METHODS ARTICLE
published: 24 November 2011
doi: 10.3389/fninf.2011.00030

Tools for brain-computer interaction: a general concept for a hybrid BCI

Gernot R. Müller-Putz[1]*, Christian Breitwieser[1], Febo Cincotti[2], Robert Leeb[3], Martijn Schreuder[4], Francesco Leotta[2], Michele Tavella[3], Luigi Bianchi[2], Alex Kreilinger[1], Andrew Ramsay[5], Martin Rohm[6], Max Sagebaum[4], Luca Tonin[3], Christa Neuper[1] and José del. R. Millán[3]

[1] Laboratory of Brain-Computer Interfaces, Institute for Knowledge Discovery, Graz University of Technology, Graz, Austria
[2] Neuroelectrical Imaging and Brain-Computer Interfaces Laboratory, Fondazione S. Lucia, Rome, Italy
[3] Defitech Chair in Non-Invasive Brain-Machine Interface, Center for Neuroprosthetics, School of Engineering, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
[4] Machine Learning Laboratory, Berlin Institute of Technology, Berlin, Germany
[5] Department of Computing Science, University of Glasgow, Glasgow, UK
[6] Department Orthopädie, Unfallchirurgie und Paraplegiologie, Experimentelle Paraplegiologie – Neurorehabilitation, Orthopaedic Hospital of Heidelberg University, Heidelberg, Germany

Edited by:
Friedrich T. Sommer, University of California at Berkeley, USA

Reviewed by:
Xin Wang, The Salk Institute for Biological Studies, USA
Daniel Llewellyn Rathbun, Eberhard-Karls University Tuebingen, Germany

***Correspondence:**
Gernot R. Müller-Putz, Laboratory of Brain-Computer Interfaces, Institute for Knowledge Discovery, Graz University of Technology, Krenngasse 37, Graz 8010, Austria.
e-mail: gernot.mueller@tugraz.at

The aim of this work is to present the development of a hybrid Brain-Computer Interface (hBCI) which combines existing input devices with a BCI. Thereby, the BCI should be available if the user wishes to extend the types of inputs available to an assistive technology system, but the user can also choose not to use the BCI at all; the BCI is active in the background. The hBCI might decide on the one hand which input channel(s) offer the most reliable signal(s) and switch between input channels to improve information transfer rate, usability, or other factors, or on the other hand fuse various input channels. One major goal therefore is to bring the BCI technology to a level where it can be used in a maximum number of scenarios in a simple way. To achieve this, it is of great importance that the hBCI is able to operate reliably for long periods, recognizing and adapting to changes as it does so. This goal is only possible if many different subsystems in the hBCI can work together. Since one research institute alone cannot provide such different functionality, collaboration between institutes is necessary. To allow for such a collaboration, a new concept and common software framework is introduced. It consists of four interfaces connecting the classical BCI modules: signal acquisition, preprocessing, feature extraction, classification, and the application. But it provides also the concept of fusion and shared control. In a proof of concept, the functionality of the proposed system was demonstrated.

Keywords: brain-computer interface, hybrid BCI, electroencephalogram, open-source, common architecture

1. INTRODUCTION

Persons having severe motor disabilities for various reasons can use a wide range of assistive devices (ADs) for managing their daily needs as well as using them for communication and entertainment purposes. The set of ADs ranges from simple switches connected to a remote controller to complex sensors (e.g., mouth mouse) attached to a computer and to eye tracking systems. All of these systems work very well after being adjusted individually for each person. However, there are still situations where the systems do not work properly, e.g., in the case of fatigue of residual muscles. In such a case, a Brain-Computer Interface (BCI) might be a good option, using brain signals (most likely the electroencephalogram, EEG) for control without the need for movement. A "brain-computer interface is a communication system that does not depend on the brain's normal output pathways of peripheral nerves and muscles" (Wolpaw et al., 2000). It therefore establishes a direct connection between the human brain and a computer (Wolpaw et al., 2002), thus providing an additional communication channel. A BCI itself contains an acquisition unit for brain signals, a signal processing chain (preprocessing, feature extraction, classification) and an application interface, driving the application and providing feedback to the user. For individuals suffering from severe palsy caused by muscle dystrophy, amyotrophic lateral sclerosis (ALS), or brainstem stroke, such a BCI constitutes a possible way to communicate with the environment (Birbaumer et al., 1999; Nijboer et al., 2008; Kübler et al., 2009). BCIs can also be used to control neuroprostheses in patients suffering from a high spinal cord injury (SCI), for example by using Functional Electrical Stimulation (FES) for grasp restoration (Müller-Putz et al., 2005).

The aim of this work performed by the TOBI (Tools for Brain-Computer Interaction[1]) developer group is to develop a hybrid BCI (hBCI) which combines existing input devices with a BCI. The BCI should be available if the user wishes to extend the types of inputs available to an assistive technology system, but

[1] http://www.tobi-project.org

the user can also choose not to use the BCI at all. Here it is of importance that the BCI itself is active, which means online EEG analysis is performed all the time. On the one hand the hBCI might decide which input channel(s) offer the most reliable signal(s) and switch between input channels to improve information transfer rate, usability, or other factors. On the other hand the hBCI could be used to fuse various input channels.

The idea of having a hybrid solution is not entirely new. In a recent work by Pfurtscheller et al. (2010), an overview of existing hybrid BCIs is given. These BCIs have in common that they all combine a BCI with another BCI or a BCI with another biosignal, which is different from our approach.

One major goal is to bring the BCI technology to a level where it can be used in a maximum number of scenarios in a simple way. To achieve this, the hBCI must be able to operate reliably for long periods, recognizing and adapting to changes. Reaching this goal requires that many different subsystems in the hBCI are able to work together. Examples are standard BCI processing, post-processing (e.g., error potentials Schalk et al., 2000; Ferrez and Millán, 2008), mental state recognition, artifact detection, adaptation of classifiers, surveillance of signal quality (including EEG signals and those from additional input devices). All of this functionality cannot be provided by one research institution alone. In consequence, a collaboration between institutions is necessary to achieve the goal of realizing the hBCI.

Achieving such a broad collaboration is not a simple task. Different research institutions and laboratories are using different BCI systems. Some of those systems are freely available, others are customized systems just used inside a single lab. But all of those systems provide the core functionality needed to form a BCI system. As shown in Mason and Birch's work (Mason and Birch, 2003), BCI systems can be separated into individual modules interconnected with different interfaces. Many current BCI systems are built in this modular fashion providing an opportunity to introduce a standard to interconnect different modules thus enhancing interchangeability and facilitating cooperation. Such a standardization attempt is furthermore in accordance with the findings from Brunner et al. (2011), demonstrating a lack of standardization in basic and clinical BCI research. Standardization would not merely simplify collaboration; it would also bring BCI systems closer to a potential end-user market.

2. STATE OF THE ART BCI PLATFORMS

To establish some kind of common ground, a common software framework was investigated. Therefore current BCI software solutions such as BCI2000 (Schalk et al., 2004), BF++ (Bianchi et al., 2003), FieldTrip[2], rtsBCI (2004–2008), Open-ViBE (Arrouet et al., 2005), and the recently published xBCI (Susila et al., 2010) were analyzed in more detail.

- BCI2000, based on Borland C++, consists of four types of modules: operator, source, signal processing, and application. These modules can be combined into an online BCI system. Modules are implemented to run as stand-alone applications and exchange data via TCP/IP, even when running on the same PC.
- The Body Language Framework (BF++) is a real-time set of C++ software libraries which implements the BCI dynamic model described in Quitadamo et al. (2008). With BF++ one can implement, analyze, and optimize complete systems. The NPXLab Suite, which is part of BF++, can perform several powerful analyses on many physiological signals (ICA, spectral analysis, averaging, etc.).
- FieldTrip is a Matlab toolbox for analysis of EEG and MEG data. It is well-known for its offline analysis capabilities but it also supports online analysis and experimentation. BrainStream[3] provides an easy way to design experiments and reuse existing modules.
- rtsBCI is a toolbox for Matlab and Simulink for Windows. It is based on the real-time Windows target provided by Matlab. New modules can easily be developed graphically or with m/c-mex-files.
- Open-ViBE consists of many modules written in C++ for neuro-feedback and virtual reality. New modules can be added to the framework as plug-ins. EEG data is acquired with a data acquisition server and sent via TCP/IP.
- xBCI is a software framework to be used on multiple operating systems. It consists of several functional modules, such as acquisition, storage, mathematical operations, signal processing, network communication, data visualization, experimental protocol, and real-time feedback presentation. With the help of a GUI new systems can be developed.

All these systems have their advantages and disadvantages. However, many of the institutes involved in BCI research have their own development and research environments, e.g., Reading BCI (Portelli and Nasuto, 2008) or Strathclyde BCI (Valsan et al., 2009). They might not want to switch over to another system, but would still profit from the easy exchange of existing processing modules blue with other institutions whithout extensive programming and adaptation effort. It is not the goal of this work to establish an entirely new BCI software because there are already a myriad of software solutions. Instead, the goal is to provide a common architecture based on a standardized data acquisition module for EEG and other signals and standardized interfaces. These interfaces allow for an easy connection of any modules (e.g., classifiers, post-processors, ...) to existing software without requiring a special environment as is the case with existing solutions.

3. CONCEPT OF THE HYBRID BCI

The proposed design (see **Figure 1**) is based on the model already suggested by Mason and Birch (2003). The model consists of modules like data acquisition, different signal processing modules, multimodal fusion, and shared control with the application interface. These modules are interconnected with different interfaces. The signal flow can be explained in the following way: signals [either from EEG, other biosignals like electromyogram (EMG), or from assistive devices] are acquired via different hardware and hardware interfaces (USB port, data acquisition cards, ...) and provided for further use. This is realized with the SignalServer

[2]http://www.ru.nl/neuroimaging/fieldtrip

[3]http://www.brainstream.nu

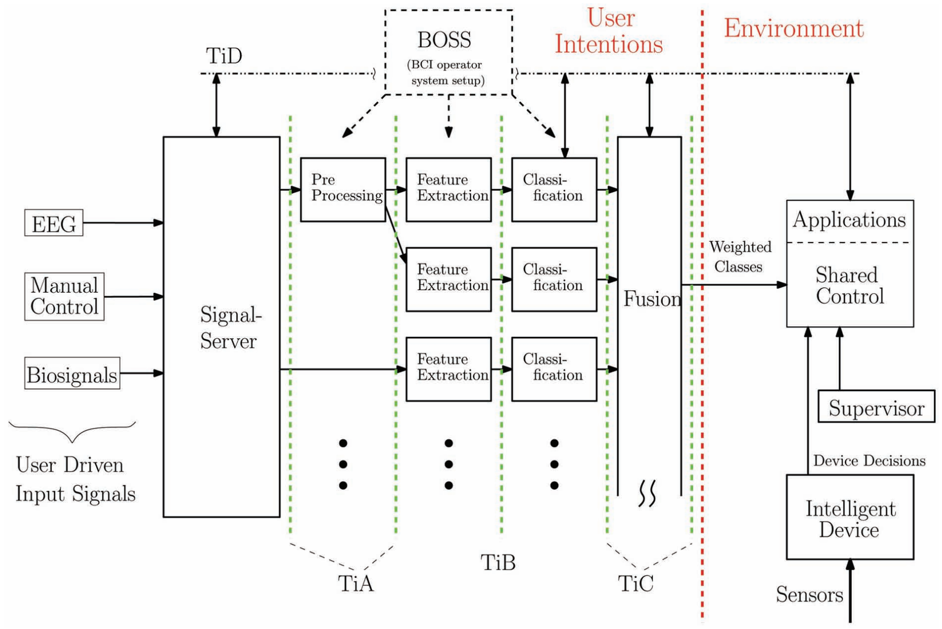

FIGURE 1 | Design sketch of the hBCI. Raw control signals from the user (EEG, assistive devices, other biosignals) are collected by the SignalServer which provides those signals in a standardized way to whatever client may connect to it. This connection is realized via TiA. Entering the classic signal processing pipeline as well as other signal processing modules several single control signals are generated. Through TiC the fusion module receives those signals and generates one control signal. Sent via TiC to the shared control the application receives control commands. The application itself is equipped with sensors and intelligence so that the user environment can support the user's control signal. Via TiD events and messages can be sent back to the SignalServer to keep information from, e.g., classifiers within the processing pipeline. A BCI operator system setup (BOSS) module will help with configuration and starting different modules. A supervisor always has the possibility to overrule decisions and perform necessary actions.

which is explained in Section 1. From here, data can be processed (e.g., in a common BCI signal processing chain, mental state monitoring, error potential recognition, EMG artifact detection, switch signal quality check, . . .) and different control signals are fed to the multimodal fusion module. The task of this module is to decide which control signal (or classification result) is best suited to controlling the application. This means that, having (i) a BCI-based on, e.g., sensorimotor rhythm (SMR) for left/right hand motor imagery (MI) classification (for more details see exemplarily Pfurtscheller and Neuper, 2001; Wolpaw et al., 2002; Millán et al., 2010), (ii) an artifact detection algorithm, and (iii) a control signal from a joystick, the fusion decides which control signal is used to control the application. The BCI output can be blocked if artifacts are found in the EEG, in which case the joystick is used instead. Decisions can be based on static or dynamic rules. The final control signal then goes on to the shared control block. This module also receives information from the environment and helps to control the application in the most appropriate way.

The main question here is how the subject might control a complex application by means of an uncertain channel such as a BCI. An answer to this fundamental issue is the well-known shared control approach (Flemisch et al., 2003; Vanhooydonck et al., 2003; Goodrich et al., 2006). The cooperation between a human and an intelligent device allows the subject to focus their attention on a final target and ignore low-level details related to the execution of an action. For instance, in the case of a BCI-based telepresence robot the introduction of the shared control helps the user to reach the target in less time with a lower number of commands. In this case the role of shared control is to take care of the low-level details related to the navigation task (e.g., obstacle detection and avoidance; Wolpaw, 2007).

3.1. INTERFACES

The interfaces are the most important part of the hBCI, facilitating a standardized communication between different blocks independently from the chosen platform and software language. Currently, many BCI laboratories have their own techniques to perform data processing. Therefore, common methods to exchange data between different components have to be established. However, a specification which only describes the format of the data to be exchanged between components is not adequate in this case. To achieve true modularity, ways to transmit and exchange data must be defined as well. Three types of viable data exchange methods exist: (I) exchange of data within the same programming language, (II) exchange within the same computer but between different programming languages, and (III) exchange between different computers (see **Figure 2**).

- I: The fastest and most efficient way to exchange data is within the same program. Because there are no restrictions, there is also no compatibility with other components. This data exchange method can be a Matlab/Simulink connection as well as a C++ system call. If preprocessing, feature extraction, and classification are connected with method I, then only the interface

definitions for TiA (Tobi interface A), TiC, and TiD have to be obeyed.
- II: A fast way for data exchange on local systems can be achieved by using shared memory. Data to be shared is stored in a defined memory region and other processes are able to read and write from and into that memory.
- III: This method of data exchange is introduced to allow distributed processing. Data is sent over the network using TCP or UDP. As this data exchange method also supports local data exchange, it has been implemented first. For interfaces TiA and TiB raw data transfer methods will be used to reduce network load.

The definition of all interfaces between different components is a critical step toward the goal of being able to exchange modules without any modifications to other parts of the hBCI system.

Figure 2 shows the different interfaces TiA–TiD with different communication layers. At present, TiA, TiC, and TiD have been implemented using layer III communication (network).

3.1.1. Interface A – TiA

TiA represents an interface developed to transmit raw biosignals (e.g., EEG) and data from simple user driven devices like buttons and joysticks. It is a client-server based system with one server and the possibility to attach multiple clients. The client-server communication is divided into (i) a control connection and (ii) a raw data transmission stream. The control connection is only used to transmit meta information from the server to the client and to exchange control commands, both using XML messages. The raw data connection is unidirectional from the server to the client and is used to transmit acquired data in a binary form as a continuous data stream. Up to now, the whole client-server communication is realized via network sockets.

3.1.2. Interface B – TiB

At present, there is no existing specification of the standardized interface TiB, as some BCI systems have a very tight coupling between feature extraction and classification (see **Figure 2**). Furthermore, TiB was not classified as urgent at this development stage, as most BCI labs are using their own unique processing chains from preprocessing to classification and only feed data into this chain and get results out of it.

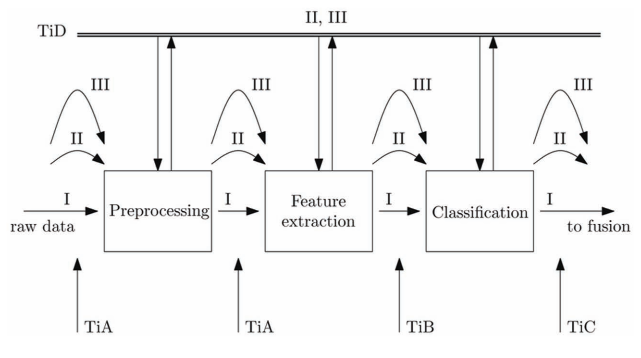

FIGURE 2 | Interfaces of the hybrid BCI with different layers. (I) Exchange of data within the same program. (II) Shared memory data exchange. (III) Network (UDP, TCP) connection.

3.1.3. Interface C – TiC

Interface C was designed to connect the classifier-like modules of the hBCI to multimodal fusion (Section 3) or other higher-level modules. The classifier outputs are pairs of values and labels. Considering the most commonly used classifier in the literature (for an overview see Wolpaw et al., 2002), following value types are used:

- a vector of distances (e.g., linear discriminant analysis for motor imagery)
- a vector of probabilities (e.g., bayesian classifier for motor imagery)
- a scalar (e.g., regression for motor imagery)
- binary selections (e.g., step wise linear discriminant analysis for P300 signals)

This interface handles only high-level data that is routed through different modules at reduced speed, typically below 50 Hz, and uses a general, platform independent, and high-level communication based on XML messages over a TCP/IP network. As data complexity at TiC is higher and it should be possible to add new classification results easily, this communication is realized in an XML oriented packet style.

3.1.4. Interface D – TiD

During BCI experiments different "events" like an acoustic beep or a cross on a computer monitor might occur. Furthermore, signals from e.g., an external trigger line might indicate external triggers, also called "markers." Accurate timing information of markers and events and its distribution are crucial within BCI systems. Therefore TiD was developed, fulfilling these requirements. TiD is based on a bus system. Clients can attach themselves to this bus if they have to deal with events. Otherwise, they can ignore these events. To ensure proper timing with the acquired data, every TiD message is equipped with a timestamp. To enhance compatibility to existing systems, message families were introduced (e.g., gdf event, BCI2000 event…). In **Figures 1** and **2**, the TiD bus is visible.

BCIs are based on time critical information, therefore, an interface has to be provided to ensure the current flexibility of today's BCI system. Markers and events are handled with interface TiD. Every TiD message is equipped with a timestamp to allow a proper association between an event and the processed data. To enhance compatibility to existing systems, message groups are planned to be used (e.g., gdf event, BCI2000 event…). In **Figures 1** and **2**, the TiD bus is visible.

3.2. MODULES

This section explains all modules shown in **Figure 1** and connected through the interfaces introduced in the previous section.

3.2.1. SignalServer

To meet the requirements for building a hBCI-based on different kinds of inputs, e.g., EEG and a joystick signal, a data acquisition system called "SignalServer" was implemented (Breitwieser et al., 2011). The SignalServer is able to acquire multirate data from different hardware devices simultaneously. It utilizes TiA to deliver the acquired data in a standardized way to facilitate the possibility of interconnecting different BCI systems, independent of the data acquisition method normally used.

Figure 3 shows the main functionality principle of the SignalServer. Different clients can connect to the server at runtime to get data from different hardware devices.

3.2.2. *Signal processing chain*

All acquired signals can be fed to specific signal processing chains. For example, EEG signals get preprocessed, features can be extracted and classified resulting in a BCI control signal. Also mental monitoring and quality check modules can be included here. But also all non-EEG signals, which are used for the hBCI control can be manipulated here. By using interface TiC, all resulting signals are forwarded to the fusion.

3.2.3. *Fusion*

Generally, the fusion module receives classification information (probabilities, class labels, regression values, . . .) from a set of processing modules. Several BCI classifiers or even several different BCIs (e.g., motor imagery, P300, error potential, . . .) together with the estimation of other biosignals (like EMG, . . .) and even assistive devices (like switches) can be merged. But also, in the most simple case, fusion does not exist and the classifier output goes directly to the application.

In **Figure 4**, an example with two BCI classifiers and one EMG classifier is given. In this case the inputs are the probabilities p(x) of all the preceding classification modules and the fused probability p(x) is the output. The output of the fusion is –like the input– based on interface TiC and is used as input to the shared control or to control the BCI feedback (if no shared control is used).

The simplest fusion would be a switch between the input channels with weights *w* at 0 and 1. First results for brain and muscular activities were already reported (Leeb et al., 2010). In the implementation shown in **Figure 4**, a hierarchical approach with static rules is given, which we have already tested with two fusion techniques in (Leeb et al., 2011). The first approach used equally balanced weights between the classifiers. The second one applied the naïve Bayesian fusion approach (Rogova and Nimier, 2004). There, we could restrict the fusion to the same timing (all classifiers were running at the same sampling rate) and resolution of the input space (both were producing probabilities).

Multimodal fusion techniques facilitate the combination of brain control with other residual motor control signals and can thereby achieve better and more reliable performances. Currently, a static approach is used, but the weights could also update dynamically based on the reliability of these input channels, or the confidence/certainty the system has on its outputs. Generally, these weights can be estimated from supervision signals such as cognitive mental states (e.g., fatigue, error potentials) and physiological parameters (e.g., muscular fatigue). Another way to derive the weights is to analyze the performance of the individual channels in achieving the task (e.g., stability over time, influence of noise, . . .).

3.2.4. *Shared control*

The role of the shared control module is to contextualize the user's intents in the current environment in order to support him/her in the control of an external application.

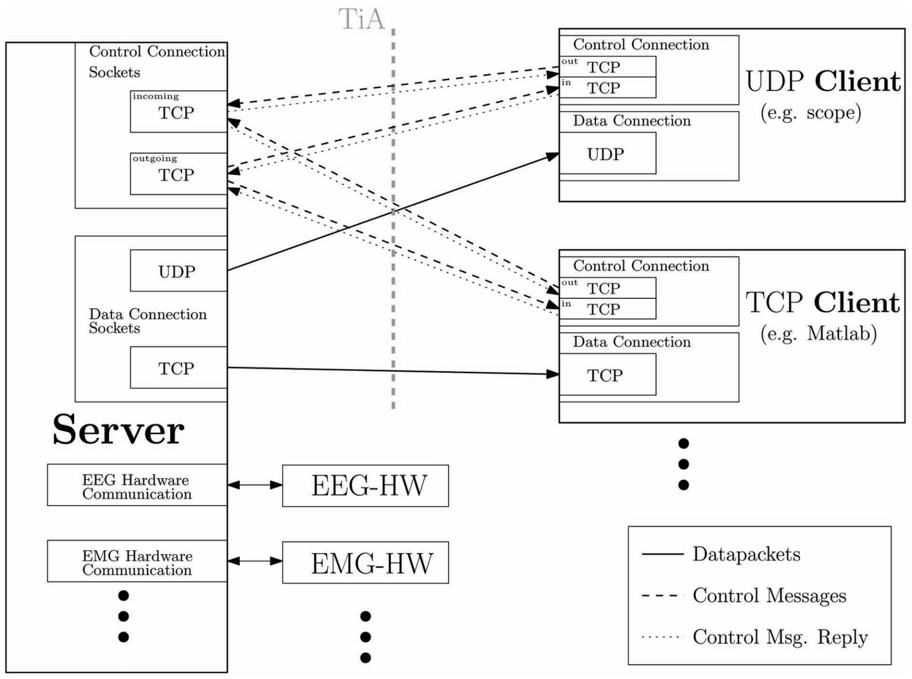

FIGURE 3 | Working scheme of the data acquisition server. Clients can connect to the server and will be configured through control messages, sent in XML format. After configuration, acquired data will be sent to every attached client using TCP or UDP, dependent on the client's requirements.

To do that, the first task of the shared control is to manage all the high-level information coming from the user and the application (environment related messages). For the user-shared control connection, the message's format is defined by TiC. For the application-shared control interface, the format is strictly application dependent. Generally, these messages are named *events*.

Secondly, the shared control has to compute the events received in order to send the final command to the application. Static rules inside the module (application dependent) are in charge of this task. However, the same concept of shared control may be used for different kinds of applications (communication, neuroprosthetic control).

The most natural example for an explanation is the control of a mobile platform (telepresence). In this case, the role of shared control is to help the user in the navigation task by taking care of all the low-level decisions (i.e., obstacle avoidance). **Figure 5** shows the general concept behind the implementation of shared control for this specific device. Here, we can define the following event generators: BCI driven by the user's intent and sensors on the device that provide environmental information. The shared control collects all these events defining the device's current status in the environment. The protocol is totally asynchronous and the computation of the new event-based situation is triggered by the arrival of new events. As soon as a new command is decided, the shared control sends it to the device that executes the action.

In this way, the shared control allows the user to focus his/her attention on the final destination and ignore low-level problems related to the navigation task.

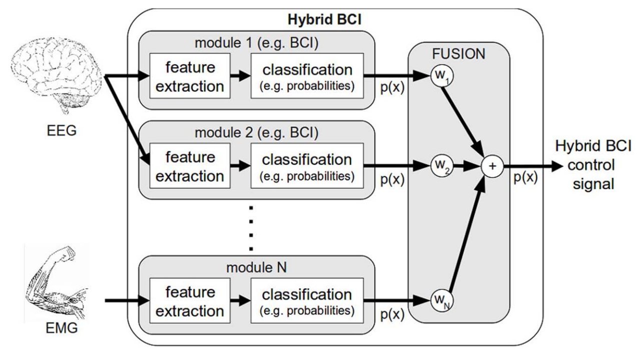

FIGURE 4 | Fusion principle of several BCI classifiers and one muscular channel to one hBCI control signal. p(x), probability; w1, w2, w3, weight of the fusion rule.

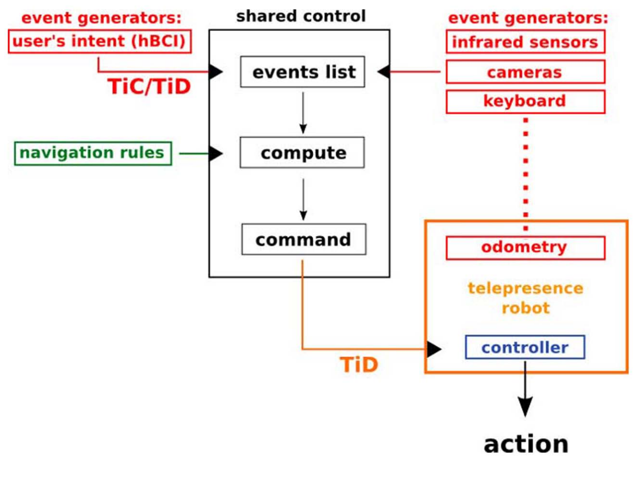

FIGURE 5 | Schema of shared control module. This figure shows an example of shared control for a mobile device. However, the same schema can be used for different kinds of applications/devices.

3.3. FEEDBACK

Feedback is very important in brain-computer interfacing. Therefore, research institutions tend to put a great deal of effort into developing their own, often very specific, feedback systems. Often, feedback is provided visually with the help of a computer screen, or by the application itself (e.g., robotic arm). Feedback can also be auditory via beep tones or music tunes, or even tactile.

There are many different programming languages currently used for the realization of those specific feedback systems. Considering these circumstances, we incorporated the pythonic feedback framework Pyff[4] (Venthur et al., 2010). It allows for easy and quick implementation of feedback applications, particularly for BCI and other biofeedback applications. It introduces a standardized communication layer between the application and the data source (BCI), which enables the experimenter to switch between different feedbacks using the same BCI setup. It supports input data in TiC format, thereby providing access to all the packaged feedback applications.

3.4. PROOF OF CONCEPT

The main goal of the proof of concept was to show the modularity of the hBCI system and to show how easily different components could be combined using interfaces TiA–TiD.

At the time of the demonstration experiment, seven different laptops were connected based on this concept and each hosted one special module of the hBCI. The novelty here was that all modules were from different BCI labs and were interconnected through the new interfaces. The remaining interfaces TiB and TiD, which were not defined before the proof of concept, were not used because the BCI chains were not split up. The schema of this setup can be seen in **Figure 6**. The modules/computers in use were as follows:

- EEG amplifier: a g.USBamp (g.tec, Graz, Austria) was used to record the EEG from the subject. Only one Laplacian channel overlying the vertex (Cz) was used.
- SignalServer: here, the settings for the amplifier were configured and data was acquired constantly. Data was provided for any number of clients that could connect to the server via TCP or UDP.
- SMR BCI: this core module of the hBCI system performed BCI-based on SMR and a linear discriminant analysis classifier was used to detect ERS (event related synchronization) after the imagination of feet movements, as, e.g., described in Pfurtscheller and Neuper (2001).

[4]http://www.bbci.de/pyff

- Artifact detection: here, a simple noise detection was applied to detect high amplitudes of EMG within the recorded EEG. Each detection of artifacts was sent to the following fusion module.
- Fusion: the fusion performed here was very simple. A detected artifact would inhibit the classifier output of the SMR BCI and thereby avoid unwanted commands caused by noisy data.
- Operator feedback (Pyff): the actual output of the classifier was permanently displayed with a liquid cursor (Kaufmann et al., 2011) implemented in Python. Only during periods of noisy data the feedback was inhibited and the feedback was on standby.
- Application and realistic feedback: the BCI was used to control the movement of an avatar's arm, visualized on a large screen. This arm could only be controlled with artifact-free BCI commands.

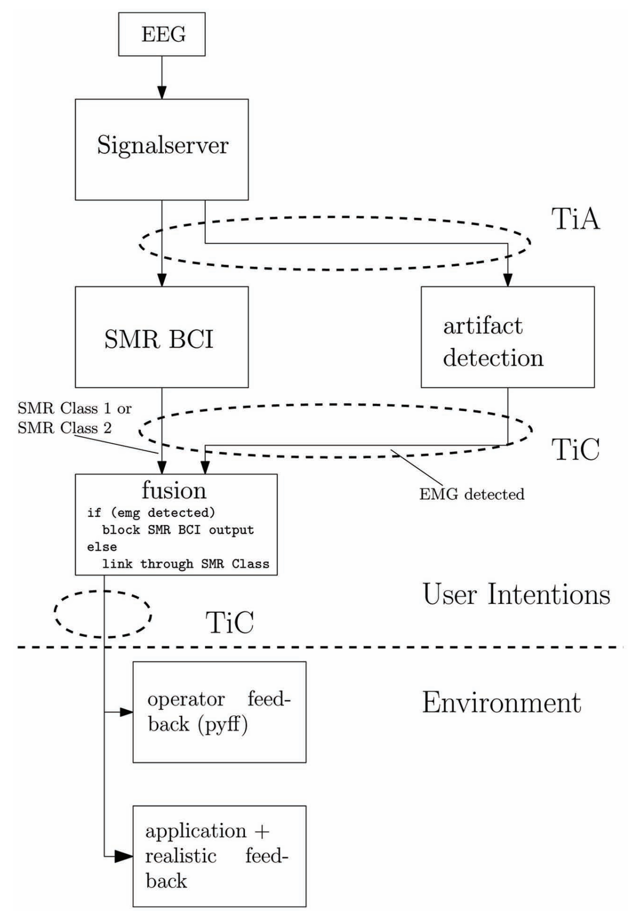

FIGURE 6 | Schema of the hybrid BCI demonstration setup. Each block indicates a computer, connected over a LAN connection using TiA and TiC.

Both types of feedback were connected to the system to show the possibility of attaching multiple clients.

4. RESULTS

4.1. SIGNALSERVER

The SignalServer is designed to support various data acquisition sources. A list of supported devices can be found in **Table 1**. Not every device mentioned in **Table 1** is supported on every platform due to limited driver availability. Generic devices are supported across all platforms.

Every implementation was tested in short and long-term runs with regards to possible memory leaks and stability. Both black and white box tests were performed. Where possible, single component tests were also carried out.

The implementation was stable in all of the applied tests as well as during longer runs of more than 12 h (for more details see Breitwieser et al., 2011).

4.2. INTERFACES: PLATFORM AND ENVIRONMENT OVERVIEW

Beside the SignalServer, also the implementations of three of the introduced interfaces, namely TiA, TiC, and TiD exist. In **Table 2** an overview of different platforms and environments for all interfaces is given. As all currently implemented interfaces' base code is written in C++ with platform independent libraries, porting to another platform is generally possible without elaborate code adjustments. Interface TiB is currently not implemented and used.

Windows XP and Ubuntu 9.04 were chosen as target platforms.

A first release of the SignalServer and the TiA can be downloaded from http://bci.tugraz.at.

Table 1 | List of devices supported by the SignalServer (modified from Breitwieser et al., 2011).

Name	Manufacturer	Type	Implementation	Testing
g.USBamp	g.tec (Guger Technologies, Graz, Austria)	Multipurpose biosignal DAQ	Done	Done
g.Mobilab	g.tec (Guger Technologies, Graz, Austria)	Mobile biosignal DAQ	Done	Done
Generic joysticks	Independent	Aperiodic user input	Done	Done
Software sine generator	–	Testing signals	Done	Done
Generic mouse	Independent	Aperiodic user input	Done	Done
BrainVision BrainAmp Series	Brain Products (Gilching, Germany)	Multipurpose biosignal DAQ	Done	Done
g.BSAmp	g.tec (Guger Technologies, Graz, Austria)	Multipurpose biosignal DAQ	Done	In progress
DAQ card	National Instruments (Austin, TX, USA)	Multi I/O card	Done	In progress
Generic keyboard	Independent	Aperiodic user input	In progress	Planned
NIRScout	NIRx Medical Tech., LLC. (Glen Head, NY, USA)	NIRS daq system	In progress	Planned
Adjustable EEG simulator	–	Testing signals	Done	Done
g.USBamp (Linux)	g.tec (Guger Technologies, Graz, Austria)	Multipurpose biosignal DAQ	Planned	Planned

Table 2 | Current implementation status of all interfaces regarding their environment and platform (srv, server; clt, client; Lx, Linux; Win, Windows).

		TiA		TiB		TiC		TiD	
		srv	clt	srv	clt	srv	clt	srv	clt
C++	Lx	Ok	Ok	–	–	Ok	Ok	Ok	Ok
	Win	Ok	Ok	–	–	Ok	Ok	Ok	Ok
Python	Lx	x	x	–	–	Ok	Ok	–	–
	Win	x	x	–	–	Ok	Ok	–	–
Matlab	Lx	x	Ok	–	–	Ok	Ok	Ok	Ok
	Win	x	Ok	–	–	Ok	Ok	Ok	Ok
Simulink	Lx	x	Ok	–	–	Ok	Ok	Ok	Ok
	Win	x	Ok	–	–	Ok	Ok	Ok	Ok

"Ok" means that functionality is available and is tested; "x" means that it is not available yet, but porting from another platform/environment is possible; and "–" means that there is no implementation work done yet.

4.3. PROOF OF CONCEPT

The demonstration was a success because the combination of all the different modules from different research labs was very simple and was accomplished after only a few minutes of preparation. Interface TiA worked without any problems and the clients could connect to the SignalServer easily. Interface TiC also worked successfully as the XML definition it used was already compatible with the different programs in the system. The basic version of fusion gave a first impression of active signal selection. In the future, more sophisticated rules, dealing with more inputs and more complicated inputs, have to be generated.

Figure 7 presents the running demonstration setup, where each laptop represents a block in **Figure 6**. A Movie S1 that shows the functionality of the system in action can be found in the Supplementary Material.

The complete system worked without any problems during the whole demonstration period, which took place over a period of more than an hour. All the components involved behaved stably throughout.

We want to highlight again that all modules were from different labs and not specially designed to be used together.

5. DISCUSSION AND CONCLUSION

This work introduces a general framework to establish a so-called hybrid BCI including a set of interface definitions. This new BCI architecture should easily enable researchers to include modules from other researchers in their own systems, independently of the particular operating systems and development languages they are using. This new concept also offers the possibility to include not only biosignals from one source (as usually EEG, EMG,...) but also from other input devices like standard mice and keyboards, and most important assistive devices.

Beside the introduction of the four interfaces the concept of the hBCI includes also two new important modules, namely fusion and shared control. Whereas fusion is necessary to form the control signal finally generated by the hBCI, the shared control uses this control signal as well as information from the application environment to improve the application control.

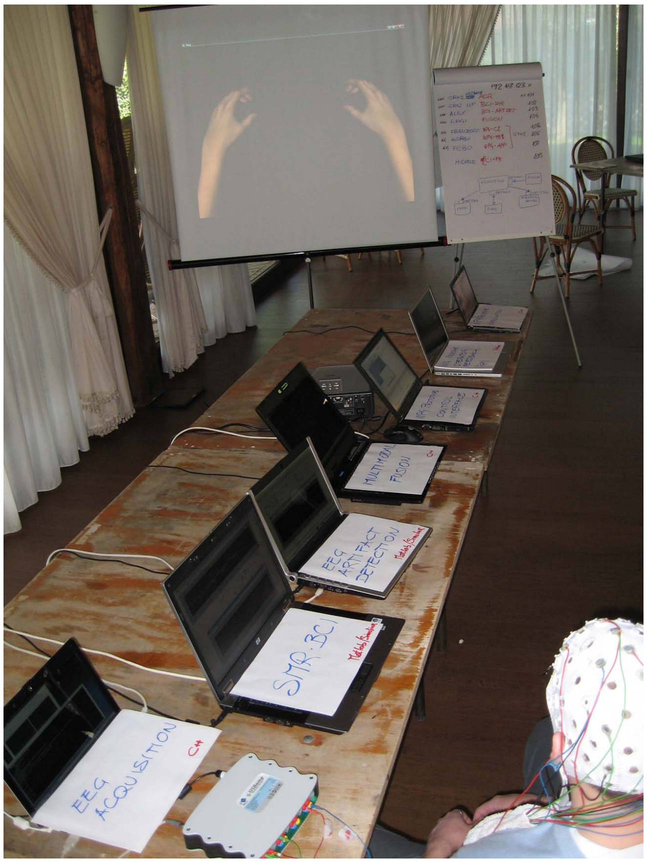

FIGURE 7 | This picture was taken during the demonstration of the proof of concept. Here, all connected computers, forming a demo hybrid BCI, are visible. The first was used for data acquisition, running the SignalServer, and the next two were used to build an SMR BCI and an artifact detection. Laptop number four was a very simple implementation of fusion. Laptops five and seven formed together the visual feedback, presented on the screen (control interface and feedback application). The sixth laptop showed an additional operator feedback which was not presented to the subject.

The proof of concept has shown that by applying the proposed interfaces a hBCI could be set up by using modules from different research labs. This is of very special importance, because all those modules were implemented in different languages and running on several operating systems.

Initial scientific papers using the general approach have recently appeared. Leeb et al. (2011) uses this principle of an hBCI to fuse EMG and EEG activity. The multimodal fusion approach yielded to a more accurate and stable control compared to a single signal use. Kreilinger et al. (in revision) uses the concept for a combined EEG and joystick control. The hBCI switches between input signals depending on the quality. It was shown that a combination of both input signals results in a longer application duration when artificial noises make the joystick signal unusable. In a recent paper by Tonin et al. (2011) a telepresence robot was controlled by applying the hBCI concept including fusion and shared control. Remarkably, the patients – novel to the task – achieved levels of performance similar to those of a group of healthy users who were familiar with the task. It is important to also mention that hBCI systems like those recently described in the review of Pfurtscheller et al. (2010), are all realizable with this hBCI concept.

5.1. FUTURE DIRECTIONS

Our work on the hBCI and the general framework forms the basis for a set of standards in BCI software. Our goal is to release the specification for all four interfaces, which would allow for interoperability of both in-house, open-source and closed-source BCI software packages. Our motivation is a wider and easier integration and collaboration of different labs involved in BCI research, as well as better hardware compatibility. In addition, a standardized BCI system could potentially facilitate the comparison of different systems and therefore also the results produced with these systems – making a first step toward standardized BCI metrics. Applying the principle of standardized interfaces used for interconnection is one step to bring current BCI technology closer to the end-user market. Furthermore, as mentioned in Brunner et al. (2011), the proposed framework would reduce the problem of standardization in basic and clinical BCI research.

5.2. EXPECTATIONS, LIMITATIONS, AND RESTRICTIONS

The hBCI presented within this work using standardized interfaces for information distribution provides a powerful way to build and interconnect BCI systems. However, it is not a BCI system itself. The modules forming a BCI, as shown in Mason and Birch's work (Mason and Birch, 2003), have to be implemented individually or could potentially be taken from another BCI system that supports the mentioned interfaces. Additionally, the success of the proposed framework is also coupled to a certain acceptance and distribution within the BCI field. Greater distribution obviously increases compatibility within BCI research; collaboration would become easier and research would be accelerated.

Furthermore, a growing community would potentially also increase the amount of contributions to the refinement of the standards and the currently written code libraries.

We expect, by providing this hBCI concept, which integrates common assistive devices with different types of BCI, mental state monitoring, and adaptation, users will be able to use their assistive system throughout the whole day with a high-level of accuracy, and BCI systems will become real assistive devices.

ACKNOWLEDGMENTS

This work is supported by the European ICT Programme Project TOBI FP7-224631. This paper only reflects the authors' views and funding agencies are not liable for any use that may be made of the information contained herein. Special thanks to Fabio Aloise (Fondazione S. Lucia) and Christoph Eibel (TU Graz) for valuable help in programming.

SUPPLEMENTARY MATERIAL

The Movie S1 for this article can be found online at http://www.frontiersin.org/Neuroinformatics/10.3389/fninf.2011.00030/abstract

REFERENCES

Arrouet, C., Congedo, M., Marvie, J., Lamarche, F., L'Ecuyer, A., and Arnaldi, B. (2005). Open-ViBE: a 3D platform for real-time neuroscience. *J. Neurother.* 9, 3–25.

Bianchi, L., Babiloni, F., Cincotti, F., Salinari, S., and Marciani, M. G. (2003). Introducing bf++: Ac++ framework for cognitive bio-feedback systems design. *Methods Inf. Med.* 42, 104–110.

Birbaumer, N., Ghanayim, N., Hinterberger, T., Iversen, I., Kotchoubey, B., Kübler, A., Perelmouter, J., Taub, E., and Flor, H. (1999). A spelling device for the paralysed. *Nature* 398, 297–298.

Breitwieser, C., Neuper, C., and Müller-Putz, G. R. (2011). Proposing standardized multirate biosignal transmission for brain-computer interfacing. *IEEE Trans. Biomed. Eng.* (in press).

Brunner, P., Bianchi, L., Guger, C., Cincotti, F., and Schalk, G. (2011). Current trends in hardware and software for brain–computer interfaces (BCIs). *J. Neural Eng.* 8, 025001.

Ferrez, P. W., and Millán, J. D. R. (2008). Error-related EEG potentials generated during simulated brain-computer interaction. *IEEE Trans. Biomed. Eng.* 55, 923–929.

Flemisch, O., Adams, A., Conway, S., Goodrich, K., Palmer, M., and Schutte, P. (2003). *The H-Metaphor as a Guideline for Vehicle Automation and Interaction.* Technical Report NASA/TM–2003-212672, Hampton, NASA.

Goodrich, K., Schutte, P., Flemisch, F., and Williams, R. (2006). "Application of the H-mode, a design and interaction concept for highly automated vehicles, to aircraft," in *Proceedings of IEEE Digital Avionics Systems Conference*, Portland, 1–13.

Kaufmann, T., Williamson, J., Hammer, E., Murray-Smith, R., and Kübler, A. (2011). Visually multimodal vs. classic unimodal feedback approach for SMR-BCIs: a comparison study. *Int. J. Bioelectromagn.* 13, 80–81.

Kübler, A., Furdea, A., Halder, S., Hammer, E., Nijboer, F., and Kotchoubey, B. (2009). A brain-computer interface controlled auditory event-related potential (P300) spelling system for locked-in patients. *Ann. N. Y. Acad. Sci.* 1157, 90–100.

Leeb, R., Sagha, H., Chavarriaga, R., and Millán, J. (2011). A hybrid brain-computer interface based on the fusion of electroencephalographic and electromyographic activities. *J. Neural Eng.* 8, 025011.

Leeb, R., Tavella, M., Perdikis, S., and Millán, J. (2010). "Towards a hybrid-bci: reliability dependent usage of either muscle or brain activity for a simple control task," in *Proceedings of the TOBI Workshop 2010: Integrating Brain-Computer Interfaces with Conventional Assistive Technology*, Rome, 19.

Mason, S. G., and Birch, G. E. (2003). A general framework for brain-computer interface design. *IEEE Trans. Neural Syst. Rehabil. Eng.* 11, 70–85.

Millán, J. D. R., Rupp, R., Müller-Putz, G. R., Murray-Smith, R., Giugliemma, C., Tangermann, M., Vidaurre, C., Cincotti, F., Kübler, A., Leeb, R., Neuper, C., Müller, K.-R., and Mattia, D. (2010). Combining brain-computer interfaces and assistive technologies: state-of-the-art and challenges. *Front. Neurosci.* 4:161. doi:10.3389/fnins.2010.00161

Müller-Putz, G. R., Scherer, R., Pfurtscheller, G., and Rupp, R. (2005). EEG-based neuroprosthesis control: a step towards clinical practice. *Neurosci. Lett.* 382, 169–174.

Nijboer, F., Sellers, E. W., Mellinger, J., Jordan, M. A., Matuz, T., Furdea, A., Halder, S., Mochty, U., Krusienski, D. J., Vaughan, T. M., Wolpaw, J. R., Birbaumer, N., and Kübler, A. (2008). A P300-based brain-computer interface for people with amyotrophic lateral sclerosis. *Clin. Neurophysiol.* 119, 1909–1916.

Pfurtscheller, G., Allison, B. Z., Brunner, C., Bauernfeind, G., Solis-Escalante, T., Scherer, R., Zander, T. O., Müller-Putz, G., Neuper, C., and Birbaumer, N. (2010). The hybrid BCI. *Front. Neurosci.* 4:30. doi:10.3389/fnpro.2010.00003

Pfurtscheller, G., and Neuper, C. (2001). Motor imagery and direct brain-computer communication. *Proc. IEEE* 89, 1123–1134.

Portelli, A. J., and Nasuto, S. J. (2008). "Toward construction of an inexpensive brain computer interface for goal oriented applications," in *Proceedings of Artificial Intelligence and the Simulation of Behaviour society 2008*, Aberdeen, 2–7.

Quitadamo, L. R., Marciani, M. G., Cardarilli, G. C., and Bianchi, L. (2008). Describing different brain computer interface systems through a unique model: a uml implementation. *Neuroinformatics* 6, 81–96.

Rogova, G., and Nimier, V. (2004). "Reliability in information fusion: literature survey," in *Proceedings of the 7th International Conference on Information Fusion*, Stockholm, 1158–1165.

rtsBCI. (2004–2008). Graz brain-computer interface real-time open source package. Available at: http://sourceforge.net/projects/biosig/

Schalk, G., McFarland, D. J., Hinterberger, T., Birbaumer, N., and Wolpaw, J. R. (2004). BCI2000: a general-purpose brain-computer interface (BCI) system. *IEEE Trans. Biomed. Eng.* 51, 1034–1043.

Schalk, G., Wolpaw, J. R., McFarland, D. J., and Pfurtscheller, G. (2000). EEG-based communication: presence of an error potential. *Clin. Neurophysiol.* 111, 2138–2144.

Susila, I. P., Kanoh, S., Miyamoto, K., and Yoshinobu, T. (2010). xBCI: a generic platform for development of an online BCI system. *IEEE Trans. Biomed. Eng.* 5, 467–473.

Tonin, L., Carslon, T., Leeb, R., and Millán, J. (2011). "Brain-controlled telepresence robot by motor-disabled people," in *Proceedings of Annual International Conference of the IEEE Engineering in Medicine and Biology Society EMBC 2011*. Boston.

Valsan, G., Grychtol, B., Lakany, H., and Conway, B. (2009). "The strathclyde brain computer interface," in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, Minneapolis, 606–609.

Vanhooydonck, D., Demeester, E., Nuttin, M., and Van Brussel, H. (2003). "Shared control for intelligent wheelchairs: an implicit estimation of the user intention," in *Proceedings of 1st International Workshop Advances in Service Robotics*, Bardolino, 176–182.

Venthur, B., Scholler, S., Williamson, J., Dähne, S., Treder, M., Kramarek, M., Müller, K., and Blankertz, B. (2010). Pyff – a pythonic framework for feedback applications and stimulus presentation in neuroscience. *Front. Neurosci.* 4:12. PMID: 21160550.

Wolpaw, J. R. (2007). Brain-computer interfaces as new brain output pathways. *J. Physiol.* 579, 613–619.

Wolpaw, J. R., Birbaumer, N., Heetderks, W. J., McFarland, D. J., Peckham, P. H., Schalk, G., Donchin, E., Quatrano, L. A., Robinson, C. J., and Vaughan, T. M. (2000). Brain-computer interface technology: a review of the first international meeting. *IEEE Trans. Rehabil. Eng.* 8, 164–173.

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

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Received: 25 July 2011; accepted: 01 November 2011; published online: 24 November 2011.

Citation: Müller-Putz GR, Breitwieser C, Cincotti F, Leeb R, Schreuder M, Leotta F, Tavella M, Bianchi L, Kreilinger A, Ramsay A, Rohm M, Sagebaum M, Tonin L, Neuper C and Millán JdR (2011) Tools for brain-computer interaction: a general concept for a hybrid BCI. Front. Neuroinform. ***5****:30. doi: 10.3389/fninf.2011.00030*