2014]. For example, Frank and Hall [2003] propose a general method by discretizing the attributes in the feature space, which allows one to obtain a separation of feature space in disjoint rectangular regions (Figure 3(a)). Then, the class probability distribution is estimated for each region in feature space. Colors are assigned to each class and the color of each region is computed as the interpolation of class colors weighted by region-specific class probability distributions. In this way, they obtain a visualization method for both class probabilities and decision boundaries. However, they restrict themselves to the case of two-class problems. When more classes are present, other issues arise.

For decision boundary and feature visualization, if the number of classes or the dimensionality of the feature space is high, it becomes necessary to either perform some form of scaling that preserves the intrinsic distance embedded in the feature space or select two or three features and project instance points and decision boundaries only on the corresponding dimensions. The latter assumes that the two features in question are not correlated with the other features (rarely the case), which means that distance ratios (between instances and decision boundaries) from the original space are not preserved. Migut et al. [2013] propose using the histogram of the distances from instance points to the classification boundary and separating the points in 4 categories: true negatives, true positives, false positives, and false negatives. The decision boundary is affixed to the origin of the plane. Positives are on the positive side of the y axis and negatives are on the negative side of the y axis. On the x axis, the positive side is true and the negative side is false. This way, they manage to make the multidimensional decision boundary clearly visible despite the fact that relative distances are not preserved.

The other approach for the visualization of feature spaces and classification outputs is to project the whole space in 2D while maintaining the original distances as much as possible. This category includes dimensionality reduction techniques such as Principal Component Analysis (PCA) and Multidimensional Scaling (MDS), which are linear embedding methods. PCA embeds objects in the 2D space by maximizing variance, whereas MDS tries to preserve pairwise distances. Nonlinear embedding techniques such as ISOMAP, local linear embeddings, Stochastic Neighborhood Embeddings (SNE), and connectivity preserving embeddings have also been applied, although they are much more computationally expensive. Other methods such as Fisher Linear Discriminant Analysis and Kernel Discriminant Analysis (nonlinear extension of the former) integrate class information by trying to maximize between class variance and minimize within class variance [Iwata et al. 2005] (Figure 3(b)).

One common point with all these techniques is that they are adapted for people who have a good understanding of machine learning. They do not have the goal of providing an intuitive visualization that is easy to interpret by nonexperts. Seifert and Lex [2009] propose a simple method, in which for each class a colored square (or image) is placed on a circle (Figure 3(c)). The squares or images for the classes are distributed equidistantly on the circle. This technique allows projection of outputs of classifiers (that return a probability distribution over classes) in the circle, using radial (or polar) coordinates. The color of each instance in the circle corresponds to that of the color of the class that instance is closest to. The main limitation of this technique is that it does not preserve distances when there are more than 2 classes, making it badly suited for visualizing features.

This last technique is actually closest to the work done in this article, as our main motivation is to provide an intuitive and simple visualization interface that does not strictly require that distances are preserved. However, in our work we do not perform radial projection in a circle, but rather a Wachspress coordinate projection in a polygon. Similarly, our premises and requirements differ in the fact that in radial class visualization the aim is to display a large set of instances to have an overall picture of a classifier, whereas the focus of our work is to provide real-time feedback about the classification. Moreover, we want to display the constituent features of the current classification results. As a consequence, we display a fixed decision boundary that starts at the bisecting point between edges and ends at the intersection of the bisecting line, somewhat like Migut et al. [2013], who use a fixed boundary and adapt instance position in accordance. This allows us to represent the current instance as a point and to make it clear what the current selected class is. Another contribution we make is the addition of a dynamic margin around the decision boundary that allows reduction of issues of noisy classification (very common in real-time classification).

The technique used in our work to compute the projection in a polygon is actually the reverse process of what Talbot et al. [2009] do in their work to allow users to select weights for the combination of a set of classifiers. Each classifier is at the edge of a polygon (with as many edges as classifiers) and users can move a point with their mouse in the Wachspress coordinate space of the polygon to select the weights. This technique for weight selection is not a visualization technique (their visualization is based on confusion matrices) in itself and falls into the other aspect of IML, that is, users interacting with classifiers directly.

5. BCI FOR HUMAN COMPUTER INTERACTION

Now that we have presented and reviewed BCIs in general, let us turn our focus specifically to HCI applications of BCIs. BCIs have mostly been successful in control applications and user state monitoring applications. A complete review of application areas can be found in Chapter 6 of [Tan and Nijholt 2010].

5.1. BCIs for HCI Applications

More generally, BCIs have the potential to play an important role in context-aware and adaptive human–computer interfaces (HCIs). In particular, passive BCIs can measure user states as diverse as arousal [Chanel et al. 2006], fatigue [Cajochen et al. 1996], vigilance [Schmidt et al. 2007], mental workload level [Grimes et al. 2008], and multimodal attention focus [Kelly et al. 2005]. This information can then serve to adapt HCIs, as is exhibited in the examples cited earlier, among others.

Another area in which passive BCIs can be of use is the evaluation of HCIs, for example, through workload estimation [Pike et al. 2014; Afergan et al. 2014; Iturrate et al. 2012].

Last, an area of HCI in which active and reactive BCI architectures are customarily used for control tasks is as a control modality in multimodal interaction settings [Gürkök and Nijholt 2012].

5.2. Control Applications

The first area of application of BCIs in HCI were speller modalities [Wolpaw et al. 2002] in the medical context and more recently as direct control modalities for HCI systems, but also as modalities in a multimodal interactive setting (e.g., with eye-tracking [Kosmyna and Tarpin-Bernard 2013]). There are many practical applications:

- Video games (see Section 5.3)
- Robotics and prosthetics [Hochberg et al. 2006]
- Virtual reality applications [Lécuyer et al. 2008; Guger et al. 2009]
- Communication (e.g., spell checkers) [Wolpaw et al. 2002; Noorzadeh et al. 2014; Yin et al. 2013]

There are two types of possible control paradigms:

- Event-based control, in which the BCI is used to trigger discrete events in an interface
- Continuous control, in which the BCI is used to directly control the movement of an object or element of the interface (e.g., the movements of a pointer)

5.3. BCIs and Gaming

Nijholt et al. [2009b] point out that in numerous experiments that use simple games, the rich and diverse game environment, rather than distracting users, leads to better attention and performance, for example, in the case of navigating in virtual reality

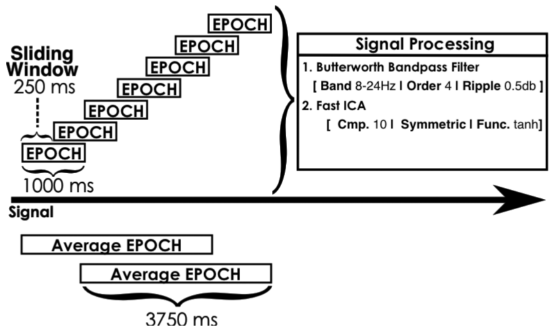

Sliding Window 250 ms
EPOCH
1000 ms
Signal
Signal Processing
1. Butterworth Bandpass Filter
[Band 8-24Hz | Order 4 | Ripple 0.5db]
2. Fast ICA
[Cmp. 10 | Symmetric | Func. tanh]
Average EPOCH
Average EPOCH
3750 ms

Fig. 4. Signal epoch segmentation and averaging in our system.

[Scherer et al. 2008; Royer et al. 2010; Lécuyer et al. 2008]. Other applications include the use of BCIs for the control of first-person shooters or RPG games (e.g., World of Warcraft [Bos et al. 2010; Kapeller et al. 2012]). BCIs are also used for controlling serious games [Sung et al. 2012; Wang et al. 2010], that is, games that have a purpose beyond playing, for example, to evaluate some phenomenon or to cure ailments (e.g., attention-deficient hyperactivity disorder (ADHD)).

Indeed, games have been shown to improve attention memory and executive control [Boot et al. 2008]. Moreover, games are an increasingly popular medium for learning in humans [Gee 2003] and can help students focus and engage more with subject matter. Thus games are an effective application context that allow alleviation of some of the limitations of BCIs, namely, by increasingly shifting the focus towards users and user learning.

In this work, we build a game directly around the experiment with the explicit purpose of making the experiment a more pleasant and meaningful experience for users as is done in Kosmyna and Tarpin-Bernard [2013].

6. THE CLBCI ARCHITECTURE

In our introductory motivations, on the one hand, we wanted to propose a BCI architecture that requires minimal training and that could be rapidly operational. On the other hand, we wanted the system to learn from the user in a more collaborative and engaging way. We put two-way feedback, both system to user and user to system, at the center of the equation to good BCI performance. This interactive learning is performed through direct user feedback and/or the intervention of the operator. In this particular work, an operator is giving the feedback for the user. The operator can parameterize the system by observing the performance of the user and letting the system know when its classification output is incorrect.

The architecture is based on MDC and advanced filtering techniques such as ICA. We roughly follow the steps of the BCI loop for the description of our system. First, we present the signal processing and acquisition aspects, then the distance measures and principles behind our MDC approach. Then we describe our visualization paradigm based on Wachspress coordinate projection. Finally, we discuss our implementation and interface that allow an interactive BCI co-learning experience.

6.1. Signal Processing and Acquisition

Figure 4 gives a graphic representation of how signals are divided into 1s epochs that each overlap over 250ms, then filtered, and averaged. Subsequently, signal processing is applied on each epoch:

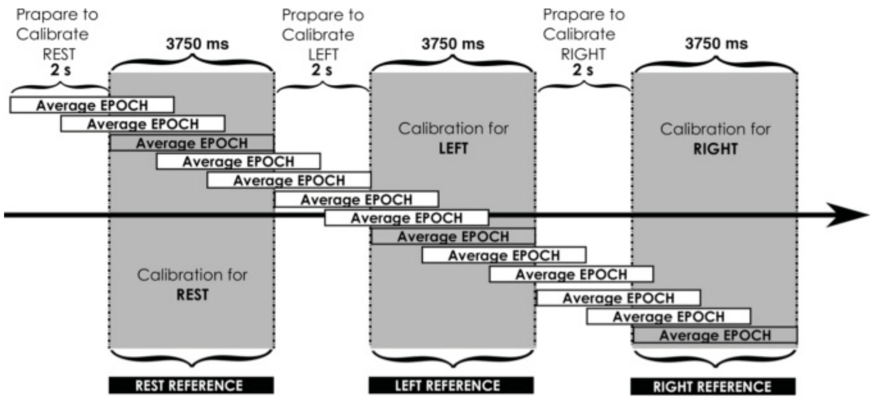


Fig. 5. Detailed view of the calibration phase.

- The Butterworth filter allows for selection of the appropriate frequencies for the modality and discarding of unwanted frequencies. The frequencies filtered by the Butterworth filter are 8Hz to 24Hz on pre-epoch signals for MI. For SSVEP, we filter between 14.5Hz and 15.5Hz for the first target and between 19.5Hz and 20.5Hz for the second target.
- The Fast ICA [Hyvärinen and Oja 1997] algorithm projects the signal data in a space in which data points are maximally independent, essentially separating task-related sources from noise sources and other interference. The difference with other ICA algorithms is that Fast ICA uses a fast algorithm based on fixed-point calculations. We applied the variant of Fast ICA with symmetric orthogonalization and using a hyperbolic tangent contrast function. We computed 10 components (each iteration produces one component; we performed 10 iterations). We used a GPL Java implementation of the original algorithm, as described by Hyvärinen and Oja [1997].
- Then, we produce average epochs that allow us to smooth the signal and remove some of the variability. Each average epoch is formed from the average of overlapping within-epoch signals of 5 consecutive epochs. If the sliding window is 250ms, then the average epoch formed from 5 epochs will be 3,750ms long.

The system thus produces an average epoch 4 times per second, which is then used for feature extraction and for classification. Thus, the classifier will yield 1 classification per second. Given that ICA is rather costly to compute, anything less than 1s led to subreal-time performance on the machine on which we performed the processing (2012 MacBook Air, i7@2.9GHz).

Given the sensitivity to noise and to variability in minimum distance–based classification, ICA or other similar processes separate noise source from authentic signals and make it easier for the distance measure to accurately capture relevant differences in brain-signal patterns. Our classifier requires only minimal training data to start functioning, as our aim was to reduce that training time to a single calibration trial per class. This allowed us to capture a reference signal for each of the classes. In the case of our system, we captured full averaged epochs as references. Thus, the calibration for each class lasted 3750ms and we left 2s of rest between each calibration trial so that users could unwind briefly and prepare for the next trial. Figure 5 graphically shows the process by which we capture the reference signal for a particular BCI task. Note that a single average epoch is used as class reference; this is why the calibration phase is synchronized with the beginning of an average epoch.

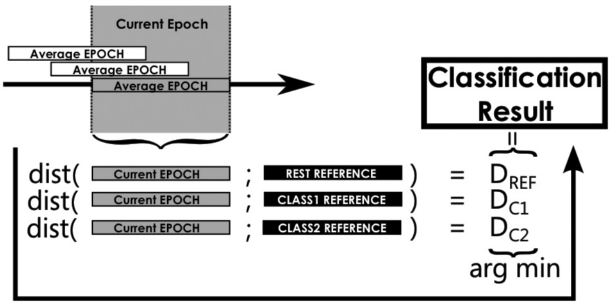

Current Epoch
Average EPOCH
Average EPOCH
Average EPOCH
Classification Result
dist(Current EPOCH ; REST REFERENCE) = D_REF
dist(Current EPOCH ; CLASS1 REFERENCE) = D_C1
dist(Current EPOCH ; CLASS2 REFERENCE) = D_C2
arg min

Fig. 6. Our classifier with three classes and one reference epoch per class, using a multinomial distance (e.g., Mahalanobis).

6.2. Feature Extraction and Classification

As shown in Figure 6, for each classification, we take the current average epoch and compute the distance between this current epoch and the reference signals for each of the class references. If the distance is multinomial (e.g., Mahalanobis, Riemannian), we can give the signal across all channels as input (for Mahalanobis, the covariance matrix of the signals across channels is computed) and we obtain a single value for each class. In other words, if we compute the distance between the current average epoch and each reference epoch, regardless of the number of channels, we always get a single value for each class. If the distance is not multinomial and takes only a single variable as input, then we need to compute the distance channel by channel and obtain as many values as channels for each class. For example, if we have 16 channels, we compute the distance between Channel 1 of the current average epoch with Channel 1 of each of the reference epochs, the distance between Channel 2 of the epoch and Channel 2 of the references, and so on. Thus, we obtain 16 values for each class.

In the case that there are several reference epochs for each class (e.g., after 4 training trials there will be 4 references for each class), we need to compute the distance between each of the references and each of the reference epochs for each class. For example, if we use the Mahalanobis distance, we will obtain 4 values for each class. If we use a Spearman distance (takes only 1 channel as input) and we have 16 channels, we will obtain $4\times16 = 64$ values for each class.

In order to make a classification decision, for each n-tuple of values across n classes, we select the class that has the lowest distance value (arg min as shown in Figure 6). In the situation presented earlier, in which there are 64 values for each class, we thus have 64 n-tuples of values and obtain 64 classification decisions. In order to make an overall decision, we compute the distribution of classification outcomes over the 64 outcomes (normalized occurrence frequency) and obtain a probability distribution over all the classes. We can then select the class with the highest probability as the current classification result. In other words, the features we use are the filtered signals themselves that we feed to distance measures.

In this work, we use 10 ICA components; thus if we use a distance such as Spearman, we would have 10 distance values per reference and per class. With 4 references, that would be 40 distance values and thus classification decisions in total.

Moreover, in order to minimize the sensitivity of the classifier, we introduce a margin M that sets a minimum value for the distance between each feature and each of the class references. Normally, we would select the class with the lowest distance from the instance. When M is introduced, the minimum distance value should be lower than m, so when the value is higher, we always select the resting state class.

$$Classification = \begin{cases} argmin_{C_i} \{d(C_i, I)\}, \text{if } \min(d(C_i, I)) < M \\ C_{REST}, otherwise \end{cases}. \tag{1}$$

In this work, we compute M automatically in a way compatible with the Wachspress visualization method. The computation of M is presented in Section 6.4.

6.3. Wachspress Coordinate Projection for Visualization (CLBCI-WDV)

Any classifier that allows us to retrieve a posterior probability distribution or likelihood across all classes for the instance/feature vector under classification (i.e., a value between 0 and 1 associated to each possible class that denotes the probability that each class corresponds to the instance[4]), we can exploit the simple technique of Wachspress coordinates to visualize the distribution graphically by projecting it in a regular polygon. Wachspress coordinates are an extension of Barycentric coordinates for arbitrary convex polygons and allow expressing the coordinates of a point inside the polygon as a weighted linear combination of the vertices of the polygon. The weights represent the relative proximity of the point to each vertex. Given the set of weights, the Euclidean coordinates of the point can be calculated using Equation (2):

$$\begin{aligned} x_A &= \sum_{V_i \in P} w_i x_{V_i} \\ y_A &= \sum_{V_i \in P} w_i y_{V_i} \end{aligned} \tag{2}$$

where x_A, y_A are the coordinates of the project point in the polygon. $V_i \in P$ are the vertices of the polygon and (x_{V_i}, y_{V_i}) are its coordinates in the 2D Euclidean space. The w_i are weights associated to each vertex.

Wachspress coordinates have known little use in the specific context of the visualization of classification outcomes. The closest research that comes to our work is the use of Wachspress coordinates to create an intuitive control for a classifier fusion system of n classifiers [Talbot et al. 2009]. The vertices in a regular polygon represent each classifier and the position of a handle in the form of a point symbolizes the normalized weight distribution. Users can use their mouse to drag the handle in order to select the desired weight distribution [Talbot et al. 2009].

In general, we can consider that a classification outcome C_0 is a distribution of normalized ($\sum_{w_i \in W} w_i = 1$) weights $W = \{w_i | i \in [1, N]\}$, where N is the number of classes. For example, in the case of the simplest form of MDCs, if we normalize the distances obtained (so that they sum to 1), each of the class weights actually corresponds to the opposite of the distance between the class templates C_i and the current instance I: $w_i = 1 - d(I, C_i)$. Depending on the distance measure used, the number of channels/components, and the number of reference epochs for each class, as mentioned in Section 6.2, we will obtain a certain number of n-tuples. Each n-tuple will lead to a classification decision; the set of all decisions can be combined to obtain a global classification outcome C_0 where probabilities are the normalized occurrence frequencies of each class in the set of classification outcomes.

This means that we can visualize the classification outcome as a point, but also each of the individual distance n-tuple that led to the classification outcome (see Figure 7 for a graphical illustration).

[4]Even for classifiers that do not give a probability distribution over the classes, we can consider that the selected class has a probability of 1 while nonselected classes have a probability of 0.

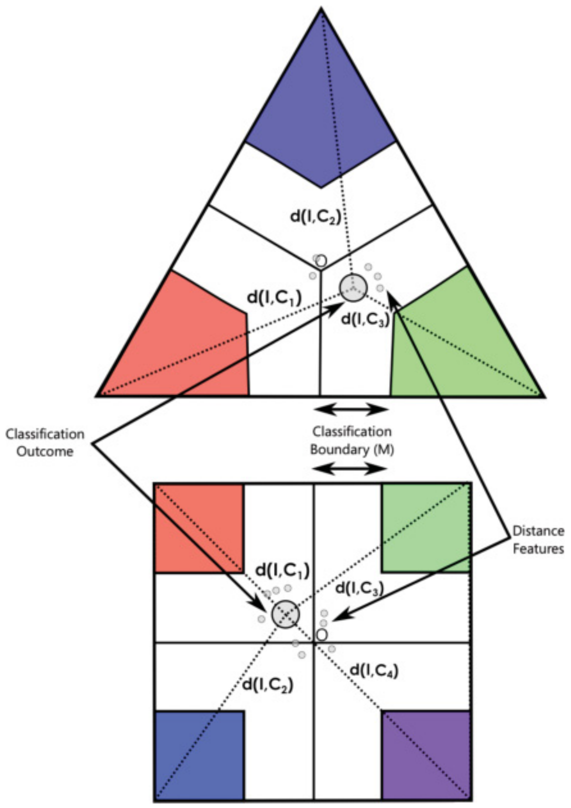


Fig. 7. An example of our 2D projected Wachspress visualization for a 3-class task and 4-class task.

By the definition of Wachspress coordinates, as described in Equation (2), the classification outcome distribution and the set of distance features across all classes actually exactly corresponds to the Wachspress coordinates of a point. Thus we could represent such distance distributions in a regular polygon, in which classification boundaries are the bisecting lines of each face that join in the center of the polygon.

The coordinates of each vertex V_i of the polygon are calculated in the following manner:

$$x_{v_i} = O_x + r \times \cos\left(\frac{2\pi \times i}{N} + \theta\right)$$
$$y_{v_i} = O_y + r \times \sin\left(\frac{2\pi \times i}{N} + \theta\right), \quad (3)$$

where O_x and O_y are the starting position offset of the polygon, r is the radius of the bounding circle to the polygon, N is the number of classes, and θ is the initial rotation of the polygon.

The classification boundaries of the minimum distance classifier can be drawn as the segments joining the middle of each segment and the center of the polygon (average of the coordinates of the vertices).

We call a classification margin M, a zone around the classification boundaries, where the classification result is "undefined." This is a way of reducing the amount of noise present in the classification process. We can draw points corresponding to classification or distance distributions by using Equation (2).

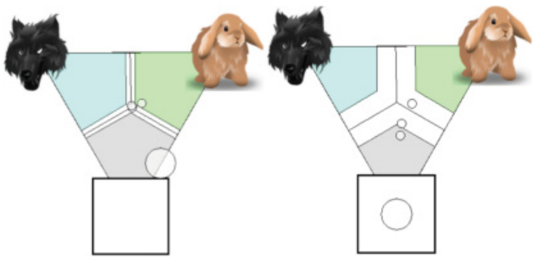


Fig. 8. Classification margin at 5% versus classification margin at 20%.

Figure 7 shows an example of polygons for a 3- and 4-class problem (triangle and square), in which we have a classification boundary, a classification point drawn (big circle) that corresponds to the classification outcome C_0 as well as several small circles that correspond to the distance n-tuples that led to the outcome C_0. When the classification result point is in the white zone, it means the classification is "undefined." When the point reaches one of the zones beyond the margin (in color), it means that the corresponding class is the classification result.

Later, we hypothesize that such a representation may potentially yield a better visualization compared to a simple bar (e.g., Figure 2(a)), because it explicitly represents the classes by a depicted image and corresponding text that reinforces the user's performance. Furthermore, this allows us to represent more than two classes without having to multiply the number of bars or axis.

One might argue that any more than 3 classes will lead to distortions in the distances. However, for the classification result point, we merely represent the proportion of distance features that are closer to each class, for which the mapping in a regular polygon is a good representation. On the other hand, for distance features, the rationale is to only keep the ratios between the distances intact, without necessarily having accurate positions of the points.

In a sense, what is important is whether the points pass the classification margin or not, and near what class they are located. As for the usefulness of displaying the distance features, we believe that it can be a positive element as long as there are only a few points. When the number of points increases, the clutter might require a much higher cognitive load for the user than necessary and thus negatively impact classification performance.

Based on the recommendations and critiques made by Lotte et al. [2013] on BCI training protocols and feedback, the idea of such visualization comes from the need of motivating the user through a more enticing and easy-to-understand feedback visualization. Moreover, we need to maintain a relatively low cognitive load in users. In turn, it must facilitate the independence of the user in interpreting the feedback.

6.4. Interface and Implementation

The point of entry into our application is a simple Launcher interface, which allows us to set most parameters pertaining to the BCI and to launch the application that we used during our experiments.

We made a prototype of our visual feedback method using Java/SWING that draws the visualization in exactly the same fashion as the illustration in Figures 8 and 10, with a wolf head and a rabbit. We added an image in each corner to represent the class that corresponds to each colored area. There is a text label in the top left corner that displays messages from the system (current class to calibrate, current class label, and so on), the visualization polygon in the middle, and a control bar at the bottom of the interface. We have added controls that allowed us to set certain parameters and manners tailored to each user prior to sessions dedicated to the task:

- Toggling the distance measure of the classifier (among Mahalanobis, Spearman, Riemannian distances)
- Triggering a new calibration phase
- Selecting individual training trials and deleting them (when a training trial is selected, all corresponding distance feature points turn red)

For the calibration phase, cues need to be given to users to indicate to them when to produce the desired signals. We displayed a message and magnified the image of the corresponding class. For the calibration of SSVEP, we made the class images flicker at the frequency associated with each class. The margin that changes in the classifier is immediately retranscribed on the visualization interface. Figure 8 illustrates a situation in which the margin goes from around 5% to 20%; 100% is the maximal relative distance possible (distances are normalized in the 0 to 1 range relative to the minimum and maximum values of the distribution before they can be projected in the polygon).

As mentioned in Sections 6.2 and 6.3, we use a classification margin M for the classifier that is also represented in WDV visualization and that is computed automatically. We compute M at the level of the visualization by computing the average Euclidean distance between the points definied by the distance n-tuples used to obtain the classification outcome. The coordinates for the points F_i are extrapolated from the normalized n-tuples using Equation (2) and the distance between all pairs of points is obtained by $D_{i,j} = D(F_i, F_j) = \sqrt{(F_{i_x} - F_{j_x})^2 + (F_{i_y} - F_{j_y})^2}$. Then, we compute M as the standard deviation (SD) of the pairwise Euclidean distances between the points as $M = StDev(D_{i,j})$. M is recomputed every time a new classification outcome is computed.

7. APPLICATION: EVENT-BASED CONTROL

Now that we have introduced the system, the visualization scheme, and the motivations behind them, we endeavor to evaluate the performance of our BCI architecture.

As we have seen in Section 2, there are event-based BCIs and continuous-control BCIs. As a first step, we want to evaluate the potential of our system with event-based control, which is the simpler type to tackle.

In event-based control tasks, the BCI is used to trigger events in the application. Event-based interaction is typically well suited for choosing between several items or options. We will use the two most common BCI paradigms for event-based interaction (SSVEP and MI). Although SSVEP typically requires little training [Nicolas-Alonso and Gomez-Gil 2012], which is not necessarily relevant in a study about user learning, the rationale behind using SSVEP is that it is a reactive BCI as opposed to an active BCI like MI. This allows us to see how the architecture behaves on two different types of BCI paradigms and whether user learning actually has an influence on SSVEP as a reactive paradigm.

The objective of this experiment is to evaluate the benefits of our system compared to a standard BCI system during several sessions over time. In this manner, we can gauge the potential of the feedback components of our system to introduce a more important learning effect in users. Namely, we want to evaluate the contributions of both our user feedback and system feedback (CLBCI-WDV).

Given that we have a particular interest in exploiting BCIs for games (as described in Section 5.3), we designed the experimental task as a small game exploiting BCI control to get the Little Red Riding Hood character home. The purpose of the game was only to be used to evaluate the BCI system as opposed to a serious game.

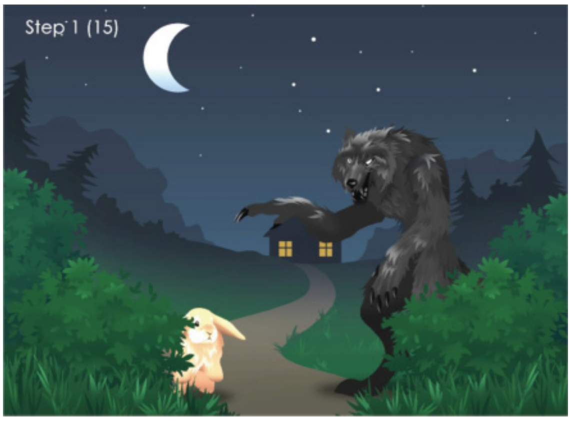


Fig. 9. The game at the start of a turn.

7.1. Task

In the shooter game, the user embodies the Little Red Riding Hood character and has to go home, but many dangers lurk on the path. At the beginning of each turn, as seen in Figure 9, the user hears the rabbit or the wolf. If the wolf is heard, then the user must “select” the wolf to shoot towards it and wound it. If the rabbit is heard, the user needs to “select” the rabbit, in which case the user grabs the rabbit to rescue it. If the user fails to “select” the wolf, an animation shows the wolf scratching the user. If the user fails to “select” the rabbit, then he or she sees an animation, showing the wolf snatching the rabbit. If the user wins a turn (the selection of an animal was correctly performed), the user gets a step closer to the house, otherwise the user remains in place. Between each turn, there is a 3s resting period.

In the case of MI, the wolf and the rabbit appear on screen and the user has to imagine feet movement to select the rabbit and hands movement to select the wolf. In the context of the game, imagining hand movement makes the hands of the character move to shoot the wolf, which is consistent with imagined hand movement. Moreover, imagining feet moving does also correspond to the idea that the rabbit is jumping around. We instructed the users to imagine that they were clenching both fists for imagined hand movement and that they were tapping both feet on the ground for imagined feet movement. For SSVEP, the wolf and the rabbit flicker at the frequencies of 15Hz and 20Hz; users must look at the one they want to select. The choice of the frequencies was constrained by the refresh rate of the screen, here 60Hz. To create a flickering, we need to alternate frame in states “on” and “off,” which means that the maximum stimulation frequency is 30Hz. The frequencies have to be multiples of 60 and not multiples of each other (to avoid interference from harmonics) [Cecotti et al. 2010].

During the game session, each trial (1 trial = 1 game turn) lasted for 10s, with 5s rest between trials. The end condition was either being successful in 10 turns or reaching the time limit of 10min. This means that there are always at most 10 correct turns, and in practice no users reached the 10min limit. Even with a random classifier, 10 correct turns are reached before the time limit. This means that, for example, for 66% task accuracy there are 10 correct and 5 incorrect turns.

It is important to note that we made certain that the animations and sounds of the game do not interfere with the BCI classification. The sound indicating the animal expected to be selected is always played before the user has to select one class, and only after a classification result is obtained do we play an animation.

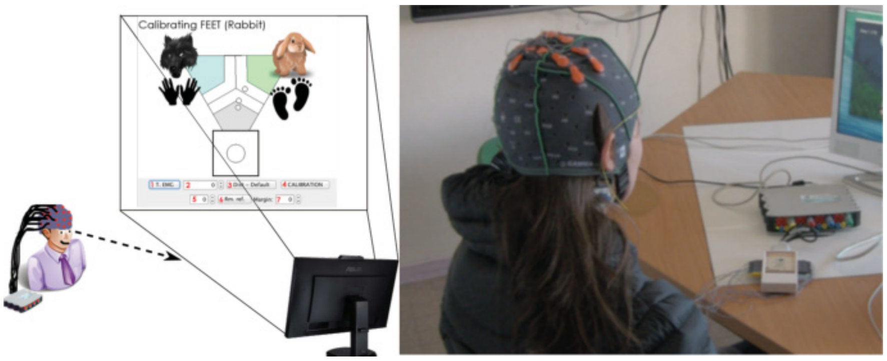
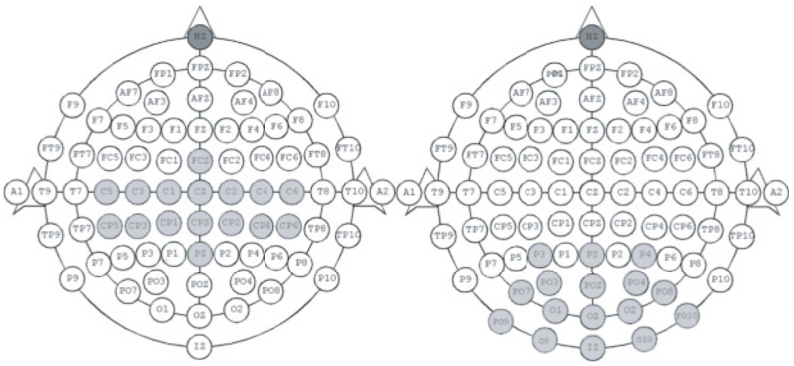


Fig. 10. An illustration of a subject during the experiment.



(a) Motor Imagery (b) SSVEP

Fig. 11. Electrode placement according to the extended 10- to 20-electrode placement system. We used a nazillion reference and a mastoid ground (right side).

Given these objectives, we make two hypotheses as to the expected behavior of the system: first, that the combination of CLBCI, user feedback, and the WDV visualization scheme will help users modulate their brainwaves better and induce a learning effect greater than in a standard state-of-the-art BCI. Our second hypothesis is that both visualization and user feedback are complementary and that visualization alone is not sufficient.

7.2. Experimental Setting

Twenty-four subjects (6 per group) aged 20 to 40 took part in the experiment (see Figure 10 for an illustration of the experimental setting). None had prior BCI experience. We randomly allocated the subjects into experimental groups.

For the user study, we use a g.tec USBAmp[5] 16-channel EEG amplifier for the acquisition of the EEG signals. The electrode positioning is shown in Figure 11. Our system ran on a Macbook Air with an i7 processor at 2.93GHz and 8GB of RAM.

[5]http://www.gtec.at/Products/Hardware-and-Accessories/g.USBamp-Specs-Features.

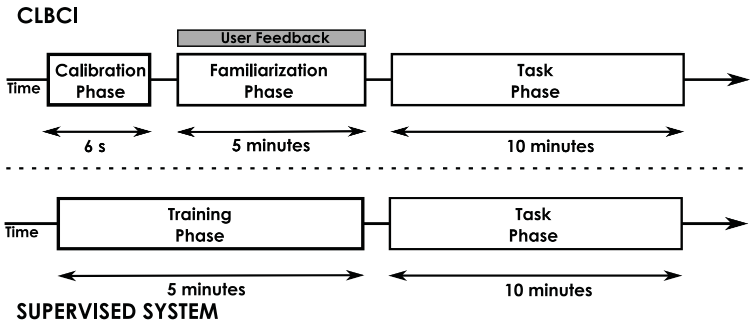


Fig. 12. Session of our system versus a supervised system.

7.3. Protocol

In order to verify the contribution of both feedback types to the results of our system, we need to consider two experimental conditions: (±) WDV, the presence of our visual feedback paradigm; (±) UF, the presence of direct user feedback for training.

The meaningful combinations to study are (+WDV, +UF), the full CLBCI system experience; (+WDV, −UF), in which only visual WDV feedback is provided without user feedback; and (−WDV, +UF), in which there was no WDV feedback but for which users can give feedback. While the (−WDV, −UF) combination does not make sense, as performance is assured to be random, we do not consider it. However, it is useful to compare our CLBCI system to a baseline system that corresponds to the prevailing training paradigm in the state of the art, thus we considered the OpenViBE implementations for both MI and SSVEP and modified them so that they interface with our game, as explained later in this section.

In summary, the experimental groups we considered are the following:

- G1 External Baseline system: Synchronous Supervised 2-class MI and SSVEP implementations from OpenViBE (Figure 12, bottom)
- G2 (+WDV, +UF): CLBCI with WDV visualization and user feedback during the familiarization phase (Figure 12, top)
- G3 (+WDV, −UF): CLBCI with only WDV visualization and no user feedback phase (Figure 12, top)
- G4 (−WDV, +UF): CLBCI with only user feedback and no WDV visualization phase (Figure 12, top)
- Random baseline: Simulated users with a classifier that randomly (uniform distribution) selects a class

The OpenViBE system is meant as an external baseline in order to have a general point of reference of where the full CLBCI system stands compared to a standard system. We will not compare G3 or G4 with the OpenViBE system.

For each of the groups except for the random baseline, we followed the following protocol below:

- 5-10min of explanation (slideshow with a detailed description of the interface)
- 5min to set up the hardware
- For each BCI paradigm (SSVEP and MI; see Figure 12 for a graphical representation):
  - 10min of calibration, familiarization
  - 10min to play the game
  - 5min of rest
- Questionnaires

The other dimension of the experiment is to show how the contribution to user learning of each of the experimental conditions evolves over time. We performed three sessions. The first session was followed with a second session 3 days to 1 week later and then by a third session 2 weeks after the second session. The rationale for the time between the first two sessions was to leave enough time so that we were certain that users had any time to assimilate what they learned. We followed the recommendations made by Kaiser et al. [2011], who argued that 3 days is the minimum time required to discount any insignificant short-term learning effect. The second session took place at most a week later (we tried to keep the time between sessions across all subjects as consistent as possible given their constraints).

We now review the groups and detail the specificities of the protocol for the training phases or calibration + familiarization in each case:

- **Group 1, Synchronous Supervised Baseline System:**
  Group 1 can be considered as a baseline, as we make use of standard supervised synchronous implementations of both MI and SSVEP from OpenViBE[6]. For MI, a CSP filter is first trained and then used as a spatial filter. Then, band power features are extracted and used to train an LDA classifier that shows state-of-the-art performance [Neuper et al. 2006]. As for SSVEP, a classifier is trained for each of the flickering targets to detect whether the user is looking at that particular target or not. The classification for each target was performed between the class in question and a resting state. A CSP filter was trained for each class, and power band features were used around the flickering frequencies of each of the targets. For example, for the 15Hz target, the frequency window selected was between 14.5Hz and 15.5Hz. See Legény et al. [2013], who also use the OpenViBE SSVEP scenario, for more details. The exact protocol followed for MI and SSVEP was:
  - For MI, for each trial: a cross appears on screen notifying the user to get ready (see Figure 2(a)). Two seconds later, a visual cue is displayed to indicate to the user which class the trial is for. The trial begins and real-time continuous feedback is displayed (Figure 2(a)) for 3,750ms. A 3s resting phase follows, and the cross is displayed again.
  - For SSVEP, the training screen had three flickering targets and one nonflickering target (for the resting state) by default, and the scenario was meant for three SSVEP classes. We removed one of the classes and hid the corresponding target. The two remaining flickering targets were set to 15Hz and 20Hz, respectively. For each trial, the system displayed a small arrow below the flickering target that users should look at for the trial. Two seconds later, the actual trial begins and lasts 3,750ms, followed by a 3s rest. Then the arrow goes to the next target, and so forth. For both SSVEP and MI, the training phase consisted of 20 trials, which amount to 5min each in total.
  - We adapted the timing of the OpenViBE training protocol to match that of CLBCI and modified the scenario to have a preprocessing of the EEG signals as close as possible to CLBCI (1s epochs, with average epochs computed from 5 epochs, with a 250ms overlap). The classification phase in OpenViBE did not always coincide with the beginning of a game session. We made sure to ignore input only from an OpenViBE classification phase that began before a game turn and then ended during the turn, as the classification could be performed before the user hears the stimulus in the game.

[6]http://openvibe.inria.fr.

- **Groups 2, 3, 4 with CLBCI:**
  In the CLBCI system, users are subjected to a calibration of one trial per class and then go through the familiarization phase.

  In this phase, users are instructed to set classification outcome goals for themselves and attempt to achieve them through the use of the BCI with the help of the WDV feedback interface. They could then give feedback if necessary. Feedback takes the following forms:
  - Triggering an additional training cycle (1 training trial for each class) to add supplemental references for each class
  - Removing previously added reference signals (including those stemming from the initial calibration)
  - Changing the distance measure used (see Appendix for an analysis of the reason behind this possibility)

  During the familiarization phase, an operator was present (distinct from the user) to guide and advise the user. See Section 7.3 for a detailed account of the role of the operator.

  In Group 4, in which there was no visual feedback and given that we know that feedback is a requirement for BCI learning, there still must be another form of feedback for user feedback to make sense. Thus, for -WDV the WDV visualization was hidden from users; however, the operator could see the WDV feedback and provide feedback vocally to users. The operator set goals for the user, observed whether the resulting classifications were correct, and reported the results to the user. Upon these observations, the operator suggested one of the possible actions (changing the distance, adding a new calibration trial, removing an existing calibration trial).

- **Random Baseline:** There are sound studies detailing the empirical classification performance for a classifier following a uniform random distribution [Müller-Putz and Scherer 2008]. For example, in the case of a 3-class problem (2 classes + resting state, as for the other systems evaluated), the empirical classification performance of a random classifier can reach up to between 50% (95% confidence interval (CI)) and 60% (99% CI) for 20 training trials and up to 70% for a 2-class problem. However, we are evaluating task performance in this experiment, and we have no way of directly using these results for random classification performance to infer the resulting "random" task performance. However, we need to determine whether each of the groups achieves a performance that is significantly different from a random task performance. To this end, we introduce the random baseline, in which we simulate a user with an agent that uses a random classifier with a uniform class selection probability. We then measure task performance in the same way as we would for users. Naturally, there is no user feedback or visualization involved.

The WDV and OpenViBE feedback were always made available on a second screen next to the game.

For all groups, after training or calibration + familiarization, we moved on to the task phase, in which users played one game session. As described in the Task subsection, during the game session, each trial (1 trial = 1 game turn) lasted for 10s, with 3s rest between trials. The end condition was either being successful in 10 turns or reaching the time limit of 10min.

Each user performed one session with MI and SSVEP on the first day, then the same session during the second and third days.

We strictly performed a new training/calibration of the classifier before each session; in other words, we did not reuse the training from Session 1 in subsequent sessions. We want to evaluate user learning alone and not the cumulative learning of the system.

The latter is bound to improve over time, which would preclude us from drawing any conclusions from the experiments.

7.4. User Feedback and Operator Intervention

We thoroughly explained the interface and how to operate it in simple terms to users. We prepared a set of slides to show and explain the following:

- The visualization paradigm in intuitive terms that did not require any understanding of EEG. We explained that the edges of the triangle corresponded to each of the actions to recognize and that the big point represented how close they were to each of the actions. We explained that when the point was in a color zone, the action associated with the zone was triggered. Then we explained the margin by saying that the point tends to be "jittery" and that the margin gives some leeway so that the point doesn't suddenly jump from one zone to another. We explained the smaller points as being the constituents of the bigger point and that the bigger point is the average of the small ones. Users understood the notions intuitively. At no time did we evoke EEG.
- Then, we explained the calibration procedure as a phase in which we need to capture examples of how they think of each class. We explained that the examples will be used to compare to what they are currently thinking about to see how "close" it is to the examples.
- Subsequently, we explained strategies that the users should employ for imagining moving their hands (werewolf) or feet (rabbit).
- We then presented each of the buttons of the interface and told the users what they were for and how to operate them. For example, we explained that the reference deletion removed some of the examples that may be bad: examples that may have been captured while the user was thinking of something else.

The operator is an expert who knows the inner workings of the CLBCI system. Users understood the instructions; however, during the familiarization phase, they usually asked the operator for advice or to actuate the interface in their stead, as that was more comfortable. The only case in which the operator had to intervene on his or her own accord and to make a decision instead of the user was when the user was not interactive at all (be it with the operator or the interface).

We instructed the users to employ the following protocol when interacting with the BCI: give oneself a goal classification outcome (e.g., rabbit) and try to get the desired result. If no control is felt, change the distance and try again with the goal setting, to determine if there is control. Then try to remove references by selecting one of the references and by observing the behavior of the small feature points. Finally, if a reference is removed, trigger a new calibration trial by pressing the appropriate button.

In order to avoid interference, we asked users to stop moving after setting themselves a goal and waiting at least 5s before gauging if they have control. For additional training trials triggered by the user, there was a delay before the capture started. All epochs around the moment during which the user clicked on the button in the interface were rejected and ignored. This way, we could strictly ensure that motor and other artifacts were not captured during training trials to not create biases in the nature of the BCI control (e.g., control due to motor artifacts rather than MI).

7.5. Quantitative Results, Discussion, and Analysis

We now consider the evaluation and analysis of the results both quantitatively and qualitatively. For the evaluation in quantitative terms, we considered two commonly used intrinsic measures:

Table I. Results for the 4 Groups, in Terms of Accuracy (Task Success Rate) as well as Average Interaction Time for SSVEP

	Session 1 (S1)		Session 2 (S2)		Session 3 (S3)	
	Accuracy	Time	Accuracy	Time	Accuracy	Time
OpenVi BE Baseline (G1)	**61.09%** $\sigma = 3.32\%$	4144ms $\sigma =$1649ms	**63.23%** $\sigma =2.96\%$	3407ms $\sigma =$1033ms	68.52% $\sigma =2.32\%$	3254ms $\sigma = 1534ms$
+WDV, +UF (G2)	51.06% $\sigma = 9.02\%$	**3766ms** $\sigma =$2055ms	**63.37%** $\sigma = 10.85\%$	**2897ms** $\sigma =$1171ms	**72.66%** $\sigma = 2.32\%$	**2798ms** $\sigma = 993ms$
+WDV, −UF (G3)	47.93% $\sigma = 12.29\%$	4728ms $\sigma =$2361ms	47.10% $\sigma = 12.38\%$	9194ms $\sigma = 1655ms$	49.32% $\sigma = 18.33\%$	8409ms $\sigma =$ 1338ms
−WDV, +UF (G4)	49.70% $\sigma =14.09\%$	4123ms $\sigma = 4345ms$	46.74% $\sigma =11.41\%$	4089ms $\sigma = 8765ms$	51.96% $\sigma =10.05\%$	4079ms $\sigma = 8656ms$

Table II. Results for the 4 groups, in Terms of Accuracy (Task Success Rate) as well as the Average Interaction Time for MI

	Session 1 (S1)		Session 2 (S2)		Session 3 (S3)	
	Accuracy	Time	Accuracy	Time	Accuracy	Time
Open Vi BE Baseline (G1)	**61.28%** $\sigma =3.72\%$	**4169ms** $\sigma =$1448ms	69.29% $\sigma =3.43\%$	**3651ms** $\sigma =$1705ms	72.59% $\sigma =3.51\%$	**3454ms** $\sigma = 1465ms$
+WDV, +UF (G2)	51.77% $\sigma =14.72\%$	4599ms $\sigma =$2835ms	**72.83%** $\sigma = 3.56\%$	**3530ms** $\sigma = 1322ms$	**81.16%** $\sigma = 7.91\%$	**3467ms** $\sigma = 1245ms$
+WDV, −UF (G3)	52.43% $\sigma = 13.73$	5403ms $\sigma = 3649ms$	51.32% $\sigma = 7.10\%$	6110ms $\sigma =$2628ms	51.92% $\sigma =3.10\%$	8210ms $\sigma =$5789ms
−WDV, +UF (G4)	58.26% $\sigma =15.11\%$	6653ms $\sigma = 5565ms$	61.32% $\sigma =4.17\%$	6610ms $\sigma =$ 2139ms	60.93% $\sigma = 5.01\%$	5809ms $\sigma =$ 3091ms

- The ratio of successful turns over unsuccessful turns
- The average time in ms necessary before a correct classification occurs at each turn. If the user remained in the resting state for 10s, the classification was considered incorrect by timeout.

For the qualitative evaluation, we ask users to fill out a survey after each session in order to evaluate their subjective views about our feedback interface and its effectiveness (classification visualization and distance feature visualization). After the second and third sessions, we additionally asked whether users felt any improvements in their performance.

For the statistical validation of the results, we first check that the distribution of values is normal with the Shapiro-Wilk normality test. We perform an ANOVA followed by a Tukey Honest Significant Difference (HSD) post-hoc test with adapted p values ($p < 0.01$ threshold). If the conditions were not met, we used a nonparametric test instead: Kruskall-Wallis for group significance and Tukey HSD, in which the t-test was replaced by the Mann-Whitney-U test. Given that we had only 6 participants per group, in several cases we fell back on the nonparametric tests. The results are presented in Table I for SSVEP and Table II for MI.

When there is no feedback, the performance in terms of accuracy for both MI and SSVEP is around 50% correct trials; there are no improvements in the second session ($p > 0.01$ not significant). During the first session, for both MI and SSVEP, CLBCI (+WDV +UF) was around 51%. The supervised system baseline showed better accuracy for MI and for SSVEP (MI = 61.28%; SSVEP = 61.09%; $p < 0.01$ significant). In terms of classification time, both CLBCI (+UF, +WDV) and the supervised systems showed times on average around 4s (SSVEP and MI; $p > 0.01$ not significant).

During the second session, the accuracy of the supervised systems remains almost the same and the difference is not significant. For CLBCI, in S2, the accuracy for MI has risen to 72.83% ($p < 0.01$ compared to S1, significant). For SSVEP, the accuracy during S2 is also better (69.37%; $p < 0.01$ significant). Thus, during the second session, the performance of WDV is better than the supervised systems. The average classification time remained stable between S1 and S2. The accuracy performances of MI and SSVEP are roughly equivalent ($p > 0.01$, difference not significant). For SSVEP, the difference in classification accuracy between CLBCI (+WDF, +UF) is not significant (A = 63.37%, SD = 10.85% vs. A = 63.23%, SD = 2.96%, $p > 0.01$), whereas for MI the difference is significant (A = 72.83%, SD = 3.56% vs. A = 69.29%, SD = 3.43%, $p < 0.01$). Classification time is significantly different for SSVEP but not for MI.

If we remove either WDV or UF from CLBCI, we see that the performance evolves little between the two sessions. It is clearly lower than that of CLBCI and the supervised system in the second session (in the order of 10%, $p < 0.01$).

For the last session, we observe that the same trend is further exacerbated. The supervised system (−UF − WDV) shows a little improvement in terms of task accuracy both for MI and SSVEP in the order of 4% (MI = 70.67%; SSVEP = 67.45%; $p > 0.01$ not significant). As for the mean time to classification, it remains the same. It is also the same with (+WDV, −UF) and (−WDV, +UF), which show marginal differences in the last session. Finally, for CLBCI, the improvement during the third session has remained consistent. The improvement is around 6% from the previous session, both for SSVEP and MI. Classification times have remained practically identical on average for (+WDV, −UF) and (−WDV, +UF); however, we note a decrease of 1 to 2s for CLBCI (+WDV+UF) in this third session. The differences between CLBCI (G2) and OpenViBE (G1) were significant for both MI (+9%) and SSVEP (+4%) in terms of classification performance. For classification time, the difference was only significant for SSVEP.

One contestation that may arise is that since we only have two classes, we would have trouble differentiating whether the results are not simply due to a random classification. To this end, we generated random results several times and averaged them to see the nature of the distributions due to random classification to serve as a baseline (Figure 13). The shape of the accuracy distribution is very different from that of the results from the other systems, even in situations in which the mean of the random baseline came close to that of one of the systems. As one would expect, the means were equal to about 50% classification accuracy. The differences between G3 and G4 were hardly significantly different from the random baseline in any of the sessions. As for CLBCI, the first session did not lead to statistically significant results for both MI and SSVEP, but Sessions 2 and 3 did. The baseline system (G1) always led to statistically significant differences from the random baseline.

All in all, the improvement of users with CLBCI is significant and has persisted over time: this suggests that WDV and user feedback together led to a durable learning effect. As for interaction times, CLBCI and the supervised system are on par, while G3 and G4 have similar averages but much larger interquartile ranges.

Furthermore, the traditional system showed much less significant improvements for users: the learning effect is smaller. Thus we have supporting evidence for the validity of H1. Note that on average, through feedback, we obtained 5 to 6 additional

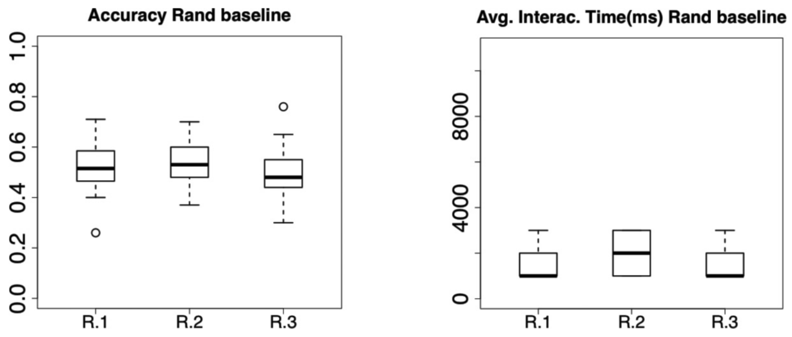


Fig. 13. Box plot of the random baseline results, emanating from a uniformly random classifier (selects a random class uniformly every second, the same classification interval as that for users).

training trials for G2 and G4. In addition, the results of G3 and G4 clearly confirm that just user feedback or the visualization scheme on their own were not sufficient to induce better learning in users. However, their conjunction in CLBCI did, which supports H2.

There are many factors that could explain the results of Group 2 and of CLBCI: performing additional calibration trials, deleting references, or changing the distance. In order to have a better idea of what happened, we performed a post-hoc analysis on the results of each user in the case of MI. In the original experimental setting, given that standard deviations were on average around 3.5 with a sample size of 6 subjects and a difference of means of 5%, the statistical power is 0.6975, which is insufficient for a finer grained analysis. We need 14 users to reach a statistical power of 0.9658 that is more adequate for a significance level of 0.05. Thus, we had 8 (same age group as the original population, with no prior BCI experience) more users perform the experiment in the same +WDF, +UF condition in the context of this post-hoc analysis.

We first examine results in relation to the distance measure chosen. The average performance with the Mahalanobis distance was 81.13% (SD 2.34%), for the Riemannian distance it was 82.11% (SD 1.96%) and for the Spearman rank correlation distance it was 80.24% (SD 2.12%). The differences observed are not significant. In the offline experiments, we performed prior to designing the experiments (Feature validation appendix), the optimal distance measure depended on the subject. The fact that here the average performance is about the same regardless of the distance suggests that selecting the distance for each user is beneficial as opposed to imposing one distance in particular.

Another important factor is the possibility of performing additional calibration trials and removing already captured references. The first step in analyzing the performance is to look at the distribution of users depending both on the number of calibration trials and the number of deletions. Figure 14 shows this as a 3D bar chart. Some combinations are impossible, as to perform n deletions there needs to be at least n+1 calibration trails. We can see that users tend to perform 2 or 3 calibration trials the most (on average 3.08, SD 1.24) and 1 deletion the most (average 1.41, SD 0.99).

We can now have a closer look at the resulting task accuracy percentage for individual values of the number of trials and the number of deletions. Figure 15 presents the results as bar charts (a) for the number of calibration trials and (b) for the number of deletions. We can clearly see that for a single calibration trail and no deletions, the performance is significantly lower (67.24% SD, 2.13%, and 69.47% SD, 1.45%, respectively). However, for other values of the number of calibrations and of deletions,

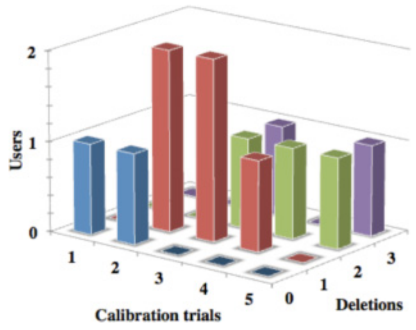
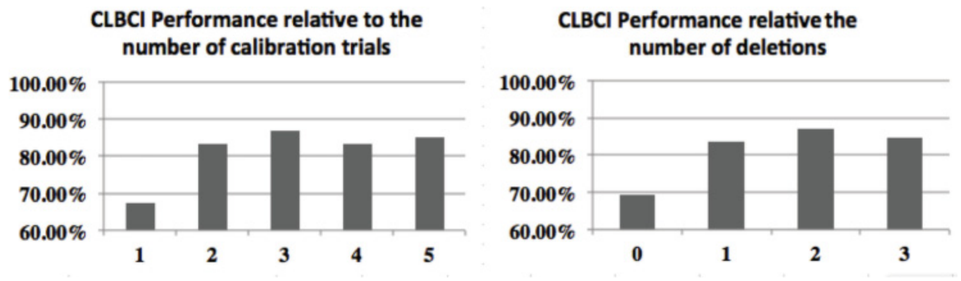


Fig. 14. 3D bar chart representing the distribution of users with regard to the number of calibration trials and of reference deletions.



(a) Number of calibration trials (b) Number of deletions

Fig. 15. Performance breakdown for +WDV +UF performance depending on the number of calibration trials and number of deletions.

there are no significant differences between the accuracies obtained. This indicates that it is preferable to make the user perform at least 1 additional calibration trial after the initial calibration and to delete at least one of the previously acquired references.

This indicates that it is always better to remove the first calibration trial and that the interaction prior to making a new calibration trial with the visualization interface leads to the second calibration trial being better. However, we cannot conclude anything more about the additional calibration trials after the first two or for deletions above 1, as the results suggest that it is dependent on users.

7.6. Qualitative Results, Analysis and Discussion

Figure 16 presents a few selected results coming from the questionnaires. Questions (a) to (e) pertain to the visualization featured in WDV and are averaged over all sessions in which WDV is present. Questions (f) to (g) pertain to whether users from each group, respectively (G1 – Baseline, G2 – +WDV, +UF, G3 – +WDV, −UF, G4 – −WDV, + UF), felt any improvement in subsequent sessions.

A majority of users felt that our visualization interface method was very useful (4/6) and was based on a meaningful representation (4/6) in terms of the classification feedback. Control over distance features was deemed adequate for half the users (3/6); half of the users felt they had moderate to no control over them (2/6 and 1/6).

Overall, most users believed that the visualization helped them to improve their performance (4/6). As for the perceived improvement between S1 and S2 (three or more days apart), for the traditional system users are torn: half felt that they improved and

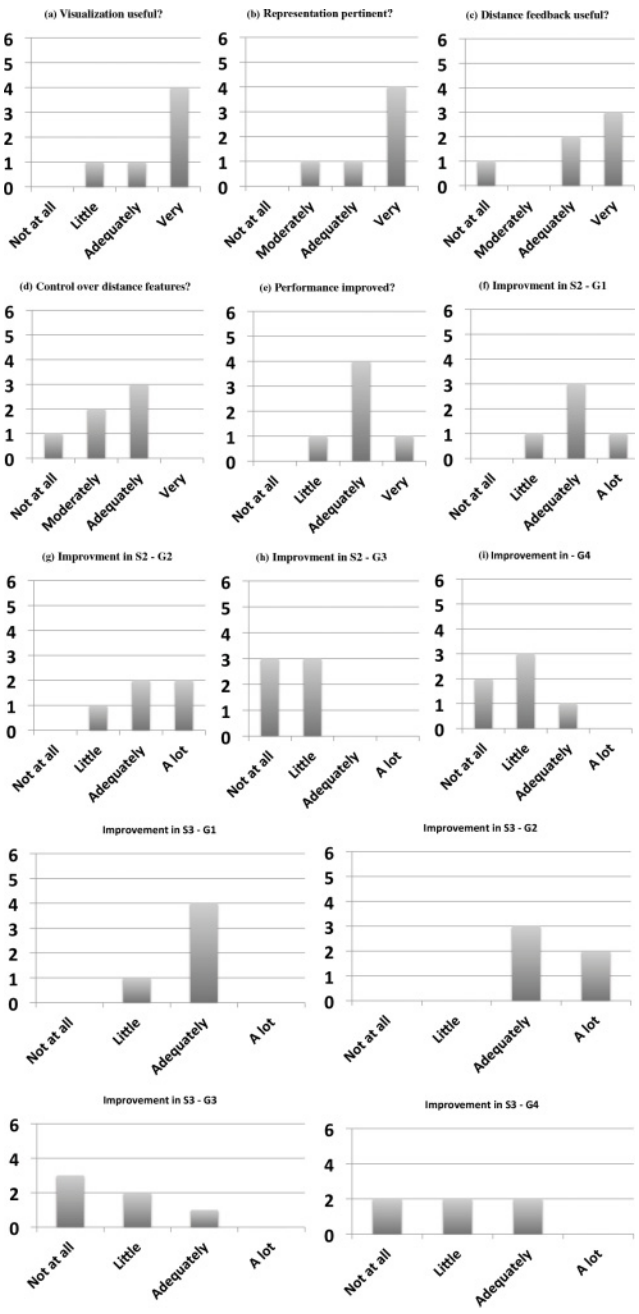


Fig. 16. Bar chart summary of the qualitative survey.

the other half felt that they did not. As for the user feedback with no visualization, users for the most part felt no or little improvement in the second session (2/6 and 3/6).

For WDV, most users felt that they improved adequately or a lot (2/6 and 2/6), with only one user who felt little improvement in the second session. For the system with no feedback, all users found little to no improvement between Sessions 1 and 2 (2/6 and 3/6), which corresponds to the quantitative findings.

As for the improvement felt by users after the third session, the results remained about the same for the supervised system (G1). Only one subject changed feedback from “a lot” into “adequately,” raising the number of subjects responding “adequately” to 4/6. For CLBCI (G2), the majority remains on “adequately” (3/6) and “a lot” (3/6). For both G3 and G4, one more user changed the opinion that the improvement was adequate; otherwise, the opinions remain the same.

8. LIMITATIONS

From our experiments, we obtained satisfactory results; however, there are within our system a certain number of limitations and specific conditions that mandate the successful use of the system.

8.1. Calibration

Given the results, we can clearly see that in the first session users did not know how to operate the system. They most likely did not provide proper reference signals, which compromised the performance of CLBCI. As the calibration phase is short, it is all the more crucial that it is done correctly in order to ensure a good initial classification. Using, as Barachant et al. [2012] suggest, a signal database to initialize the minimum distance classifier would lead to a better initial preperformance, possibly on par with the supervised system. However, in the second session, once users better understood what to do after the feedback session and initialize the system properly, we observed a much better performance.

8.2. Adaptation to Other BCI Paradigms

While a design around distance measures makes an easy adaptation of the system for reactive or active BCI paradigms possible, there is some effort involved in making the adaptation successful. First, the best distance measure might not be the same. Second, some aspects of the signal processing step need to be adapted: for example, with SSVEP it is best to have a separate Butterworth filter narrowly restricting the frequencies kept to those corresponding to the flickering. Similarly, some distances, such as the Riemannian distance as proposed by Barachant et al. [2012], have a specific covariance matrix form for each type of BCI paradigm and may not readily be set into place.

8.3. Sensitivity to Noise

As we used minimum distance classification, which is based on single-trial classification, the amount of averaging of the signal is not necessarily sufficient to allow for a perfectly stable classification. The classifier might undergo rapid successions of classification changes. In such cases, some form of classification smoothing must be employed. One way is to have a majority vote on several successive classification outputs before allowing the classification result to change. Another solution is to set-up a boundary for the minimum distance discrimination through the constraint that the values of the distances to each reference class must have a pairwise difference greater than a certain constant delta. Again, the values for the specific parameters are highly subject-dependent and may need to be manually selected or approximated through some heuristic.

8.4. Visualization

Although the visualization we propose is well suited to our constraints, the fact that distances are not preserved is problematic for higher numbers of classes, especially when trying to visualize distance features. The fact that the polygon we use is regular means that vertices are equidistant, which is equivalent to making the assumption that class references are equidistant from one another (this issue is shared with radial class visualization), which is very likely not the case. The integration of elements of MDS may be used in order to compute more representative positions for vertices, at the cost of a less user-friendly and clear interface.

Another limitation is that our visualization paradigm is well suited for a small amount of distance features. When there are too many features, the display gets too cluttered and individual features become difficult to track, although users can still appreciate the trend in the movement of the feature points.

8.5. Generalizability to Other BCI Systems

In this article, we study the effect of user feedback and system feedback only on the CLBCI system. OpenViBE serves as an external baseline. The next step in the study of WDV and user feedback is to generalize them to other BCI systems and platforms. For example, we could implement WDV and user feedback in OpenViBE and see if there are any performance improvements with the addition of WDV or UF.

9. APPLICATION TO OTHER DOMAINS

As pointed out in the related work IML, variations of our approach exist and are usable with a wide range of machine learning applications. However, the exact system we proposed is only directly applicable to a subset of ML applications that involve a streaming real-time classification. This specifically pertains to either physiological signals from the user or in general any process that is related to detecting the subjective preferences of users. Such applications can include speech-to-text processing, handwriting detection, or detection of subjective preference depending on the mood of the user and other such applications.

10. CONCLUSIONS AND FUTURE WORK

We have presented CLBCI that integrates user feedback and WDV, a visualization method for classification outcomes for BCIs. Through our experiments in the context of a simple shooter game, we evaluated our system in a user study in which we show that deploying a simple BCI with minimal training and then training both the user and the system iteratively through mutual feedback is feasible. We also show that it is the synergy between user feedback and system feedback that allows a good performance in the setting of the regular use of a BCI as a control modality (i.e., for a one-off use, a standard BCI with a long training phase would work better). Furthermore, a qualitative survey of our subjects indicates that the visualization scheme is appreciated and well understood by users for the classification outcomes, which may explain the success of the synergy of feedbacks.

We maintain that our combination of WDV and UF shows that it is possible to make a general framework in which users can intuitively operate the system without having to learn needless details about the neurobiology and signal processing.

One of the next steps in our work is to extend the system further to accommodate better for continuous-control tasks rather than event-based tasks and to improve on the visualization scheme to make the 2D mapping more accurate. Another step is to implement WDV/UF as a standalone program that can be used for other BCI systems such as OpenViBE or BCI2000 through standard protocols.

APPENDIX

1. Feature Validation

We considered the pairwise combination of several distance measures to build the classifier. In order to choose the best one, we need to perform a prior offline analysis. We recorded signals from two subjects and considered three distance measures for each BCI paradigm: Mahalanobis (Maha), Spearman Correlation (Corr), Riemannian distance (Riem).

In order to calculate how representative the distance measures (noted d (x,y)) we used are (and the resulting features), we calculated, for each measure and each component, the correlation between the distance differences among d(left, current), d(right, current), d(rest, current) and the expected classification result. In other words, we can estimate the classification accuracy at the individual distance and component level. Table I shows the results of this analysis (classification accuracy and the p value for each combination of distance measures). In bold is the best combination. We did not find that any of the distances had an absolute advantage over the others: it varies from paradigm to paradigm and from subject to subject. Thus we decided to add a control that allows us to dynamically switch the distance used during the online phase so that it could be easily adapted to each user.

Table I.. Accuracy and Significance of Distance Measure Combination in our Distance-Based-Classifier

	Distance Metrics	Accuracy Subject A	Accuracy Subject B
MI	Maha.	75.25 %	75.25 %
	Riem.	**76.23** %	74.88 %
	Corr.	75.34 %	**76.53** %
SSVEP	Maha.	**73.01** %	**73.35** %
	Riem.	72.44 %	71.44 %
	Corr.	**73.27** %	72.24 %

2. Classification

In a further effort towards reducing the sensitivity of the classifier and rapid successive classification changes, we added a threshold of successive identical classification outcomes before the system classification output changed. As seen in Figure 1, initially the classification result is left twice consecutively, then the classifier gives a right; however, the system continues to output left. For the system to change its output to right, we need 4 successive right classifications from the classifier (with no gaps).

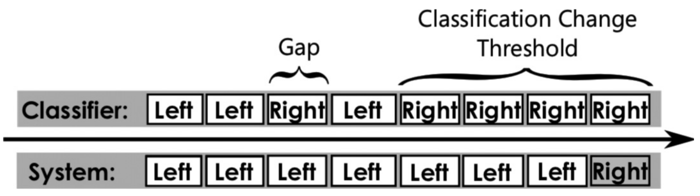


Fig. 1. The classification change threshold.

3. Visualization

The margin M in the visualization interface that exactly corresponds to the idea of margin is defined in the Minimum Distance Classifier. M is a real number between 0 and 1 and represents a minimum threshold for the class weight before a given class is considered the result of the classification. The margin can be drawn as the lines equidistant to the classification boundary at distance M.

For example, if we have three classes, the classification value is the class with the maximal weight w_i. Since there are 3 classes, the maximal weight will need to be bigger than one-third (0.3333...). If we now introduce a margin M = 0.1, even though there are three classes, the maximal value will need to be bigger than $\frac{1}{N} + M = \frac{1}{3} + 0.1 = 0.43333\ldots$ and if the maximal value is $w_1 = 0.38$, then we consider the classification outcome as "undefined."

REFERENCES

D. Afergan, E. M. Peck, E. T. Solovey, A. Jenkins, S. W. Hincks, E. T. Brown, R. Chang, and R. J. K. Jacob. 2014. Dynamic difficulty using brain metrics of workload. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'14)*. ACM, New York, NY, 3797–3806.

B. Alsallakh, A. Hanbury, H. Hauser, S. Miksch, and A. Rauber. 2014. Visual methods for analyzing probabilistic classification data. *IEEE Transactions on Visualization and Computer Graphics*, 20, 12.

S. Amiri, A. Rabbi, L. Azinfar, and R. Fazel-Rezai. 2013. A Review of P300, SSVEP, and Hybrid P300/SSVEP Brain Computer Interface Systems. In R. Fazel-Rezai (Ed.). *Brain-Computer Interface Systems—Recent Progress and Future Prospects*, InTech, Rijeka, Croatia.

A. Barachant, S. Bonnet, M. Congedo, and C. Jutten. 2012. Multiclass brain-computer interface classification by Riemannian geometry. *IEEE Transactions on Biomedical Engineering* 59, 4, 920–928.

A. Barbero and M. Grosse-Wentrup. 2010. Biased feedback in brain-computer interfaces. *Journal of Neuroengineering and Rehabilitation* 7, 1, 34.

B. Blankertz, F. Losch, M. Krauledat, G. Dornhege, G. Curio, and K.-R. Müller. 2008. The Berlin brain–computer interface: Accurate performance from first-session in BCI-naïve subjects. *IEEE Transactions on Biomedical Engineering* 55, 10, 2452–62.

B. Blankertz and C. Schäfer. 2002. Single trial detection of EEG error potentials: A tool for increasing BCI transmission rates. In *Artificial Neural Networks—ICANN 2002*. Springer, Berlin, 1137–1143.

W. R. Boot, A. F. Kramer, D. J. Simons, M. Fabiani, and G. Gratton. 2008. The effects of video game playing on attention, memory, and executive control. *Acta Psychologica* 129, 3, 387–398.

D. P. Bos, M. Obbink, A. Nijholt, G. Hakvoort, and M. Christian. 2010. Towards multiplayer BCI games. In *BioS-Play: Workshop on Multiuser and Social Biosignal Adaptive Games and Playful Applications*. 1–4.

C. Cajochen, K. Kräuchi, M.-A. von Arx, D. Möri, P. Graw, and A. Wirz-Justice. 1996. Daytime melatonin administration enhances sleepiness and theta/alpha activity in the waking EEG. *Neuroscience Letters* 207, 3, 209–213.

H. Cecotti, I. Volosyak, and A. Gräser. 2010. Reliable visual stimuli on LCD screens for SSVEP based BCI. In *18th European Signal Processing Conference (EUSIPCO'10)*. 919–923.

G. Chanel, J. Kronegg, D. Grandjean, and T. Pun. 2006. Emotion assessment: Arousal evaluation using EEGs and peripheral physiological signals. In B. A. Gunsel Jain, T. A. Murat, and B. Sankur (Eds). *Multimedia Content Representation, Classification and Security*. Springer, Berlin, 530–537.

M. Congedo, M. Goyat, N. Tarrin, G. Ionescu, L. Varnet, B. Rivet, R. Phlypo, N. Jrad, M. Acquadro, and C. Jutten. 2011. "Brain Invaders": A prototype of an open-source P300-based video game working with the OpenViBE platform. In *5th International BCI Conference*. 1–6.

J. B. F. Van Erp, F. Lotte, and M. Tangermann. 2012. Brain-computer interfaces: Beyond medical applications. *Computer*, 26–34.

J. A. Fails and D. R. Olsen Jr. 2003. Interactive machine learning. In *Proceedings of the 8th International Conference on Intelligent User Interfaces (IUI'03)*. ACM, New York, NY, 39–45.

S. Fairclough. 2011. Physiological computing: Interfacing with the human nervous system. In *Sensing Emotions.* 1–20.

M. Fatourechi, A. Bashashati, R. K. Ward, and G. E. Birch. 2007. EMG and EOG artifacts in brain computer interface systems: A survey. *Clinical Neurophysiology* 118, 3, 480–494.

R. Fiebrink, P. R. Cook, and D. Trueman. 2011. Human model evaluation in interactive supervised learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)*. ACM, New York, NY, 147–156.

E. Frank and M. Hall. 2003. Visualizing class probability estimators. In N. Lavrač, D. Gamberger, L. Todorovski, and H. Blockeel (Eds). *Knowledge Discovery in Databases: PKDD 2003 SE—17*. Lecture Notes in Computer Science. Springer Berlin, 168–179.

J. P. Gee. 2003. What Video Games Have to Teach Us About Learning and Literacy. *Comput. Entertain.*, 1, 1, 20.

D. Grimes, D. S. Tan, S. E. Hudson, P. Shenoy, and R. P. N. Rao. 2008. Feasibility and pragmatics of classifying working memory load with an Electroencephalograph. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'08)*. ACM, New York, NY, 835–844.

J. Gruzelier, T. Egner, and D. Vernon. 2006. Validating the efficacy of neurofeedback for optimising performance. *Progress in Brain Research* 159, 421–31.

C. Guger, C. Holzner, and C. Groenegress. 2009. Brain computer interface for virtual reality control. In *Proceedings of ESANN 2009*. 443–448.

H. Gürkök and A. Nijholt. 2012. Brain–computer interfaces for multimodal interaction: A survey and principles. *International Journal of Human-Computer Interaction*, 28, 5, 292–307.

C. R. Hema, M. P. Paulraj, S. Yaacob, A. H. Adom, and R. Nagarajan. 2009. Single trial motor imagery classification for a four state brain machine interface. In *5th International Colloquium on Signal Processing—Its Applications, 2009 (CSPA'09)*. 39–41.

L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, J. and P. Donoghue. 2006. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature* 442, 7099, 164–71.

A. Hyvärinen and E. Oja. 1997. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9, 1483–1492.

Z. Ìscan, Ö. Özkaya, and Z. Dokur. 2011. Classification of EEG in a steady state visual evoked potential based brain computer interface experiment. In *Proceedings of the 10th International Conference on Adaptive and Natural Computing Algorithms – Volume Part II (ICANNGA'11)*. Springer-Verlag, Berlin, 81–88.

I. Iturrate, R. Chavarriaga, L. Montesano, J. Minguez, and J. D. R. Millan. 2012. Latency correction of error potentials between different experiments reduces calibration time for single-trial classification. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference.* 3288–3291.

I. Iturrate, L. Montesano, and J. Minguez. 2010. Single trial recognition of error-related potentials during observation of robot operation. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. 4181–4184.

T. Iwata, K. Saito, N. Ueda, S. Stromsten, T. L. Griffiths, and J. B. Tenenbaum. 2005. Parametric embedding for class visualization. In L. K. Saul, Y. Weiss, and L. Bottou (Eds.). *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 617–624.

V. Kaiser, G. Bauernfeind, T. Kaufmann, and A. Kreilinger. 2011. Cortical effects of user learning in a motor-imagery BCI training. *International Journal of Bioelectromagnetism* 13, 2, 60–61.

C. Kapeller, C. Hintermüller, and C. Guger. 2012. Augmented control of an avatar using an SSVEP based BCI. In *Proceedings of the 3rd Augmented Human International Conference (AH'12)*. ACM, New York, NY, 27:1–27:2.

A. Kapoor, B. Lee, D. Tan, and E. Horvitz. 2010. Interactive optimization for steering machine classification. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'10)*. ACM, New York, NY, 1343.

S. P. Kelly, E. C. Lalor, R. B. Reilly, S. Member, and J. J. Foxe. 2005. Visual spatial attention tracking using high-density SSVEP data for independent brain–computer communication. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 13, 2, 172–178.

N. Kosmyna, F. Tarpin-Bernard, and B. Rivet. 2013. Towards a general architecture for a co-learning of brain computer interfaces. In *2013 6th International IEEE/EMBS Conference on Neural Engineering (NER)*. IEEE, 1054–1057.

N. Kosmyna and F. Tarpin-Bernard. 2013. Evaluation and comparison of a multimodal combination of BCI paradigms and eye tracking with affordable consumer-grade hardware in a gaming context. *IEEE Transactions on Computational Intelligence and AI in Games*, 5, 2, 150–154.

R. Kus, D. Valbuena, J. Zygierewicz, T. Malechka, A. Graeser, and P. Durka. 2012. Asynchronous BCI based on motor imagery with automated calibration and neurofeedback training. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 20, 6, 823–835.

A. Lécuyer, F. Lotte, R. Reilly, and R. Leeb. 2008. Brain-computer interfaces, virtual reality, and videogames. *Computer* 42, 10, 66–72.

J. Legény, R. Viciana Abad, and A. Lécuyer. 2013. Towards contextual SSVEP-based BCI controller: Smart activation of stimuli and controls weighting. *IEEE Transactions on Computational Intelligence and AI in Games* 5, 2, 111–116.

P. Lidberg. 2011. Barycentric and Wachspress coordinates in two dimensions: Theory and implementation for shape transformations. Uppsala University, UUDM Project Report.

A. Llera, V. Gomez, and H. J. Kappen. 2012. Adaptive classification on brain-computer interfaces using reinforcement signals. *Neural Computation* 24, 11, 2900–2923.

F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi. 2007. A review of classification algorithms for EEG-based brain-computer interfaces. *Journal of Neural Engineering* 4, 2, R1–R13.

F. Lotte, F. Larrue, and C. Mühl. 2013. Flaws in current human training protocols for spontaneous brain-computer interfaces: Lessons learned from instructional design. *Frontiers in Human Neuroscience* 7 (September), 568.

M. A. Migut, M. Worring, and C. J. Veenman. 2013. Visualizing multi-dimensional decision boundaries in 2D. *Data Mining and Knowledge Discovery*, 1–23.

J. D. R. Millán and J. Mouriño. 2003. Asynchronous BCI and local neural classifiers: An overview of the Adaptive Brain Interface project. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 11, 2, 159–161.

J. Müller-Gerking, G. Pfurtscheller, and H. Flyvbjerg. 1999. Designing optimal spatial filters for single-trial EEG classification in a movement task. *Clinical Neurophysiology* 110, 787–798.

G. Müller-Putz and R. Scherer. 2008. Better than random? A closer look on BCI results. *International Journal of Bioelectromagnetism* 10, 1, 52–55.

C. Neuper, G. R. Müller-Putz, R. Scherer, and G. Pfurtscheller. 2006. Motor imagery and EEG-based control of spelling devices and neuroprostheses. In C. Neuper and W. Klimesch (Eds.). *Event-Related Dynamics of Brain Oscillations*. Progress in Brain Research. Elsevier, 393–409.

L. F. Nicolas-Alonso and J. Gomez-Gil. 2012. Brain computer interfaces, a review. *Sensors (Basel, Switzerland)*, 12, 2, 1211–79.

K. Nigam and R. Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the 9th International Conference on Information and Knowledge Management (CIKM'00)*. ACM, New York, NY, 86–93.

A. Nijholt, B. Reuderink, and D. Oude Bos. 2009a. Turning shortcomings into challenges: brain-computer interfaces for games. In A. Nijholt, D. Reidsma, and H. Hondorp (Eds.). *Intelligent Technologies for Interactive Entertainment SE—15*. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer, Berlin, 153–168.

A. Nijholt, B. Reuderink, and D. Oude Bos. 2009b. Turning shortcomings into challenges: brain-computer interfaces for games. A. Nijholt, D. Reidsma, and H. Hondorp (Eds.). *Intelligent Technologies for Interactive Entertainment*, 1, 153–168.

A. Nooh, J. Yunus, and S. Daud. 2011. A review of asynchronous electroencephalogram-based brain computer interface systems. *International Conference on Biomedical Engineering and Technology* 11, 55–59.

S. Noorzadeh, B. Rivet, and C. Jutten. 2014. 2D for brain-computer interfaces: Two 3D extensions of the P300-speller. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'14)*. 2–7.

G. Pfurtscheller and F. H. Lopes da Silva. 1999. Event-related EEG/MEG synchronization and desynchronization: basic principles. *Clinical Neurophysiology*, 110, 11, 1842–1857.

M. F. Pike, H. A. Maior, M. Porcheron, S. C. Sharples, and M. L. Wilson. 2014. Measuring the effect of think aloud protocols on workload using fNIRS. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'14)*. ACM, New York, NY, 3807–3816.

A. Royer, A. Doud, M. Rose, and B. He. 2010. EEG control of a virtual helicopter in 3-dimensional space using intelligent control strategies. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 18, 6, 581–589.

R. Scherer, F. Lee, A. Schlogl, R. Leeb, H. Bischof, and G. Pfurtscheller. 2008. Toward self-paced brain–computer communication: navigation through virtual worlds. *IEEE Transactions on Biomedical Engineering* 55, 2, 675–682.

E. Schmidt, W. Kincses, and M. Schrauf. 2007. Assessing driver's vigilance state during monotonous driving. In *4th International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design*. 138–145.

C. Seifert and E. Lex. 2009. A novel visualization approach for data-mining-related classification. In *2009 13th International Conference on Information Visualisation*. 490–495.

V. J. Shute. 2008. Focus on Formative Feedback. *Review of Educational Research* 78, 1, 153–189.

Y. Sung, K. Cho, and K. Um. 2012. A development architecture for serious games using BCI (brain computer interface) sensors. *Sensors (Basel, Switzerland)* 12, 11, 15671–88.

J. Talbot, B. Lee, A. Kapoor, and D. Tan. 2009. EnsembleMatrix: Interactive visualization to support machine learning with multiple classifiers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'09)*. 1283–1292.

D. S. Tan and A. Nijholt (Eds.). 2010. *Brain-Computer Interfaces—Applying our Minds to Human-Computer Interaction*. Springer, New York, NY.

E. Thomas, M. Dyson, and M. Clerc. 2013. An analysis of performance evaluation for motor-imagery based BCI. *Journal of Neural Engineering*, 10, 3, 031001.

C. Vi and S. Subramanian. 2012. Detecting error-related negativity for interaction design. In *SIGCHI Conference on Human Factors in Computing Systems (CHI'12)*, 493.

C. T. Vi, I. Jamil, D. Coyle, and S. Subramanian. 2014. Error related negativity in observing interactive tasks. In *SIGCHI Conference on Human Factors in Computing Systems (CHI'14)*. ACM, New York, NY, 3787–3796.

C. Vidaurre, C. Sannelli, K.-R. Müller, and B. Blankertz. 2010. Machine-learning based co-adaptive calibration: A perspective to fight BCI illiteracy. In M. Graña Romay, E. Corchado, and M. T. Garcia Sebastian (Eds.). *Hybrid Artificial Intelligence Systems SE—50*. Lecture Notes in Computer Science. Springer, Berlin, 413–420.

Q. Wang, O. Sourina, and M. K. Nguyen. 2010. EEG-based "serious" games design for medical applications. In *2010 International Conference on Cyberworlds.* 270–276.

T. Wang, J. Deng, and B. He. 2004. Classifying EEG-based motor imagery tasks by means of time–frequency synthesized spatial patterns. *Clinical Neurophysiology* 115, 12, 2744–2753.

C. Wei, Y. Lin, Y. Wang, Y. Wang, T. Jung, and S. Member. 2013. Detection of steady-state visual-evoked potential using differential canonical correlation analysis. In *6th International IEEE/EMBS Conference on Neural Engineering (NER'13)*. 57–60.

J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. 2002. Brain-computer interfaces for communication and control. *Clinical Neurophysiology* 113, 6, 767–791.

E. Yin, Z. Zhou, J. Jiang, F. Chen, Y. Liu, and D. Hu. 2013. A novel hybrid BCI speller based on the incorporation of SSVEP into the P300 paradigm. *Journal of Neural Engineering*, 10, 2, 026012.

X. Zhu, B. Gibson, and T. Rogers. 2011. Co-training as a human collaboration policy. *AAAI*, 20(Nosofsky 1986).

Received July 2014; revised January 2015; accepted January 2015