Towards a Free and Open Source BCI System Written in Python

B. Venthur[1], B. Blankertz[1]

[1]Berlin Institute of Technology, Neurotechnology Group, Berlin, Germany

Correspondence: Bastian Venthur, TU-Berlin, Marchstr. 23, 10587 Berlin, Germany. E-mail: bastian.venthur@tu-berlin.de

Abstract. The following paper describes the current state of our effort to create a completely free and open source BCI System in Python. A general BCI system consists of three parts: signal acquisition, signal processing, and feedback and stimulus presentation. Accordingly, our software consists of three parts: Mushu for the signal acquisition, Wyrm for the signal processing and Pyff for the feedback and stimulus presentation. All three components combined form a complete BCI system. Through usage of well defined interfaces each component can also be used stand-alone in a different setting (e. g. as part of a different BCI system). The whole system runs on all major operating systems, like Windows, Mac OS and Linux, is written in Python and is free and open source software licensed under the terms of the GNU General Public License.

Keywords: BCI, Python, Signal Acquisition, Signal Processing, BCI Feedback, Software

1. Introduction

In this paper we describe the three software components which will form together a complete, free and open source BCI system written in Python.

Similar projects to create complete BCI systems exist, like BCI2000 [Schalk et al., 2004] or OpenVibe [Renard et al., 2010]. BCI2000 is free only for non-commercial and educational usage and OpenVibe being truly free software is licensed under the terms of the GPL. Both are written in C/C++ which gives better performance, but makes it also much harder to write BCI experiments and applications for non-computer scientists. OpenVibe mitigates this by allowing for an drag- and drop like approach for visual programming of experiments.

We decided to use Python as the programming language of choice because we think it is an excellent general purpose programming language with a large and comprehensive standard library. Together with SciPy, NumPy and matplotlib [Jones et al., 2001], Python is a powerful and free alternative to commercial packages like Matlab. Studies [Prechelt, 2000] have shown that programming in high level languages like Python significantly shortens the amount of time needed to implement a solution and leads to shorter and thus less error-prone code compared to more low level languages like C or C++. This is particularly important for software which is going to be used and modified not only by computer scientists, but students and researchers from different fields.

2. Components of our BCI System

Fig. 1 shows an overview over the general structure of a BCI system. EEG data is measured via EEG caps from the subject's head and the signal is amplified through the EEG amplifiers. A signal acquisition software collects the data from the amplifier and forwards it to the signal processing where usually some machine learning algorithm extracts information from the EEG data and forwards it to the feedback/stimulus presentation.

With Mushu [Venthur and Blankertz, 2012] we are providing the signal acquisition part, with Wyrm the signal processing, and with Pyff [Venthur et al., 2010] the feedback and stimulus presentation. All three components will form a complete BCI system written in Python.

2.1. Mushu

When doing BCI experiments with EEG data one always has to use EEG amplifiers to acquire the brain signals. Those amplifiers usually come with their own software and different vendors have different formats for saving and online streaming of the EEG data. Often the software provided by the amplifier vendors only runs on Microsoft Windows systems, leaving out the Mac OS and Linux users. With Mushu solve those problems altogether: we want a signal acquisition software that runs on all major operating systems, that supports a wide range of EEG hardware, that produces and outputs data in a standardized format independent from the amplifier. We already reverse-engineered the g.USBamp and the EPOC [Venthur and Blankertz, 2012] system and wrote pure Python drivers which allow to

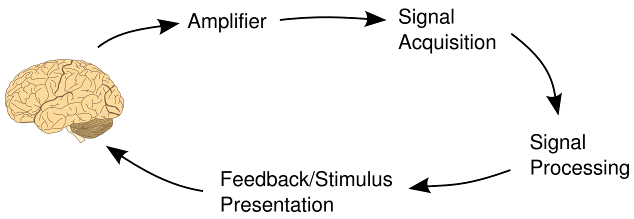

Figure 1: General structure of a closed loop BCI system. EEG data is acquired from the subject, fed through the amplifier and collected by the signal acquisition. The signal acquisition forwards the EEG data to the signal processing where information is extracted from the data and forwarded to the feedback/stimulus presentation.

use those systems on all Operating Systems without the need for external drivers. Other drivers for systems like: BrainAmp, TMSi Mobita, and Enobio will follow.

2.2. Wyrm

Wyrm represents the signal processing part of our system. It is suitable for online experiments and off-line analysis. Wyrm is currently under development, but it already contains crucial parts of a BCI toolbox like Common Spacial Patterns (CSP) [Blankertz et al., 2008] for processing, and Linear Discriminant Analysis (LDA) for classification. Wyrm is not yet ready for the public to use but a prototype doing successful classification already runs in our lab.

2.3. Pyff

Pyff is a framework to develop and run BCI feedback and stimulus applications in Python. It was designed to make the development of feedback and stimulus applications as easy as possible. We particularly developed it with non-computer scientists in mind which often create and implement new BCI paradigms. The framework communicates with the rest of the BCI system via a standardized communication protocol using UDP and XML and is therefore suitable to be used with any BCI system that may be adapted to send its control signal via UDP in the specified format. We also provide adapters for BCI2000 and any TiC (www.bcistandards.org/softwarestandards/tic) compatible BCI.

3. Outlook

At the time of writing this abstract. The here introduced BCI system is still work in progress. While Pyff is mature and published years ago, Mushu and Wyrm are working prototypes and not yet ready for release. The source code is nevertheless publicly available and contributors are invited to test the code and contribute. We continue to develop this system and aim for having it in a usable state in 2014.

Mushu, Wyrm, and Pyff are free- and open source software licensed under the terms of the GNU General Public License (GPL). The repositories are github.com/vethur/(mushu,wyrm,pyff).

Acknowledgments

This work was supported in part by grants of BMBF 01GQ0850 and 16SV5839.

References

Blankertz, B., Tomioka, R., Lemm, S., Kawanabe, M., and Mller, K. (2008). Optimizing spatial filters for robust EEG single-trial analysis. *IEEE Sig Proc Mag*, 25(1):41–56.

Jones, E., Oliphant, T., Peterson, P., et al. (2001). SciPy: Open source scientific tools for Python.

Prechelt, L. (2000). An empirical comparison of C, C++, Java, Perl, Python, Rexx and Tcl. *IEEE Comput*, 33(10).

Renard, Y., Lotte, F., Gibert, G., Congedo, M., Maby, E., Delannoy, V., Bertrand, O., and Lécuyer, A. (2010). OpenViBE: an open-source software platform to design, test, and use brain-computer interfaces in real and virtual environments. *Presence: teleoperators and virtual environments*, 19(1):35–53.

Schalk, G., McFarland, D., Hinterberger, T., Birbaumer, N., and Wolpaw, J. (2004). BCI2000: a general-purpose brain-computer interface (BCI) system. *IEEE Trans Biomed Eng*, 51(6):1034–1043.

Venthur, B. and Blankertz, B. (2012). Mushu, a free- and open source BCI signal acquisition, written in Python. In *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, pages 1786–1788.

Venthur, B., Scholler, S., Williamson, J., Dhne, S., Treder, M. S., Kramarek, M. T., Mller, K.-R., and Blankertz., B. (2010). Pyff—A Pythonic Framework for Feedback Applications and Stimulus Presentation in Neuroscience. *Front Neurosci*, 4.