Composing an Ensemble Standstill Work for Myo and Bela

Charles P. Martin
Department of Informatics and RITMO
University of Oslo
charlepm@ifi.uio.no

Alexander Refsum Jensenius
RITMO and Department of Musicology
University of Oslo
a.r.jensenius@imv.uio.no

Jim Torresen
Department of Informatics and RITMO
University of Oslo
jimtoer@ifi.uio.no

ABSTRACT

This paper describes the process of developing a standstill performance work using the Myo gesture control armband and the Bela embedded platform for low latency audio and sensor processing. A standstill work, where less movement results in more sound, inverts the typical control paradigm of musical instruments. Such works have been previously demonstrated using motion capture systems, however the combination of Myo and Bela allows a portable and extensible version of this concept while introducing muscle tension as an additional control parameter. We describe the technical details of our setup and introduce our Myo-to-Bela software bridge, as well as a new cross-platform Myo-to-OSC bridge to assist with prototyping compositions using the Myo controller.

Author Keywords

Bela, Myo, embedded instrument, motion instrument

CCS Concepts

•**Applied computing** → *Performing arts; Sound and music computing;* •**Human-centered computing** → *Gestural input;*

1. INTRODUCTION

The Myo armband from Thalmic Labs Inc. is a commercial gesture control interface that includes a 9-axis inertial measurement unit (IMU), as well as 8 electromyograph (EMG) sensors, in a convenient battery-powered package. The device communicates wirelessly via Bluetooth Low Energy (BLE). The Myo is typically worn on a user's lower arm where the EMG sensors pick up signals from many of the muscles connected to hand and fingers. Many of the Myo's commercial applications focus on using postures and/or gestures for starting and stopping processes, such as swiping to move between slides in a presentation; in contrast, musical performance applications often use the continuous data streams from the EMG and IMU sensors [10].

Recent work by Jensenius et al. [7] explored the potential of the Myo to support microinteraction—the use of the smallest possible control actions–in digital musical performance. We have extended that preliminary work to produce a fully mobile setup (Figure 1) for performing with the Myo that is robust enough for a variety of interactive situations. Our system makes use of Bela, an audio processing platform for Beaglebone single-board Linux computers. The system has been tested in the performance of *Stillness Under Tension*, featuring four Myo performers.

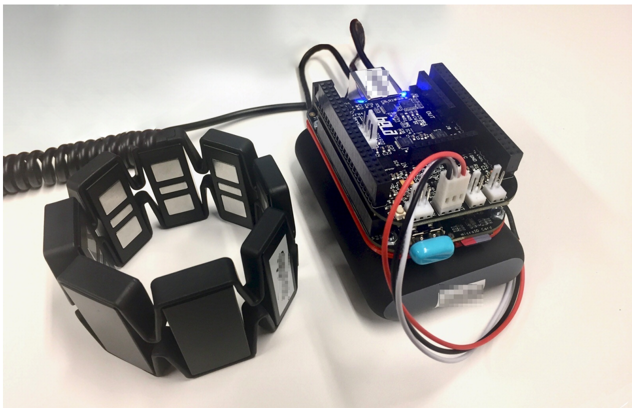

Figure 1: Our mobile performance setup using a Bela and Myo, powered by a USB battery. Audio output is via the Bela's line output which can be connected to headphones, a mobile speaker, or a PA system.

Since its commercial release in 2015, the Myo has proven to be popular for performative interactions among electronic artists, musicians, and dancers. However, in our research and musical explorations with the Myo, enthusiasm for the system has been tempered by two technical limitations:

1. Performances with the Myo typically use Thalmic's Myo Connect software and a third-party bridge to send sensor data to relevant software, such as the Myo-for-Max object[1]. Unfortunately, Thalmic Labs do not provide a Linux version of Myo Connect, which hinders the use of Myo on embedded platforms such as Raspberry Pi and Beaglebone.

2. The BLE dongle that ships with the Myo can generally handle up to three simultaneous connections, so ensemble works are limited to three armbands unless multiple laptops are used. An ensemble work where performers wear two armbands each (one on each arm) would therefore require a complex multi-computer setup. Our attempts at such have been technically challenging, with connectivity issues and software crashes.

[1]Myo-for-Max: `https://github.com/JulesFrancoise/myo-for-max/`

UNDER REVIEW, Not for Distribution

In this paper we describe our solutions to these two problems. Addressing the first issue, we have implemented a custom Bela render file that connects Bela to the Myo, as well as a pure-Python Myo-to-OSC bridge that allows composition with Myos on desktop Linux as well as other platforms. Secondly, by connecting Myos to Belas, we work around the limitations of holding multiple BLE connections by simply using one Bela per performer. In the following sections we discuss the technical setup for connecting Myo to Bela and our cross-platform prototyping workflow. In Section 4 we discuss how these setups are applied in *Stillness Under Tension*, a standstill ensemble work for Myo.

2. RELATED WORK

Since becoming commercially available in 2015, the Myo has been examined several times at NIME and used in many different artistic performances. Nymoen et al. [10] compared the Myo's IMU performance to a high-quality motion capture system and experimented with using the EMG to select musical interactions, and the IMU to control them. Their conclusion was that the Myo's portability outweighed its sensing limitations, and allowed for motion-controlled NIMEs to be deployed in a variety of locations, such as the high school in their study. Benson et al. [1] used the Myo to naturally control 3D spatialisation of live instrumental sounds. Di Donato et al. [4] created MyoSpat which also used the Myo to control spatialisation of sound, as well as theatrical lighting; however, they also used machine-learning interpretations of the sensor data to control audio effects.

Typical applications of Myo to artistic performance make use of Thalmic Lab's Myo Connect software to manage Myo BLE connections and provide a high level C++ programming interface to IMU, EMG, and gesture classification data. Software such as MyoMapper [3] is often used to bridge between Myo Connect and computer music environments, by forwarding the incoming data in OSC streams that can be picked up in Pure Data, Max, Wekinator, and other software. As Myo Connect is only available for Windows and MacOS platforms, existing bridges such as MyoMapper cannot be used on Linux-based computers or embedded systems.

A trend has emerged over the last few years of developing self-contained NIMEs using embedded computers such as the Raspberry Pi or Beaglebone. Berdahl and Ju [2] argued that such computers could be integrated into the NIME itself, resulting in a compact and standalone instrument that would continue to function longer than a laptop-based setup. They demonstrated this concept with *Satellite CCRMA* that integrated a Raspberry Pi, Arduino, and Pure Data into a compact NIME-building platform. McPherson et al. [8] showed that an embedded platform can actually have superior interactive performance than a laptop setup on the important measure of action–sound latency. Their Bela platform uses a version of Linux with hard-realtime extensions and a custom ADAC expansion board for the Beaglebone Black to provide an integrated instrument-building platform that integrates Pure Data for accessible prototyping [9]. The Bela's integrated hardware and software platform was an advantage in our work, as we were able to easily replicate a single NIME design across four Bela setups without needing to assemble any additional hardware.

2.1 Standstill Performance

Previous work has examined the artistic implications of stillness, and defined the term *standstill* to mean voluntary, physical stillness that may, nevertheless, still be associated with sounds or continuous artistic processes [6]. This work notes that humans cannot stand completely still due to breathing and other involuntary motion that can be termed *micromotion*. The practice of standstill examines the extent to which it is possible for humans to stand completely still, and takes advantage of micromotion to control artistic processes. One interaction that has been explored in several previous performances are inverse mappings between micromotion and sound [5], where, for example, less motion is mapped to louder sounds. The concept of performative microinteraction counters the typical expectation of both performers and audiences—that more actions should result in more sound—and thus presents a challenging playground for developing new musical interfaces.

3. SYSTEM DESIGN

In this section we describe our hardware and software workflow for composing works using Myo armbands with the Bela embedded audio system.

3.1 Myo-to-Bela in C++

Working with Myo interfaces on the Bela platform is complicated by the fact that GNU/Linux operating systems are not officially supported. However, the BLE specification for the Myo is openly available[2] and several community-developed Myo libraries have emerged. For integration with Bela, we used `MyoLinux`[3], a C++ library for Myo that uses the virtual serial port provided by the Myo's included BLE dongle.

We implemented a custom render callback (`render.cpp` file) based on the Bela's `libpd`-enabled render file that establishes a Myo connection and sets the Myo to stream EMG and IMU data. As recommended in the Bela documentation, an `AuxiliaryTask` object is used to poll the Myo connection for new data. We implemented functions to process this data to produce, for example, Euler rotation angles and the magnitude of the accelerometer vector. The data is then sent to outlets in the Pure Data patch as follows:

emg EMG readings, a list of 8 floats in $[-1, 1]$.

ori A list of quaternions (x, y, z, w).

euler A list of Euler angles (roll, pitch, yaw) scaled to be in $[-1, 1]$.

acc A list of acceleration values (x, y, z).

gyr A list of gyroscope values (x, y, z).

accmag magnitude of acceleration vector (scalar).

gyrmag magnitude of gyroscope vector (scalar).

Our render file and installation instructions for the `MyoLinux` library are available on GitHub[4]. These data outlets and setup configuration were sufficient for the needs of our current performance project, however, many improvements could be made to improve flexibility for more use-cases. At present the system only supports one Myo, selected by MAC address; however, it would also be possible to connect to multiple Myos and send their data to separate outlets in `libpd`. The `MyoLinux` library could also be extended to support reading the Myo's internal gesture classification data. It may be possible to use the system BLE adapter available

[2]Myo Bluetooth Specification: `https://github.com/thalmiclabs/myo-bluetooth`

[3]`https://github.com/brokenpylons/MyoLinux`

[4]Bela Myo Example repository: Link to follow review.

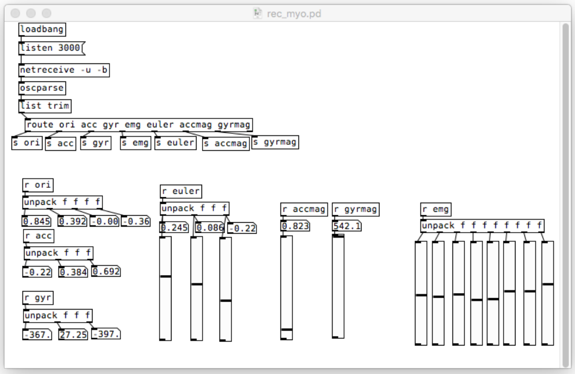

Figure 2: Pd patch visualising Myo data received from `myo-to-osc`. This patch allows prototyping on a computer before moving to the Bela platform.

on the BeagleBone Black Wireless, rather than the USB BLE dongle, although this would require significant modification.

3.2 Myo-to-OSC in Python

In parallel with our efforts to connect Myo to Bela, we have also developed a pure-Python solution for connecting Myo armbands to OSC applications, particularly on Linux although other platforms are also supported. Our `Myo-to-OSC`[5] application builds on previous Python Myo libraries but adds a command line interface for listing the MAC addresses and names of available Myos, and processes the Myo's data into artistically useful quantities, such as the Euler angles and movement magnitudes listed above. A Pure Data patch to visualise the OSC output (see Figure 3.2) is included in our application. This patch also sends Myo data received over OSC to identically named outlets as in the Myo-to-Bela render file, so interactions with the Myo can be tested in Pure Data on a computer before transferring the composition to the embedded Bela platform.

3.3 Performance Setup with Myo and Bela

Our performance setup includes a self-contained unit for each performer, these consist of a Bela with the attached BLE dongle and the wireless Myo armband (shown in Figure 1). The Bela receives power from a regular 3300 mAh battery pack, which usually lasts for 2-3 hours of performance. An important element here is that that the Bela can boot directly into a pre-configured patch. This means that a performer can just turn on the power, wait 40 seconds for the boot sequence to load the patch, and be start performing. This process appears to be reliable and we have not experienced any failures to load the patch or connect to the Myos. For sound rendering, we have used the Bela's built-in line output to connect to headphones for personal practice, using battery-powered speakers for rehearsal, and connecting to a PA mixer for performance.

4. STILLNESS UNDER TENSION

Our composition was inspired by the MicroMyo demonstration [7], with an additional focus on the potential for sonic variety and interaction within an ensemble. Like MicroMyo, sound generation was made from eight sine oscillators, with the amplitude of each being modulated from the signal of

[5]Myo-to-OSC Git Repository: Link to follow review.

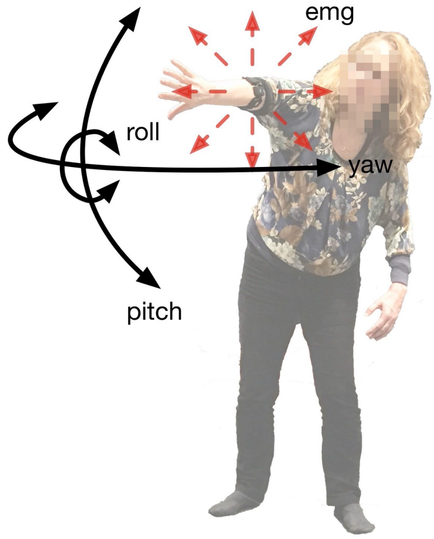

Figure 3: In our performance setup, both IMU and EMG signals from the Myo are used to control a synthesised instrument. The eight EMG sensors control the amplitude of eight sine oscillators, while the pitch and yaw of the IMU signal govern the frequency of these oscillators. The roll activates the gain on a distortion effect.

one EMG sensor. This allows a performer to 'select' a mix of the eight tones by tensing different muscles in their arm, for example, by holding their wrist in different positions. The frequencies of the eight oscillators were selected algorithmically with a base frequency controlled by the pitch (angle) of the Myo, and spread of the other frequencies controlled by the yaw. The roll angle was used to control the gain of a distortion effect that applied to the mix of all eight oscillators. This mapping is illustrated in Figure 3.

As mentioned previously, our compositional idea was based on the use of an 'inverse' mapping from action to sound, in which large movements result in no sound. This was accomplished by calculating the general *quantity of motion* (QoM) as the sum of the magnitudes of the current acceleration and gyroscope vectors. If this QoM measure rose above a certain threshold, a master amplitude parameter was sent to zero, effectively stopping the sound playback. After such a reset, the performer would need to become still again, starting a 30-second ramp to get back the full gain of the sound. This inverse mapping defines the 'standstill' nature of the composition; performers must hold their arm as still as possible in order to produce any sound. We found it best to keep the QoM threshold at a punishingly low level in order to force the performers not to move. Even then, some skilled performers were able to achieve very slow frequency sweeps using controlled movements.

So, given that this composition required the performers to stand still, how did they actually perform? What were they able to do to create musical structures and interact with each other? Low level structure in this composition–at the level of individual notes and articulations–was available through improvisation with the EMG sensors. Even though the performers were not able to move their arm position, they were still able to tense the muscles in their arm to activate different sounds. Although this interaction is limited, it requires focus to accomplish without moving so much that the standstill threshold was triggered.

The high level structure of the composition was defined

Figure 4: The premiere of *Stillness Under Tension*, a quartet standstill performance, in November 2017. An identical Pd patch was operating on four different Belas, each paired to one performer's Myo. Each performer could use their arm muscles to activate different oscillators, while their arm direction selected a different timbre and base frequency. A video of the performance is available: `https://vimeo.com/252904434/20f6e3e908`

by the performers' physical position. Although each Bela ran the same patch, it was natural that the performers took slightly different angles on stage. As each performer had a different arm position (as shown in Figure 4, the oscillators had different settings, resulting in a kind of harmonic variation between the performers' sounds. The distortion effect was controlled by the roll axis of the Myo, which the performers could control by rotating their forearm. A neutral arm position (Myo logo facing up) was matched to the lowest distortion, while the maximum amount of the effect was produced when the performers arms were inverted (palm facing up). This position requires more energy to hold still, particularly with an arm raised, which matched well with the noisy and intense sound that could be produced.

To structure performances of *Stillness Under Tension*, the performers alternated between different standstill poses, where they would eventually be able to explore sounds with their micromotion, while larger motion would immediately silence their Bela without disrupting the stillness of the other performers. The group agreed to assume around four poses over 12 minutes to enable each performer to explore a variety of sounds and to facilitate contrasts between the sonic possibilities of the composition.

5. CONCLUSION AND FUTURE WORK

This work has described the development and debut performance of a standstill ensemble performance work using the Myo armband and Bela. We have detailed how the Myo gestural interface may be integrated with the Bela embedded platform, a process that could also be useful for other embedded Linux computers such as Raspberry Pi. Our technical solution involved adapting the Bela's Pure Data integration to use an existing Linux library for the Myo. We also described our pure-Python solution for bridging Myo to OSC, which can be used for Bela-prototyping but also for performance with Linux computers. Finally, we described an ensemble standstill performance using the Myo's IMU and EMG to emphasise microinteraction while maintaining both low- and high-level structural variety. This performance relied on the integration with Bela to provide reliable connections to four Myos.

The completely self-contained technical solution is, perhaps, the greatest benefit of this research, and it has not, to date, failed us in either rehearsal or performance. This setup will allow us to focus mainly on musical issues in the future development of new compositions for our ensemble and for effortlessly scaling up with more performers.

Acknowledgements

This work was partially supported by the Research Council of Norway through the projects EPEC (#240862) and MICRO (#250698), and through its Centres of Excellence scheme (#262762).

6. REFERENCES

[1] C. Benson, B. Manaris, S. Stoudenmier, and T. Ward. Soundmorpheus: A myoelectric-sensor based interface for sound spatialization and shaping. In *Proc. Int. Conf. NIME*, volume 16 of *2220-4806*, pages 332–337, Brisbane, Australia, 2016. URL: `http://www.nime.org/proceedings/2016/nime2016_paper0065.pdf`.

[2] E. Berdahl and W. Ju. Satellite CCRMA: A musical interaction and sound synthesis platform. In *Proc. Int. Conf. NIME*, pages 173–178, Oslo, Norway, 2011. URL: `http://www.nime.org/proceedings/2011/nime2011_173.pdf`.

[3] B. Di Donato, J. Bullock, and J. Bledsoe. Myo Mapper: Myo armband to OSC mapper. Audio Developer Conf. (ADC), 2017. `doi:10.13140/RG.2.2.12505.62564`.

[4] B. Di Donato, J. Dooley, J. Hockman, J. Bullock, and S. Hall. Myospat: A hand-gesture controlled system for sound and light projections manipulation. In *Proc. ICMC*, Shanghai, China, 2017.

[5] A. R. Jensenius. Sonic Microinteraction in "the Air". In M. Lesaffre, P.-J. Maes, and M. Leman, editors, *The Routledge Companion to Embodied Music Interaction*, pages 431–439. Routledge, New York, 2017.

[6] A. R. Jensenius, K. A. V. Bjerkestrand, and V. Johnson. How still is still? Exploring human standstill for artistic applications. *Int. J. Arts and Technology*, 7(2/3):207–222, 2014. `doi:10.1504/IJART.2014.060943`.

[7] A. R. Jensenius, V. G. Sanchez, A. Zelechowska, and K. A. V. Bjerkestrand. Exploring the Myo controller for sonic microinteraction. In *Proc. Int. Conf. NIME*, pages 442–445, Copenhagen, Denmark, 2017. URL: `http://www.nime.org/proceedings/2017/nime2017_paper0083.pdf`.

[8] A. McPherson, R. Jack, and G. Moro. Action-sound latency: Are our tools fast enough? In *Proc. Int. Conf. NIME*, volume 16 of *2220-4806*, pages 20–25, Brisbane, Australia, 2016. URL: `http://www.nime.org/proceedings/2016/nime2016_paper0005.pdf`.

[9] G. Moro, A. Bin, R. H. Jack, C. Heinrichs, A. P. McPherson, et al. Making high-performance embedded instruments with Bela and Pure Data. In *Int. Conf. on Live Interfaces*, Sussex, UK, 2016.

[10] K. Nymoen, M. R. Haugen, and A. R. Jensenius. Mumyo - evaluating and exploring the myo armband for musical interaction. In *Proc. Int. Conf. NIME*, pages 215–218, Baton Rouge, Louisiana, USA, May 2015. URL: `http://www.nime.org/proceedings/2015/nime2015_179.pdf`.