Oppermann, Hannes; Wichum, Felix; Esch, Lorenz; Haueisen, Jens;
Klemm, Matthias:

MagCPP: a C++ toolbox for combining neurofeedback with Magstim transcranial magnetic stimulators

Original published in:	Current directions in biomedical engineering. - Berlin : De Gruyter. - 6 (2020), 3, art. 20203128, 4 pp. (Annual Meeting of the German Society of Biomedical Engineering ; (Leipzig) : 2020.09.29-10.01)
Original published:	2020-11-26
ISSN:	2364-5504
DOI:	10.1515/cdbme-2020-3128
[*Visited:*	2021-06-07]

Hannes Oppermann *, Felix Wichum, Lorenz Esch, Jens Haueisen, Matthias Klemm

MagCPP: A C++ toolbox for Combining Neurofeedback with Magstim transcranial magnetic stimulators

Abstract: Transcranial magnetic stimulation (TMS) is an established method to treat various neurological diseases, such as depression, Alzheimer's disease, and tinnitus. New applications for TMS are closed loop neurofeedback (NF) scenarios, which require software control of the TMS system, instead of the currently used manual control. Hence, the MagCPP (https://github.com/MagCPP) toolbox was developed and is described in this work. The toolbox enables the external control of Magstim TMS devices via a C++ interface. Comparing MagCPP to two other toolboxes in a TMS application scenario with 40% power, we found that MagCPP works faster and has lower variability in repeated runs (MagCPP, Python, MATLAB [mean±std in seconds]: 1.19±0.00, 1.59±0.01, 1.44±0.02). An integration of MagCPP in a real-time data processing platform MNE-CPP with an optional GUI demonstrates its ability as part of a closed-loop NF-scenario. With its performing advantages over other toolboxes, MagCPP is a first step towards a complete closed loop NF scenario and offers possibilities for novel study designs.

Keywords: TMS, EEG, closed loop, BCI, Data acquisition, Data processing, Medical software

https://doi.org/10.1515/cdbme-2020-3128

***Corresponding author: Hannes Oppermann:** Institute of Biomedical Engineering and Informatics, Technische Universität Ilmenau, Ilmenau, Germany, E-Mail: hannes.oppermann@tu-ilmenau.de
Felix Wichum, Jens Haueisen, Matthias Klemm: Institute of Biomedical Engineering and Informatics, Technische Universität Ilmenau, Ilmenau, Germany
Lorenz Esch: Athinoula A. Martinos Center for Biomedical Imaging, Massachusetts General Hospital, Charlestown, MA, USA;
Institute of Biomedical Engineering and Informatics, Technische Universität Ilmenau, Ilmenau, Germany;
Boston Children's Hospital, Boston, MA, USA

1 Introduction

Non-invasive transcranial brain stimulation (NTBS) techniques, such as Transcranial Magnetic stimulation (TMS), are already widely used tools to study the relationship between cortical activity and behavior, to trace the timing at which activity in a particular cortical region contributes to a given task, and to map the functional connectivity between brain regions [1, 2]. In conjunction with Electroencephalography (EEG), TMS can be used to directly stimulate specific cortical regions and, at the same time, measure the stimulation induced changes of activity and connectivity [3, 4]. Accordingly, this method can be used to identify abnormal connectivity due to brain damage [5].

So far, most research studies utilize TMS in offline related scenarios, with the data being analyzed after the actual measurement session. Such offline studies have, for example, demonstrated that TMS-feedback improves executive function in autistic patients [6].

From a neuroscientific point of view, the potentials for real-time processing of neuronal data are manifold. Such approaches not only enable a faster and more intuitive insight on instantaneous brain functions, but more importantly they create the foundation for a wide range of neurofeedback (NF) scenarios. Due to their high temporal resolution, Magnetoencephalography (MEG) and EEG are ideal candidates for processing brain activation in real-time. Processing MEG/EEG data in real-time, introduces the following, nontrivial challenges: the low Signal-to-Noise-Ratio (SNR), the large amount of incoming data and the high computational cost of complex analysis procedures. Despite the challenges, MEG/EEG real-time processing can be become of interest to the neuroscience community as it has the potential for a fundamental change in neuroscientific work - away from experiments with fixed paradigms in favor of highly dynamic and adaptable paradigms depending on the subject's brain state. NF scenarios enable researchers to test hypotheses about specific brain properties by monitoring this

property in real-time and adapting the experimental interventions according to the state of this parameter.

TMS with concurrent EEG real-time processing methods offer the opportunity to introduce real-time feedback to neuroimaging studies. To our knowledge, there exist no major contributions to include sophisticated real-time processing steps, e.g., real-time source or connectivity estimation, in an EEG/TMS measurement session. Theoretical concepts and hypotheses have been proposed to integrate cortical stimulation into a NF scenario or BCI system [7, 8]. Still, only little work on directly embedding TMS into an advanced closed loop pipeline has been proposed so far [9, 10].

The MNE-CPP framework aims at combining real-time data processing with advanced processing steps, which normally are performed after rather than during the measurement. These processing steps include real-time data pre-processing, source estimation and functional connectivity estimation. MNE-CPP [11] is written in C++ in order to cope with the computationally intensive processing steps. MNE Scan [12] is a standalone application built with MNE-CPP, which can merge multiple advanced processing steps together and provide the results in real-time to subsequent processing or visualization steps. The real-time results could be used to guide TMS during the measurement, either manually or via a robotic arm.

To the best of our knowledge, no open source interface is currently available, which provides a C++ interface to communicate with Magstim TMS devices. We propose MagCPP as a C++ interface to be used in C++ based projects, e.g., MNE Scan. In the following, we describe the MagCPP interface implementation and compare its performance to MagPy [13] and MAGIC [14], which are interfaces for TMS devices based on Python and MATLAB, respectively. Moreover, we present a first use case scenario based on a closed-loop pipeline, implemented in MNE Scan. Finally, we provide a short discussion and outlook of MagCPP.

2 MagCPP toolbox

2.1 Overview

MagCPP is a platform-independent (Windows, Linux, MacOS) toolbox, written in C++ to control Magstim Rapid² devices (The Magstim Company Ltd., Whitland, UK). It was inspired by the Python toolbox *MagPy* and is released under the GNU General Public License (v3). The toolbox is a standalone and open source software and can be freely downloaded from GitHub (https://github.com/MagCPP).

2.2 Requirements

MagCPP's only external dependency is Qt (https://www.qt.io). As MagCPP employs parallel processing with several threads, a computer with a multi-core CPU should be preferred. The computer needs to possess a serial port or a USB-to-serial adapter to connect to the Magstim TMS device using a serial cable or a QuickFire cable [13]. The latter allows low latency triggering of TMS pulses.

2.3 Toolbox structure and usage

The toolbox is structured into three device classes and two communication classes. By calling the appropriate device constructor, an object is created, and the communication threads are started. The *SerialPortController* class monitors the serial port, receives and sends messages to the stimulator. It serves as a connection piece between the software and the device. Depending on the state, e.g. connected or armed, messages are sent by the *ConnectionRobot* class at regular intervals to maintain the current status and to avoid a connection abort.

A complete run for TMS application is shown as sequence diagram in Figure 1. It consists of initializing, connecting to the device, setting the power, bypassing the safety switch to trigger shots, arming, firing and disconnecting.

Time-sensitive triggering of the device poses a difficulty considering that commands are regularly send to maintain the current status.

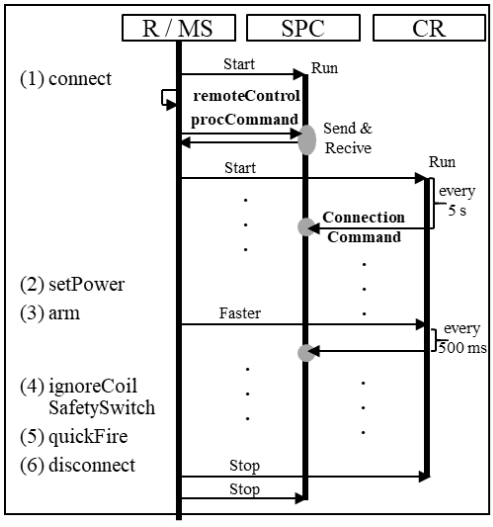

Figure 1: Sequence diagram of a complete run with necessary functions (1) – (6). Three columns with boxes (*Rapid*/*Magstim* R/MS, *SerialPortController* SPC and *ConnectionRobot* CR) symbolizing threads. The internal flow is in the center. Internal methods are written in bolt. Grey forms are for communication with Magstim device.

These could be overlaid with new commands and cause a delay. For this reason, the *ConnectionRobot* can be paused for a short time, to allow privileged handling of time-sensitive commands.

2.4 Characterization

2.4.1 Measurement paradigm

A run with all necessary functions is called here a complete run, which is shown in Figure 1. A wait time of one second after the arm command is required to ensure that the device is prepared to fire.
In the following, the software packages MagCPP, MAGIC and MagPy are compared. Each measurement was repeated for 15 times. Examined were:

- the influence of an USB-to-serial adapter in comparison to a regular serial port (investigated using MagPy),
- the adjustment of different power settings,
- the differences between *quickFire* and *fire* command.

The difference between the *quickFire* and the *fire* command were examined only in MagCPP and MagPy, because there is no *quickFire* function in MAGIC. To receive feedback about successful function calling from the software, all functions were set to generate a return value. Values with more than three scaled median absolute deviations were removed as outliers.

2.4.2 Hardware setup

First, the influence of an USB-to-serial adapter in comparison to a regular serial port was investigated, using the MagPy software package. No significant differences between a regular serial port and a USB-to-serial adapter could be found, which is shown in Table 1. As the computer with the regular serial port was not available for further measurements, all subsequent measurements were taken with a Microsoft Surface Pro 4, Windows 10 Pro, Intel® Core™ i7-6650U, 16 GB RAM and an USB-to-serial adapter.

Table 1: Comparison between the serial port and USB-to-serial port adapter.

	Mean (s)	Std (s)
Serial Port	1.443	0.018
USB-to-serial adapter	1.447	0.014

2.4.3 Results

Figure 2 shows the runtimes of the three examined toolboxes. MagCPP is the fastest, while MagPy requires the longest runtime. The largest variances of runtimes were measured at 20% TMS power with MagPy and MAGIC. The small variance in runtimes for MagCPP is similar for all power settings.

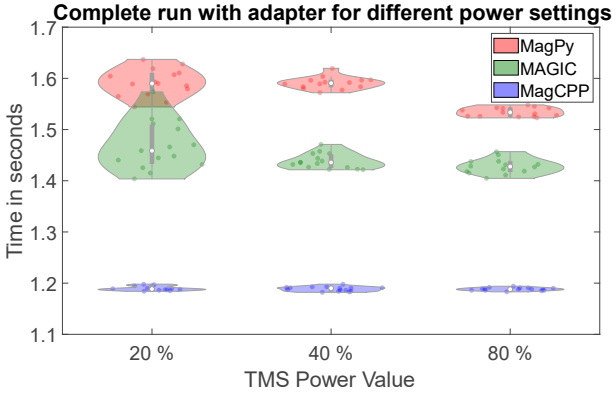

Figure 2: Runtimes of complete runs with MagCPP (blue), MagPy (red) and MAGIC (green) at different TMS power levels.

The results of the comparison of the quickFire and fire command are depicted in Table 2. MagCPP performs also faster than MagPy for all investigated TMS power levels. No significant difference was found between the fire and the quickFire command.

Table 2: Runtime comparison of QuickFire (qF) and fire (f) command using MagCPP and MagPy at different power settings (values are mean and standard deviation in italic)

Power	20 %		40 %		80 %	
Mode	**qF**	**f**	**qF**	**f**	**qF**	**f**
MagCPP	1.20	1.19	1.21	1.19	1.20	1.19
	0.00	*0.01*	*0.01*	*0.00*	*0.00*	*0.00*
MagPy	1.55	1.59	1.55	1.59	1.55	1.53
	0.02	*0.03*	*0.02*	*0.01*	*0.03*	*0.01*

3 Closed loop Neurofeedback scenario

3.1 Integration into MNE-CPP

An alternative to the standalone version of MagCPP is an integrated version of MagCPP in MNE Scan. This integration is also available on GitHub (MagCPP/mne-cpp). Combined with data acquisition and signal processing plugins, provided

by MNE Scan, a closed loop NF-scenario can be established. Our TmsNeurofeedback plugin allows to set the conditions under which a TMS impulse is triggered. Additionally, the possibility of visual feedback to the subject is given.

3.2 Graphical-user interface

The TmsNeurofeedback plugin also implements a graphical user interface (GUI). It provides device dependent and feedback settings. The user can choose between single pulse and repetitive TMS (rTMS) mode. Furthermore, the user can choose between static and dynamic power values. In case of static power, the desired value must be selected. In dynamic power mode, the TMS power is set in dependence of the input signal. In the visual feedback settings, it is possible to change the ranges for positive, neutral and negative feedback depending on the input value. Furthermore, it is possible to set the time delay between two feedback images.

4 Discussion

In this work, a new software package called MagCPP is introduced, which implements an interface to Magstim Rapid² devices in C++. We found significant performance differences compared to MagPy (Python) and MAGIC (MATLAB) and show that MagCPP is suitable for an application in NF scenarios. The presented measurements and comparisons to other software packages were all carried out on Windows with Rapid² devices. We do not expect fundamentally different results on other platforms. The comparison of the different software packages was based on code runtimes, which were measured using appropriate functions provided by each programming language. As each programming language implements its own method to measure to runtime of a certain piece of code, our results may also reflect this influence. The overhead introduced by those runtime measurement functions is usually much smaller than the differences found between the software packages, which are in the order of a few hundred milliseconds. It remains to be characterized how long it takes for a command to be sent to the device via the QuickFire cable and to be processed there (e.g. to fire the stimulator).
The ability to set the TMS power depending on the state of the brain offers a new possibilities study design. The implemented toolbox was not yet tested in human subjects.

5 Conclusion

A new open source software MagCPP to control Magstim Rapid² TMS devices was established. Its usage for real-time processing is feasible and has been tested with a quickFire cable under Windows. In contrast to the MagPy and the MAGIC toolboxes, the presented MagCPP toolbox can be integrated into MNE-CPP and an optional GUI is available. The presented use case is a big step towards a complete closed loop NF scenario. It offers possibilities for adaptive stimulus intensities and novel study designs.

Author Statement

Research funding: DFG Ha2899/26-1 and Free State of Thuringia 2019 FGR 0083. Conflict of interest: Authors state no conflict of interest.

References

[1] Pascual-Leone, A. Transcranial magnetic stimulation in cognitive neuroscience – virtual lesion, chronometry, and functional connectivity. Curr Opin Neurobiol 2000;2(10):232–237.
[2] Kunze T, Hunold A, Haueisen J, Jirsa V, Spiegler A. Transcranial direct current stimulation changes resting state functional connectivity. A large-scale brain network modeling study. NeuroImage 2016;140:174–187.
[3] Ilmoniemi RJ, Virtanen J, Ruohonen J, Karhu, J, Aronen HJ, Näätänen R, et al. Neuronal responses to magnetic stimulation reveal cortical reactivity and connectivity. Neuroreport 1997;8(16):3537–3540.
[4] Zrenner C, Belardinelli P, Müller-Dahlhaus F, Ziemann U. Closed-Loop Neuroscience and Non-Invasive Brain Stimulation. A Tale of Two Loops. Front Cell Neurosci 2016;10:92.
[5] Borich, MR, Wheaton LA, Brodie SM, Lakhani B, Boyd LA. Evaluating interhemispheric cortical responses to transcranial magnetic stimulation in chronic stroke. A TMS-EEG investigation. Neurosci Lett 2016;618:25–30.
[6] Sokhadze, E.M. et al., 2014. Neuromodulation integrating rTMS and neurofeedback for the treatment of autism spectrum disorder: An exploratory study. Applied Psychophysiology and Biofeedback, 39(3–4), pp.237–257.
[7] Bergmann TO, Karabanov A, Hartwigsen G, Thielscher A, Siebner HR. Combining non-invasive transcranial brain stimulation with neuroimaging and electrophysiology. Current approaches and future perspectives. NeuroImage 2016;140:4–19.
[8] Sergeeva EG, Henrich-Noack P, Bola M, Sabel BA. Brain-state-dependent non-invasive brain stimulation and functional priming. A hypothesis. Front hum neurosci 2014;8:899.
[9] Gharabaghi A, Kraus D, Leão MT, Spüler M, Walter A, Bogdan M et al. Coupling brain-machine interfaces with cortical stimulation for brain-state dependent stimulation. Enhancing motor cortex excitability for neurorehabilitation. Front hum neurosci 2014;8:122.
[10] Walter A, Ramos MA, Spüler M, Naros G, Leão MT, Gharabaghi A et al. Coupling BCI and cortical stimulation for brain-state-dependent stimulation. Methods for spectral estimation in the presence of stimulation after-effects. Front neural circuit 2012;6:87.
[11] Esch L, Dinh C, Larson E, Engemann D, Jas M, Khan S, et al. MNE: Software for Acquiring, Processing,and Visualizing MEG/EEG Data. In: Supek S, Aine C, editors. Magnetoencephalography. Cham: Springer 2019:355-371.
[12] Esch L, Sun L, Klüber V, Lew S, Baumgarten D, Grant PE et al. MNE Scan. Software for real-time processing of electrophysiological data. J Neurosci Methods 2018;303:55–67.
[13] McNair NA. MagPy: A Python toolbox for controlling Magstim transcranial magnetic stimulators. J Neurosci Methods 2017;276:33–37.
[14] Habibollahi SF, Rogasch NC, McNair NA, Biabani M, Pillen SD, Marshall TR et al. MAGIC. An open-source MATLAB toolbox for external control of transcranial magnetic stimulation devices. Brain stimul 2018;11(5):1189–1191.