Raspberry PI based electromyography signal acquisition and processing system.

Author - Toshesh Meena,
Co-author - Vidhi Sharma, Sushil Jhakar, Sunil Kumar Sharma
Swami Keshvanand Institute of Technology, Department of electrical engineering, Jaipur, Rajasthan, India.

Abstract : This paper describes the development of a portable system using Raspberry Pi 3B+ system on chip platform for physiological signal acquisition and processing. For programming Python language is used because of availability of large number of open-source libraries and multi platform compatibility. This portable system along with processed results can be used for designing graphical interfaces for effective vital signal analysis and diseases diagnosis. Moreover, it shows a lot of scope for developing effective human machine interface. For application demonstration we use Python libraries for acquisition and processing of surface electromyography (SEMG) data. Obtained results are used to show a comparison between SEMG data for six different forearm exercises. Along with that we also demonstrate human machine interface development by controlling servos (representing actuators in prosthetics) using SEMG data.

Index Terms- Surface electromyography, Raspberry Pi 3B+, Myoware SEMG sensor, Python programming, Physiological signal.

I. INTRODUCTION

The Raspberry Pi is a very powerful computing device having the dimensions of a business card. Raspberry Pi board costs only $40 and does the work of computer costing hundreds of dollars, although its purpose is not to replace traditional computers but to supplement them, perfect for use in compact processing projects like this [1]. Electromyogram (EMG) signals are a measure of the electrical activity of muscles. There are two types of sensors that can be used to record this electrical activity, in particular surface EMG (sEMG), measured by non-invasive electrodes, and intramuscular EMG. Out of the two, sEMG allows for non-invasive electrodes to be applied at the body surface, that measure muscle activity.Electromyogram (EMG) signals are a measure of the electrical activity of muscles. There are two types of sensors that can be used to record this electrical activity, in particular surface EMG (sEMG), measured by non-invasive electrodes, and intramuscular EMG. Out of the two, sEMG allows for non-invasive electrodes to be applied at the body surface, that measure muscle activity [2]. Biomedical signals express wide range of distinctions in time and frequency domain. Python is primary language used for processing. Python is interpreter- and object-oriented program language that is widely used among academicians for educational and research purposes [3]. This approach should allow for a more flexible system for academic studies. For example, A patient wearing the proposed system could be helped during a assisted rehabilitation and myoelectric prosthetic designing.

II. HARDWARE

The system's hardware is composed of 3 parts, physiological sensor attached to Arduino UNO board along with Raspberry Pi 3B+ board performing primary signal processing. Major hardware components are further explained in detail below -

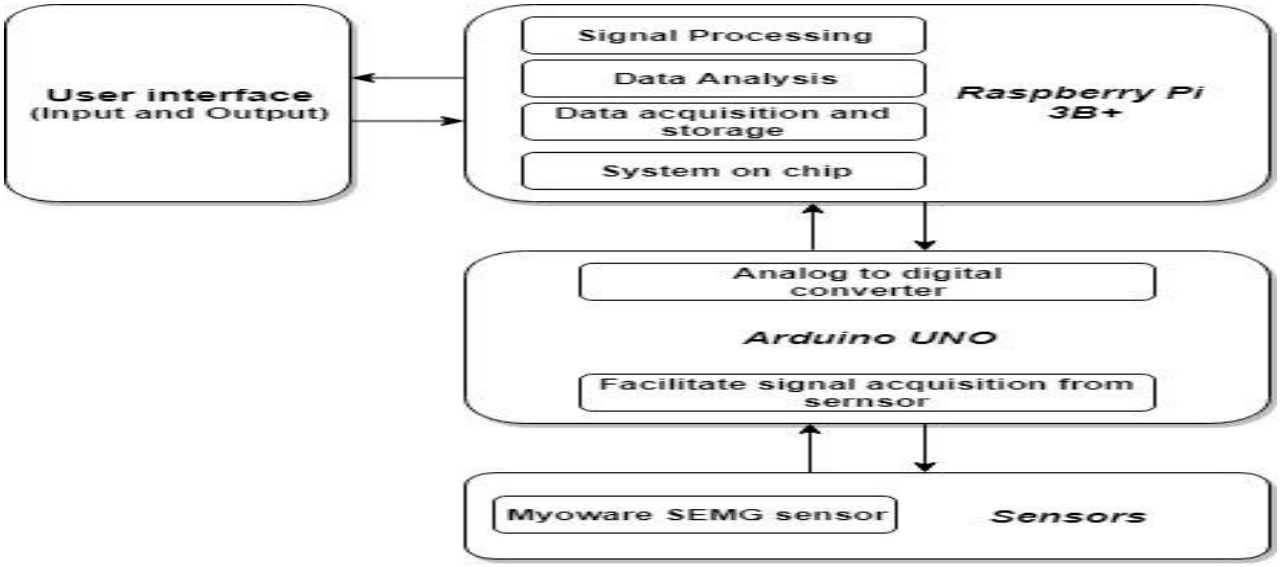

Figure-1: Block diagram showing working of portable system with Myoware SEMG sensor.

2.1 Raspberry Pi 3B+

Raspberry Pi platform is used primary processing and user interaction. Raspberry Pi which has 1.4 GHz processor with 1GB SDRAM is more than capable to handle the load and moreover it is attached to a 7 inch display and a mini wifi keyboard for user input [4].

2.2 Arduino UNO

Arduino board is used for signal acquisition and data transfer to raspberry pi. Most Sensors used for physiological signal acquisition generate analog output but Raspberry Pi is only capable of reading digital signals and has no way to read analog signal. That's why we use Arduino board as ADC (analog-to-digital converter) [5]. Arduino is connected to Raspberry Pi using a proprietary cable and is working is master-slave configuration, where Arduino is working as slave and Raspberry Pi is working as master. Arduino IDE is used to communicate between the two devices.

2.3 Myoware SEMG sensor

The described portable system in this paper can be used for acquiring several types of physiological data. For demonstration we are using Myoware SEMG sensor for acquiring forearm SEMG data [6]. This sensor provides a easy way to measure EMG data. The sensor has 3 muscle electrodes and those are mid muscle, end muscle and reference electrodes [7]. These electrodes measure the muscle action potential and send it to Arduino board. The sensor is connected to Arduino board using 3 cables. Terminals 'SIG', '+Vs', 'GND' of sensor are connected to terminals 'A0', '+Vs', 'GND' of Arduino board respectively. In similar manner other sensors can be attached to Arduino board to acquire signal.

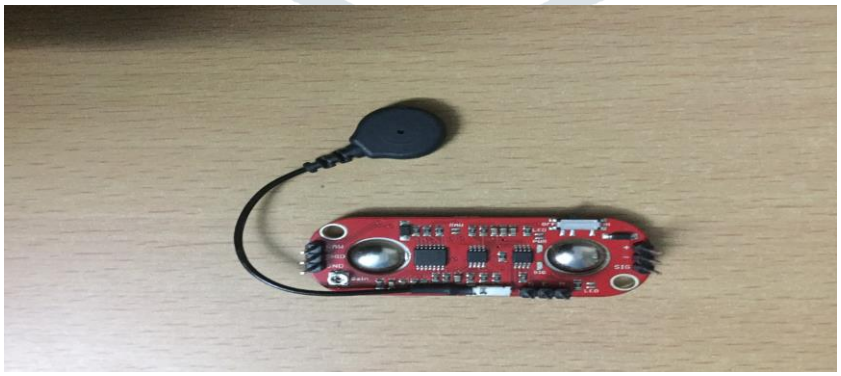

Figure-2: Myoware SEMG sensor.

2.4 SG90 servo

As already stated above the portable system can also be used for development of effective human machine interface such building of cheap myoelectric prosthetics. To demonstrate that SG90 is used as actuator for prosthetic arm. SG90 is a micro sized servo with

torque of 1.80 Kg-cm and speed of 0.12 sec/60 deg [8]. Servo is connected to Arduino board and mapped to muscle sensor to replicate arm motion.

Figure-3: Complete Raspberry Pi and Arduino setup along with the SEMG sensor.

III. PROGRAMMING AND SOFTWARE

Programming is done in Python 3 (for Raspberry Pi) and C (for Arduino UNO).

3.1 Python 3

Programming in Python 3 is done to acquire data from Arduino, Save that data in .csv and .txt files, Design filters and functions to process the data, Represent data graphically and much more. For EMG data processing, two methods can be used. Eithier by custom butterworth bandpass filter or by using BioSPPy Python library [9].

3.2 Arduino IDE 1.8.8

Programs written in C language are used in Arduino IDE 1.8.8. They are written to achieve serial communication between Arduino board and Raspberry Pi, Acquire data from sensors and control movement of servos [10].

IV. METHODOLOGY

The basic methodology for using this system is described in steps as follow-

(1) First step is to establish serial communication between the Arduino and Raspberry Pi. here we define the serial connection port in both devices. here we are using the serial port with 9600 baud rate. To confirm proper function of serial communication we can use Python scripts to save serial data in .txt or .csv file and also plot a real time graph of data with time [11].

(2) Second step is to attach the sensor to Arduino board, in our case we are attaching a Myoware SEMG sensor. Using serial communication we continuously import the data readings from sensor into Raspberry Pi and using a Python script we plot a real time graph of data (some latency can be observed in output due to speed of communication).

(3) Now Python scripts are used to process the data. As defined above we are using a SEMG sensor, so we will demonstrate processing of its data in two different ways-

(a) First ways is by using a customized python script using the basic python libraries. Here, we designed a custom butterworth band pass filter with following settings.

Low-pass cut off frequency = 450 Hz
High-pass cut off frequency = 20 Hz
Sampling frequency = 1000 Hz

The butterworth function created using scipy butter function (library function) is of 4th order. Results achieved are discussed in next section [12].

(b) Another Method for data processing is use of a specialized library, BioSPPY (Biological signal processing in python). BioSPPY is a toolbox for biosignal processing written in Python. The toolbox bundles together various signal processing and pattern recognition methods geared towards the analysis of biosignals [13]. It includes sub-packages which provide methods to process common physiological signals. Right now it can process six types of signals-

- Blood volume pulse,
- Electrocardiogram,
- Respiration,
- Electromyogram,
- Electrodermal Activity,
- Electroencephalogram

Here we are using EMG sub-package to process the data. For example for following code-

biosppy.signals.emg.emg(*signal=None, sampling_rate=1000.0, show=True*)

Input parameters are signal (*array*) – Raw EMG signal, sampling_rate (*int, float, optional*) – Sampling frequency (Hz), show (*bool, optional*) – If True, show a summary plot.

Output parameters are ts (*array*) – Signal time axis reference (seconds), filtered (*array*) – Filtered EMG signal, onsets (*array*) – Indices of EMG pulse onsets.

More specific codes can be used to achieve desired results and graphically showing those results.

For Second demonstration of this systems capabilities to help in designing effective human machine interface and cheaper prosthetics. A servo arm is connected to Arduino along with the Myoware muscle sensor. The reading from muscle sensor are mapped to the servo position using an Arduino code. As output of this code, muscle sensor reads the muscle movements from arm and relay them to servo, in short the servo position replicates the arm position effectively demonstrating the systems capabilities and usage possibilities.

V. OBSERVATIONS AND RESULTS

For taking SEMG data readings we are using a combination of six different forearm exercises performed for a period of six seconds. For observation each of six exercise was performed five times with an interval of five seconds. There was a total of five test subjects used, four male and one female all on age group 20-22 year. The exercises performed are-

- Cylindrical grasp
- Tip
- Hook or Snap
- Palmer
- Spherical
- Lateral

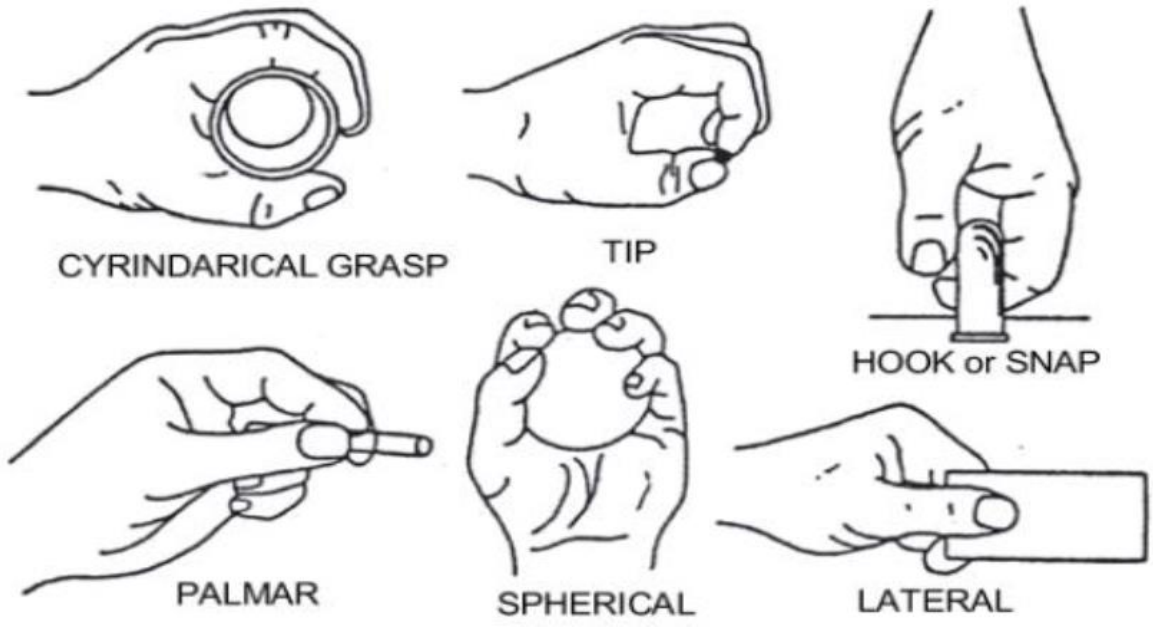

Figure-4: Shows six different forearm SEMG exercises.

The data recorded for each exercise is sorted separately and mean values are calculated for each one of them. Following table and graph shows the comparison between different exercises [14].

Exercise	**Mean value (Hz)**
Cylindrical grasp	227
Tip	155
Hook or Snap	383
Palmer	145
Spherical	359
Lateral	160

Table-1: Mean SEMG values for different exercise.

The data recorded is of rectified EMG, so the rms and mean values are same. Above recorded data is shown graphically to provide a better comparison between different exercises. From above we can see that hook and spherical type of exercises have maximum mean values.

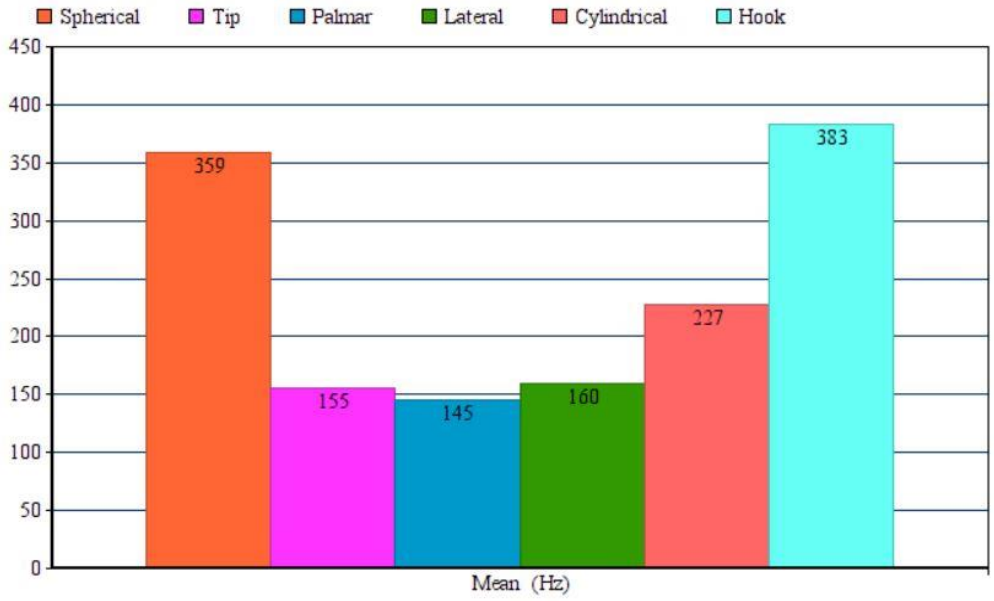

Figure-5: Shows six different forearm SEMG exercises graphical comparison.

Data signal acquired from sensor is rectified already. It is processed filtered and mean value calculation.

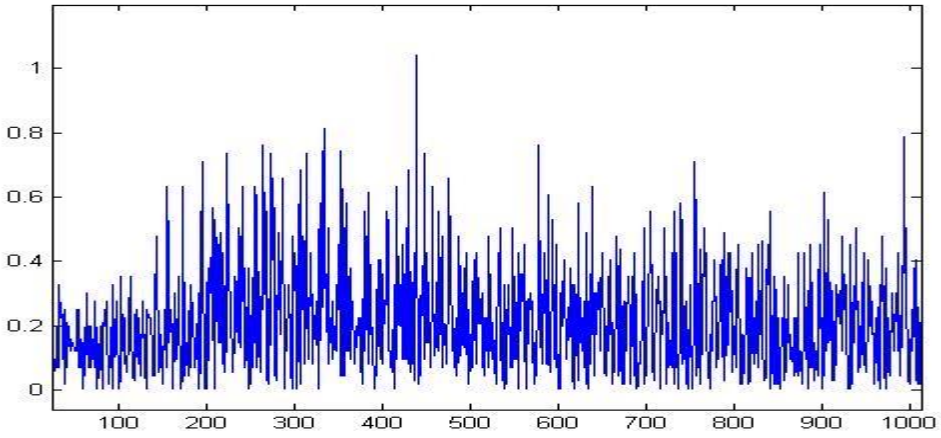

Figure-6: Plot of data acquired from sensor.

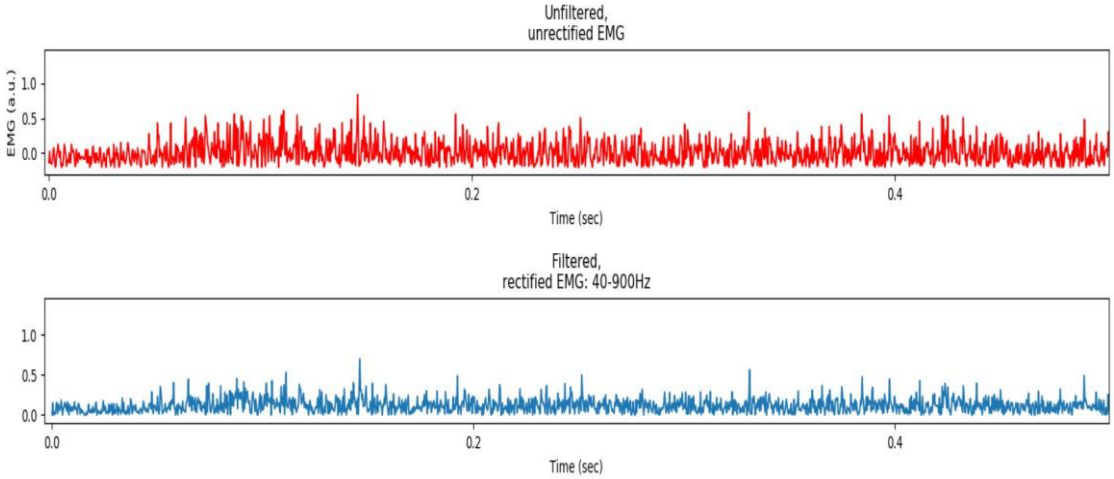

Figure-7: Graph shows comparison between unfiltered (red) and filtered (blue) signal.

VI. Conclusion and Future Scope

In conclusion we can say that the designed system and demonstrated applications effectively showcase the capabilities of the system and its future scope of development. The use of BioSPPy toolbox allows less experienced users to use the device and achieve outputs. Also Due to addition of an Arduino board the drawback of all digital signals in Raspberry Pi is taken care of too. Further Python scripts can be built to perform signal classification, data clustering and even pattern recognition. In field of Human machine interfaces too this device can prove to a very useful because of its small size and open source platform usage such as gesture/ muscle control or automatic alerts for any physiological reading. This device can also be used in development of cheaper myoelectric prosthetics [15] [16]. And most importantly the scope of usage is unlimited because with a different set of libraries and scripts this system can be used for a totally different operation [17] [18].

References

[1]Raspberry Pi specification and details https://www.raspberrypi.org/ on March 2019.

[2]F. Hug, "Can muscle coordination be precisely studied by surface electromyography?" J. Electromyogr. Kinesiol., vol. 21, no. 1, pp. 1–12, Feb. 2011.

[3]C. Day. "Python Power", Computing in Science & Engineering, Vol 16, Issue 1, pp. 88–88, 2014.

[4]https://projects.raspberrypi.org/en/pathways/getting-started-with-raspberry-pi. as on March 2019.

[5]Concept of ADC https://circuitdigest.com/microcontroller-projects/arduino-uno-adc-tutorial as on March 2019.

[6]F. Hug, "Can muscle coordination be precisely studied by surface electromyography?" J. Electromyogr. Kinesiol., vol. 21, no. 1, pp. 1–12, Feb. 2011.

[7]Myoware sensor usage https://learn.sparkfun.com/tutorials/myoware-muscle-sensor-kit?_ga=2.219397853.57939867.1553535012-511941039.1552671449#myoware-muscle-sensor as on March 2019.

[8]Arduino and servo communication https://www.arduino.cc/en/Reference/Servo as on march 2019.

[9]BioSPPY library documentation https://biosppy.readthedocs.io/en/stable/index.html as on March 2019.

[10]Arduino 1.8.8 IDE https://www.arduino.cc/en/Reference/ as on march 2019.

[11]Using pySerial to Read Serial Data Output from Arduino https://engineersportal.com/blog/2018/2/25/python-datalogger-reading-the-serial-output-from-arduino-to-analyze-data-using-pyserial as on March 2019.

[12]PYTHON: ANALYSING EMG SIGNALS https://scientificallysound.org/2016/08/22/python-analysing-emg-signals-part-1/ as on March 2019.

[13]This module provides methods to process Electromyographic (EMG) signals https://biosppy.readthedocs.io/en/stable/biosppy.signals.html#biosppy-signals-emg as on March 2019.

[14]sEMG for Basic Hand movements https://archive.ics.uci.edu/ml/datasets/sEMG+for+Basic+Hand+movements as on March 2019.

[15]A. L. Goldberger, et al, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals," Circulation.101(23):e215-22, June 2000.

[16]A. M. Gavrovska, "Combined Measurements: Tools for Cardiac Related Analysis," BAW 2015 (Brain Awareness Week) Second Conference Human-Machine Interface from Student-to-Student Interface, Belgrade, March, 2015.

[17]Komi, P. V., and Tesch, P., "EMG frequency spectrum, muscle structure, and fatigue during dynamic contractions in man." European Journal of Applied Physiology 42, 41–50, 1979 .

[18]R. Dias and J.M. Silva, "A Flexible Wearable Sensor Network for Biosignals and Human Activity Monitoring", 11th International Conference on Wearable and Implantable Body Sensor Networks Workshops, 2014.