EPFL

EMG decoding based on Arduino electrodes

Changling Li, supervised by Vincent Mendez, Translational Neural Engineering Lab

Abstract. Recent advances in EMG-based systems have been made with relative success to control upper-limb prostheses, electronic devices and machines. Nevertheless, the existing EMG-based systems are sometimes not intuitive for the user, uncomfortable to wear or not affordable. In this project we propose a cheap and easy to use system that utilizes 6 channel surface EMG (sEMG) electrodes with 6 MyoWare muscle sensors and Arduino UNO microcontroller as an analog to digital converter. The idea is to find a good setup and measure the quality of EMG signals for further applications, for instance to control a robotic hand prosthesis based on forearm muscle contractions. The obtained results show that these Myoware sensors could be as good as the Noraxon sensors, with SNR value up to 6.8.

1. Introduction

People with wrist disarticulation or transradial amputees have great difficulties to make use of their remaining forearm, specially for daily tasks that require most of the time finger and grasping motion. This decreases their quality of life and the existing hand prosthesis on the market are either uncomfortable or not intuitive to use. One of the current research in EMG field is looking into the feasibility of using EMG to decode the user's intentions, for example forearm band with surface EMG sensors to control a hand prosthesis. But the problem of high cost remain, thus not every amputee can benefit from these solutions. This is why in this work the focus is on MyoWare muscle sensors that are quite affordable, along with an Arduino UNO microcontroller for a system that is easy to setup.

These channels have a bipolar configuration; a pair of poles aligned along the muscle fiber direction reads the electric potential during the contraction, with a reference electrode for common-mode rejection .[1] The main works in this project are on how to setup and use this system and the measure of quality of these sensors. Then, the results will determine if it is indeed possible to use these sensors to create a flexible control system based on measuring forearm muscles activation via electric potential (referred as EMG). The ultimate goal of this project would be to predict the hand movement using sEMGs signals from 6 channels around the forearm to control a hand prosthesis.

2. Setup and Retrieving data from Arduino electrodes

2.1. Understanding how to setup the hardware stuff

The first step was to understand how to make the setup works. This system is composed of an Arduino UNO and its USB cable to connect to a laptop, 6 MyoWare Muscle sensors, 6 MyoWare cable shields, 6 three conductor sensor cable with disposable surface electrode, 6 MyoWare LED shields, a bread board and some jumper wires to connect all these stuff.

Fig. 1. Myoware muscle sensor and a reference electrode. [2]

First, one may understand that directly sticking this sensor to the skin (see Fig. 1) is not really comfortable for the user, let alone 6 sensors. Thus, the idea is to use cables with electrode pad leads sticked to the skin. To do so, the sensors are stacked on the Myoware cable shields by soldering connectors into the 3 pinholes (”R”/”E”/”M”) that correspond to the 3 electrodes. One can also stack LED shields on top to have a visual cue of the contractions but this is not interesting in this project.

To power these sensors, one has to connect one jumper wire to the 5V pin on the Arduino UNO to the bread board and another jumper wire to the ground (GND) pin to another line on the bread board. Then, the user has to connect one wire to the hole ”+” on the right of the sensor (see Fig. 1) to the bread board, next to the wire connected to the 5V pin. Same thing for the hole ”-” on the sensor, this time next to the wire connected to the GND pin. Repeat for the 5 other sensors. Finally, one has to connect the Arduino with the USB cable to a machine for example a laptop. This will supply 5V of power to these sensors.

Once the system is powered, it is time to place the electrodes. Three electrodes are needed for each sensor. One should be on the middle of the forearm, while the other line up more or less two centimeters further in the direction of the muscle. These pairs of electrodes can be placed in a linearly spaced array around the forearm without big concern for on the positioning accuracy to capture signals from several muscles simultaneously[1].

The last one, the reference electrode, should be placed on a bony of nonadjacent part of your body near the muscle. In this project the elbow is a good placement for the reference electrodes. One has to make sure that the skin is dry and clean before sticking these electrodes. [3]

Then, one can connect the jack plug of the cables to the shield and push the snap connectors to the electrodes. The colored one need to be on the forearm muscles while the black one on the elbow. One can test if the setup works by contracting the muscles, the red light in the middle of the sensor should respond to the muscle contraction and rest. The final setup should look like the Fig. 2 below (without the sensor on the skin).

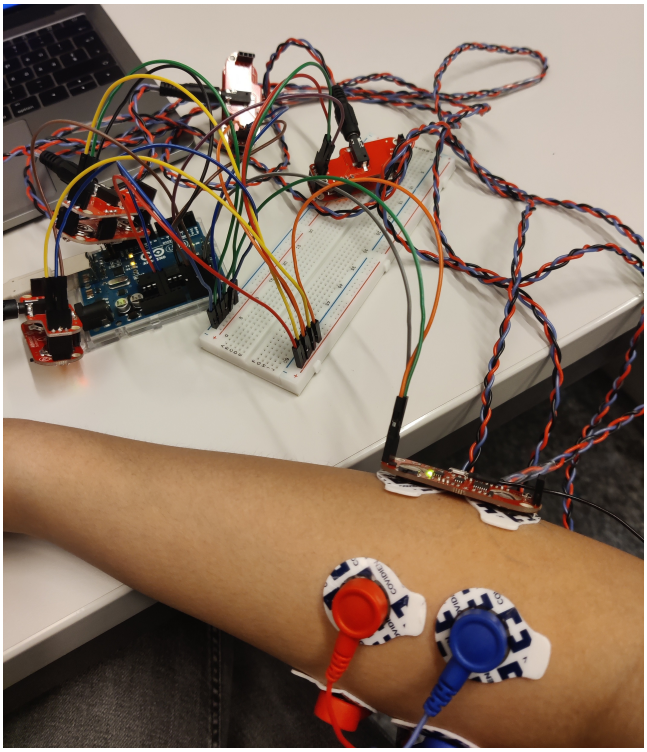

Fig. 2. Overview of the MyoWare setup.

2.2. Collecting and sending data to streams

Once the setup is ready, it is time to collect the data. There a two outputs mode on the sensors, the EMG envelope (denoted by SIG, see Fig. 1) and the raw EMG (denoted by RAW).

One can choose if they want to collect raw signals or the envelope or both. For the raw signals, a jumper wire must connect the ”RAW” pin on the end of the sensors to the A0-A5 analog pins on the Arduino. For the EMG envelope, the ”SIG” pin must be connected to external A6-A11 analog pins as there are only 6 analog pins on the Arduino.

To collect the EMG data, one should open the Arduino IDE, go to ”Tools” and select the board ”Arduino Uno” and the corresponding port that should look like ”/dev/cu.usbmodem14101” or ”COMx” on Windows. Then, the code ”sendbytes.py” msut be uploaded to the Arduino for it to execute indefinitely when it is powered. This code tells to the Arduino to sample the data and send them to the laptop.

One of the challenges was to make sure that the sampling frequency was set at 1000Hz. At first, the function Serial.print() was used to send the data into the computer but it was not great. The sampling frequency barely reached 500Hz as this function converts the data to ASCII characters before sending them to the laptop. Thus, it takes too much time. Thus, the data is directly sent to the machine as series of bytes and the frequency easily exceeded 1000Hz. To set the sampling frequency at 1000Hz, the code tells to the Arduino to wait for 1ms each time it samples a data.

Then, in order to collect the data and making it available and readable for any devices on the network, the bytes are decoded on the laptop and pushed to a stream outlet using the liblsl library. Moreover, a quick check was made on the time required to sample the data, transmit it to the laptop and decode the bytes, just before pushing them to the stream. As it was less than 20

milliseconds, the timestamps of the streams are then used instead of time given by the Arduino UNO.

One can install serial library by typing ”pip install Pyserial” in the terminal and run the code ”collectsendEMG.py” in the terminal to collect and send the data. The user needs to indicate to the terminal the Arduino port to retrieve the data. Then, the user can choose if he/she wants the raw EMG data, the EMG envelopes or both. If both are wanted, two separate streams will be created for each type of data. This is why the user needs to connect the wires to the correct analog pins (A0 to A5 for raw data, A6 to A11 for the envelope) as previously explained, otherwise there would be a mix of raw and processed data sent in the streams.

The user must check that nothing touches the soldering points on the sensors during the experiments and be careful not to move the cables too much, otherwise it would disrupt the EMG signals. They are easily overwhelmed by electrical disturbances[1].

3. Quality of signals

Now that we are able to retrieve the data at a sampling rate of 1000Hz, it is time to see if the EMG signals acquired with the Myoware sensors are good enough for future applications.

Just as a reminder, the Myoware sensors can output amplified, rectified and integrated signals (EMG's envelope), and amplified raw signals. It is possible to adjust the amplification of the EMG envelope by turning the potentiometer (denoted by ”Gain”, see Fig. 1) with a screwdriver on the sensor, just below the ”GND” hole.

3.1. Signal pre-processing and measure of quality

In order to measure the quality of the EMG signals, the signal-to-noise ratio (SNR) for both outputs of the sensors are quantified. To do so, the envelope and the raw signals are recorded at the same time with one channel on the forearm, on the extensor digitorum muscle.[4] At the beginning, the muscle is at rest, then a long contraction is made by making a fist. Here is a graph with both outputs :

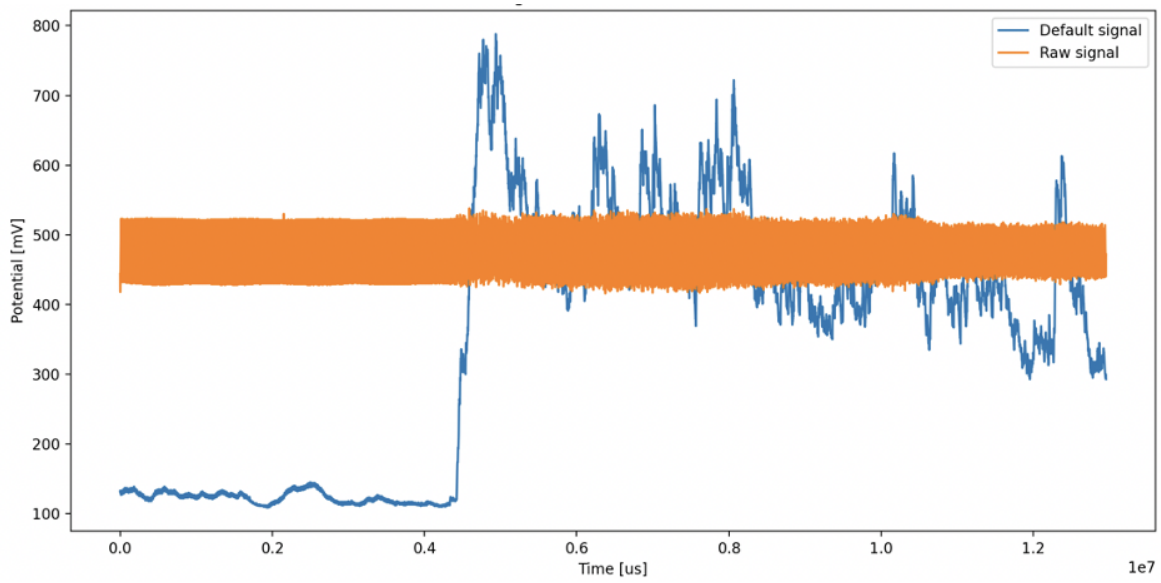

Fig. 3. Default signal (EMG's envelope) and raw signal acquired at the same time with cables and one channel on forearm muscle. The contraction begins around 4.3 seconds.

To compute the SNR, $\hat{E}_r$ is defined as the time-averaged values of EMG data during the resting motion and $\hat{E}_c$ the time-averaged values during the contraction. Thus, the SNR ratio is calculated as follows [5] :

$$SNR = \frac{\hat{E}_c}{\hat{E}_r} \tag{1}$$

The SNR of default signal is around 3.8 and the rectified raw signal around 1. The default signal seems to be better than the raw on which is not surprising as the noises were still there in the raw data. To see if it possible to improve the SNR by pre-processing the raw signal, a Notch filter is used to remove the power line interference at 50Hz and its harmonics. A band-pass filter of 15 to 495 Hz is also applied to reduce external noises and making sure that only the EMG signal remains, with rectification. An amplification was not necessary as the raw EMG signal output is already amplified by the sensor. The cables for each are braided in order to reduce their movements, and thus the noise.

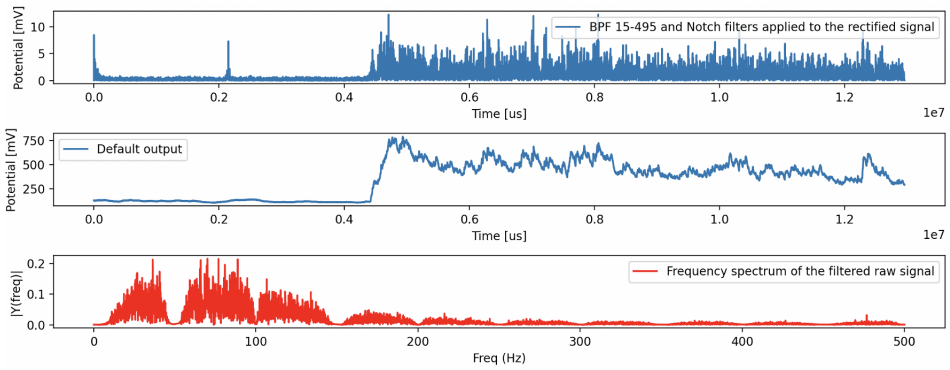

Fig. 4. Default signal (EMG's envelope), filtered and rectified "raw" signals sampled at the same time with cables and one channel on forearm muscle. Notch filter at 50Hz, BPF 15-495Hz and rectification are applied on the raw signal. Last graph is the frequency spectrum of the filtered EMG data.

This time, the SNR of the filtered signal is around 6.86, which is really good and even better than the default output of the MYoware sensor. This means that one could use the Myoware sensors for future applications by processing the raw EMG signal for better results than with the already computed EMG envelope. However, in this experiment the contraction was really strong, thus giving high EMG values. This is probably not representative of simple finger movement, in the case of predicting the motion of fingers using the EMG signals from the forearm muscles.

3.2. Comparison Noraxon vs. Myoware

Now that the SNR is calculated, a comparison between Myoware and Noraxon sensors was done to assess the quality of MyoWare EMG data. Noraxon sensors are indeed better and more professional for medical use than Myoware. In this experiment, a hand choregraphy was recorded with different grasp types: Rest, Open, Power grasp, Ulnar grasp and pinch grasp, with 6 EMG channels at 2kHz and continuous finger angle at 60Hz with Noraxon sensors. This choregraphy was also made With Myoware 6 EMG channels at 1kHz and for both outputs, the raw data and the envelope. Then, each recording gives one dataset that consists of EMG signals and another with finger kinematics (values proportional to finger angles) to be able to link user intention and EMG signals. Finger angles are saved as a matrix where rows correspond to time and columns correspond to one joint (Thump opposition, Thumb, Index, Middle, Ring, Pinky), as you can see in Figure 5 below.

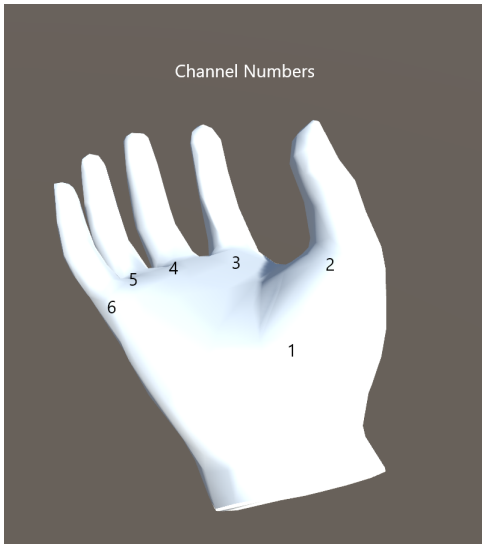

Fig. 5. Hand model where each joint are denoted by its number.

In order to compare both signals, the Pearson correlation coefficient was calculated for each EMG channel (precisely the envelope) against each finger kinematic to see the difference of quality between these two sensors. To obtain the envelope a Notch filter at 50Hz, a band-pass filter 5-500 Hz, a rectification and finally a low-pass filter at 3Hz are applied to the raw data.

This gives us 3 tables of 36 coefficients :

	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
F1	-0.048444	0.018039	0.016941	-0.008527	0.007317	-0.067707
F2	0.138276	0.118679	0.135048	0.042777	0.122543	0.239093
F3	0.039411	0.036723	0.093793	-0.022395	0.041693	0.149426
F4	0.098625	0.096723	0.136258	-0.046579	0.062347	0.175254
F5	0.075568	-0.040768	0.001786	-0.071447	-0.039111	0.119913
F6	0.098345	-0.057871	0.022847	-0.016182	-0.062143	0.127154

Fig. 6. Noraxon correlation coefficients saved as matrix where rows correspond to finger kinematics and columns to computed EMG envelopes of raw data.

	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
F1	-0.011906	0.000556	-0.007897	0.027853	-0.123155	-0.052851
F2	0.044014	-0.021813	0.084168	0.069059	0.234158	0.084755
F3	-0.015332	-0.090590	-0.005122	-0.032401	0.119379	-0.010742
F4	0.041326	-0.089944	0.061088	0.013847	0.183570	0.040933
F5	-0.043206	-0.095088	0.025619	-0.053396	0.176262	0.067434
F6	-0.056607	-0.122498	-0.008167	-0.083058	0.164048	0.016308

Fig. 7. Myoware correlation coefficients saved as matrix where rows correspond to finger kinematics and columns to EMG envelopes.

	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6
F1	0.012175	0.017444	-0.050542	0.052205	-0.039152	-0.096690
F2	0.072539	0.092063	0.108573	0.177039	0.047730	0.046356
F3	-0.002180	0.013597	0.008162	0.097811	-0.039406	-0.019737
F4	-0.076376	0.033373	0.058800	0.073662	0.048092	0.027253
F5	-0.091568	-0.097005	0.026091	0.035325	0.036845	-0.008149
F6	-0.100329	-0.106374	0.031447	0.049449	0.011685	0.003566

Fig. 8. Myoware correlation coefficients saved as matrix where rows correspond to finger kinematics and columns to EMG envelopes of raw data.

There are not many significant coefficients, but one could tell that the Myoware channel 5 in Fig. 7 is the equivalent of Noraxon channel 6 thanks to the highest coefficient 0.23 for the thumb (finger 2). Which means that the electrodes were placed on the same muscle for both channels. By focusing on the column values of these channels, one can see that the coefficients of Myoware

channel 5 are pretty much in the same range as the Noraxon channel 6, if not a bit better. This could mean there is almost no difference of quality between both sensors, which is surprising with the cheap Myoware sensors and handmade soldering. However, as the Noraxon recording was the first one, the user was still not really familiar with the choregraphy, so there are wrong hand gestures at some points in the EMG data, which could lower the correlation coefficients.

For the comparison between Noraxon and raw EMG of Myoware, this time it is the Myoware channel 4 (in Fig. 6) that is equivalent to the channel 6 of Noraxon, because of the coefficient of 0.17 in finger 2. One could see that the correlation coefficients are lower this time for the Myoware. This is maybe due to the fact that the user was still not really familiar with the hand choregraphy during the Myoware recording (because the Myoware recording with the envelope was the last one). But also the soldering points on each Myoware sensors were probably in contact with other objects, for instance the cables. This gives raw EMG data with lots of noise due to the movements of cables during the choregraphy or bad contacts with the soldering points. We can see on the Figure 11 below that even after the filtering, the Myoware still has lot of noise that could support these assumptions.

Nonetheless, one could assume that without these noise the resulting correlation coefficients would be better and probably in the same range as the Noraxon coefficients. But one must do another recording to check if it is indeed the case.

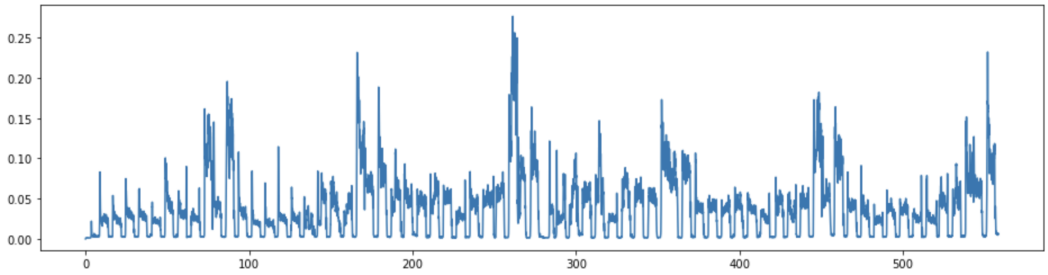

Fig. 9. Graph of Noraxon EMG channel 6 envelope after filtering the raw data.

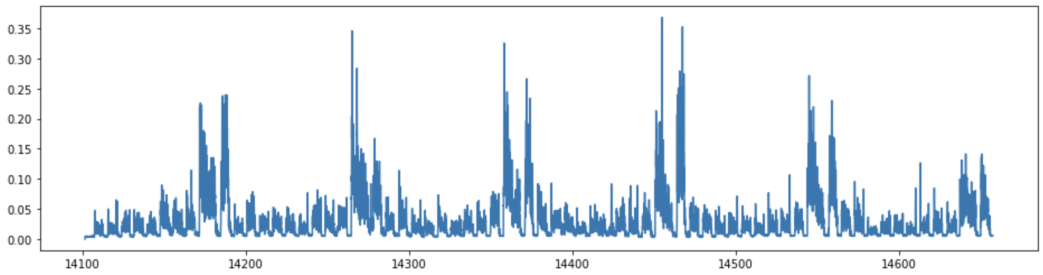

Fig. 10. Graph of Myoware channel 5 envelope directly given from the sensor.

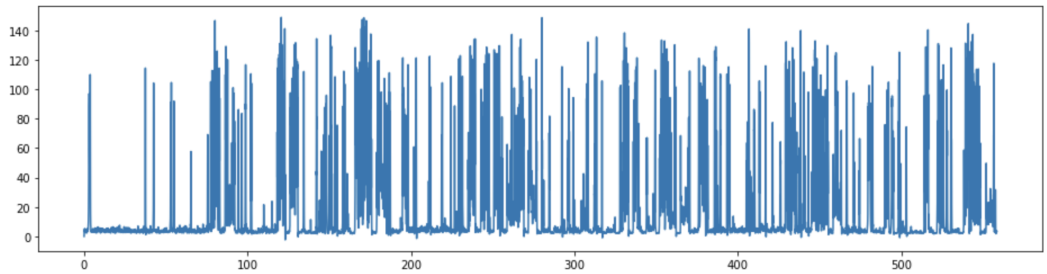

Fig. 11. Graph of Myoware channel 4 envelope after filtering the raw data.

4. Conclusion

In overall, the MyoWare Muscle sensors could be used to predict finger and grasp motions by doing a regression of EMG data using the signals provided by these sensors. The EMG signals have good SNR and there was little difference in signal quality with the Noraxon sensors. One should maybe do another hand choregraphy recording. These results give confidence for further applications such as regression. Nonetheless, care must be taken to ensure that the sensors are not touching anything. Moreover, there are too much cables to manage in this setup. It would be better to assemble them tightly together in a tube to reduce their micro movements.

References

[1] M. Simão, N. Mendes, O. Gibaru and P. Neto, "A Review on Electromyography Decoding and Pattern Recognition for Human-Machine Interaction," in IEEE Access, vol. 7, pp. 39564-39582, 2019, doi: 10.1109/ACCESS.2019.2906584

[2] MyoWare Muscle sensor, Sparkfun, https://www.sparkfun.com/products/13723

[3] Advancer Technologies, "Myoware User Manual : 3-lead Muscle / Electromyography, Sensor for Microcontroller Applications" (2015)
http://cdn.sparkfun.com/datasheets/Sensors/Biometric/MyowareUserManualAT-04-001.pdf

[4] Jones, O. "Muscles in the Posterior Compartment of the Forearm" Last updated : March 19, 2020 https://teachmeanatomy.info/upper-limb/muscles/posterior-forearm/

[5] Togo, S., Murai, Y., Jiang, Y. et al. Development of an sEMG sensor composed of two-layered conductive silicone with different carbon concentrations. Sci Rep 9, 13996 (2019). https://doi.org/10.1038/s41598-019-50112-4