DOI: 10.3217/978-3-85125-467-9-64

On the use of ROS as a common infrastructure for robotic BCI driven applications

L. Tonin[1,2*], A. Cimolato[2], E. Menegatti[1,2]

[1]Intelligent Autonomous Systems Lab, Department of Information Engineering, University of Padova Italy
[2]EXiMotion s.r.l., Padova, Italy

*Via Ognissanti 72, IT-35129, Padova, Italy. E-mail: luca.tonin@dei.unipd.it

Introduction:
The number of Brain-Computer Interface (BCI) driven applications to control actual devices is rapidly increasing, ranging from robotic arms to mobile platforms. However, each research group integrates BCI systems into robot control in different ways depending on their background and their software packages. This makes difficult to propagate open-source software, to share source code and to replicate experimental results. Herein, we propose a common design for BCI driven applications based on the Robot Operating System (ROS) [1], a middleware framework that in the last years became the worldwide standard de-facto in robotics.

Material, Methods and Results:
A ROS implementation is based on four fundamental elements: nodes, messages, topics and services. **Nodes** are stand-alone, independent processes—namely, software modules—that perform specific computations. Inter-nodes communication is provided by message exchange. **Messages** are data structures that support standard primitives as well as custom, user-defined nested typed fields (i.e., C-like structures). Nodes communicate by publishing and/or subscribing to a given topic. **Topics** can be considered as stream channels (peer-to-peer communication based on TCP protocol). Finally, nodes can broadcast synchronous request-response transactions via specific routines named **services**. Although ROS was designed for robotic applications, its infrastructure perfectly matches the basic requirements of any BCI system. In fact, regardless of the unique characteristics of each BCI system, they share the same information flow to implement the BCI loop: acquisition, processing, classification, and feedback modules. In a ROS-based implementation of the BCI loop, each of the aforementioned modules is straightforwardly implemented as a stand-alone node. Each node communicates with the rest of the system by publishing on predefined and standardized topics. For instance, an EEG acquisition device would publish on the */data/eeg/raw* topic, and any EEG processing modules will be notified when new EEG data are available. The same design can be replicate for other modules (e.g., for the classification modules) as well as for additional acquisition devices. The strong competitive advantage of using ROS is the fact that the whole communication framework is provided "out-of-the-box" and thus, researchers can concentrate efforts on the implementation of the own custom modules—if necessary.

Discussion:
ROS shares and extends the main advantages of the most common BCI platforms (e.g., BCI2000, OpenVibe, CIP; see [2] for an extended review). ROS is open-source software released under BSD and GPL license, it is cross-platform, multi-language (e.g., C++, Python) and multi-architecture and it ensures highly efficiency in terms of resources and memory. In particular, one the most important advantage of ROS is its strong modularity: people can design and implement its own ROS package with specific functionalities and thus, distribute it through common repositories. In a BCI framework, this means that it would be possible to easily integrate together several user-contributed, deeply tested packages implementing different steps of the BCI loop (e.g., classification, artifact removal). It is paradigmatic what happened in the robotic community: the impressive number of available packages in the ROS ecosystem (more than 3000 in less than 5 years) demonstrates that such a distributed approach for sharing reliable code definitely works. With respect to the aforementioned BCI platforms, the adoption of ROS infrastructure would allow integrating BCI and control of robotic devices in the same ecosystem. Following the idea of a hybrid BCI and a shared control approach to drive external devices [3], the BCI output signal might be directly used into algorithms for navigation or trajectory generation of prosthetic robot arms, which are already available, tested and widely adopted in ROS.

Significance:
A standard infrastructure is becoming increasingly important to handle the proliferation of BCI driven devices in a common way. A BCI based on ROS would match the fundamental requirements of any BCI system as well as would provide a direct interface for most available robotic platforms. Furthermore, it would allow sharing the same code across different groups, by increasing the reliability and the efficiency of BCI driven applications.

References

[1] Quigley M, et al. ROS: an open-source Robot Operating System, In *ICRA Workshop on Open Source Software*, 2009

[2] Brunner C, et al.. BCI software platforms. In Towards Practical Brain-Computer Interfaces. Springer, 2012.

[3] Leeb R, et al. Towards Independence: A BCI Telepresence Robot for People with Severe Motor Disabilities. *Proceedings of the IEEE*, 103(6): 969-82, 2015.