University of Dayton
eCommons

Honors Theses

University Honors Program

4-2018

The Application of Neurologically-Controlled Robotics to Actuated Feeding Arms

Timothy Dombrowski

Follow this and additional works at: https://ecommons.udayton.edu/uhp_theses

Part of the Electrical and Computer Engineering Commons

eCommons Citation

The Application of Neurologically-Controlled Robotics to Actuated Feeding Arms

Honors Thesis

Timothy Dombrowski

Department: Electrical Engineering

Advisors: Raul Ordonez, Ph.D, S.M, Electrical and

Computer Engineering Department

Temesguen Kebede, Ph.D, S.M., Robotics Laboratory

April 2018

The Application of Neurologically-Controlled Robotics to Actuated Feeding Arms

Honors Thesis

Timothy Dombrowski

Department: Electrical Engineering

Advisors: Raul Ordonez, Ph.D, S.M, Electrical and

Computer Engineering Department

Temesguen Kebede, Ph.D, S.M., Robotics Laboratory

April 2018

Abstract

The medical industry is constantly performing research and trying to combat various diseases that afflict humans. Despite advancements in technology, there still remain diseases that have no cure but seem prime candidates for neurologically controlled robots. One such category of diseases consists of various muscular dystrophic diseases. Diseases such as ALS and Parkinson's have limited options regarding treatment, but by brain controlled interfaces (BCI's), robotics can help mitigate the impact on a patient's quality of life. By utilizing a functioning mind, an electroencephalographic (EEG) helmet can be used to control various exoskeletal systems and even prosthesis in order to compensate for a damaged motor system. Through the use of neurologically controlled robotics, a user's motor control and motor strength can be rehabilitated and maintained despite the effects of muscular dystrophic diseases. The goal of this project is to use this experiment to demonstrate the current effectiveness of brain actuated robotics and telepresence that utilize an EEG Sensor. From this assessment, recommendations and further improvements can be made to this existing technology for it to be better suited for electrical engineering and biomedical applications, while simultaneously taking the technology into a new realm of application.

UNIVERSITY *of*

DAYTON

Table of Contents

Introduction

For this experiment, the initial goal was to use electroencephalographic (EEG) sensor technology in a new realm of application. At its roots, EEG technology utilizes brain wave outputs as inputs in a computer interface to actuate mechanical systems [8]. The three main waves utilized in this fashion are alpha, beta, and gamma waves of the brain [14]. From this research project, advances in the utility and ease of implementation of EEG technology were desired. Multiple approaches and methodologies were considered when utilizing electroencephalographically obtained data. With EEG technology not being commonplace in everyday applications, any technical insight would prove beneficial to the industry.

Literature Review

Throughout any undertaking of experimentation in the realm of electroencephalographic technology, it was important to consider the numerous aspects of the technology, both positive and negative. The largest benefit of EEG technology is that a successful setup can compensate for impaired portions and even entire systems of the body. So long as the mind is fully functioning, EEG sensors can be used to bridge the gap in physical functionality of a user. The complexity of this application is limited solely by the type of sensor involved and a user's ability to control either mental states or amplitude of brainwaves, depending on precisely how the EEG technology is implemented. Once perfected, EEG sensors can seamlessly pass commands to a mechanical system, matching a user's thought input. The other large benefit of

electroencephalographic technology is that each user can also have a customized profile that can be adjusted to create a better fit.

Naturally, when using EEG technology with direct input from the human brain, one of the constraining factors is the brain itself. The human brain is not capable of multitasking and can only actively focus on one task at a time [11]. To compensate for this, the brain actively switches from one task to another numerous times a second. The result from this can be problematic when utilizing data, as this causes all types of brainwaves to undergo rapid changes on the millisecond scale. Suffice to say, such rapid spikes in brain activity often have ramifications, the most dire of which being a large amount of interference and noise data recordings. To counteract this, most applications of EEG technology should utilize some sort of filter or rejection criteria [7]. Filters and rejection criteria actively limit the number of erroneous inputs provided to an EEG device with the brain. These analytical tools mend some of the shortcomings of EEG technology. Additionally, these tools allow EEG sensors an amount of pliability to be attached to each user, enhancing and providing a customized experience and reading.

While EEG sensors possess a great amount of utility, there are some drawbacks associated with the technology that should be taken into account. First and foremost, the largest limitation of the technology was tied to the number of nodes attached to the sensor. The Mindwave headset utilizes two nodes: one that is used to sense and register brainwaves, one to act as a ground. The active node attaches to the forehead while the

ground node connects to the user's ear lobe (Figure 1).

Figure 1: Neurosky Mindwave
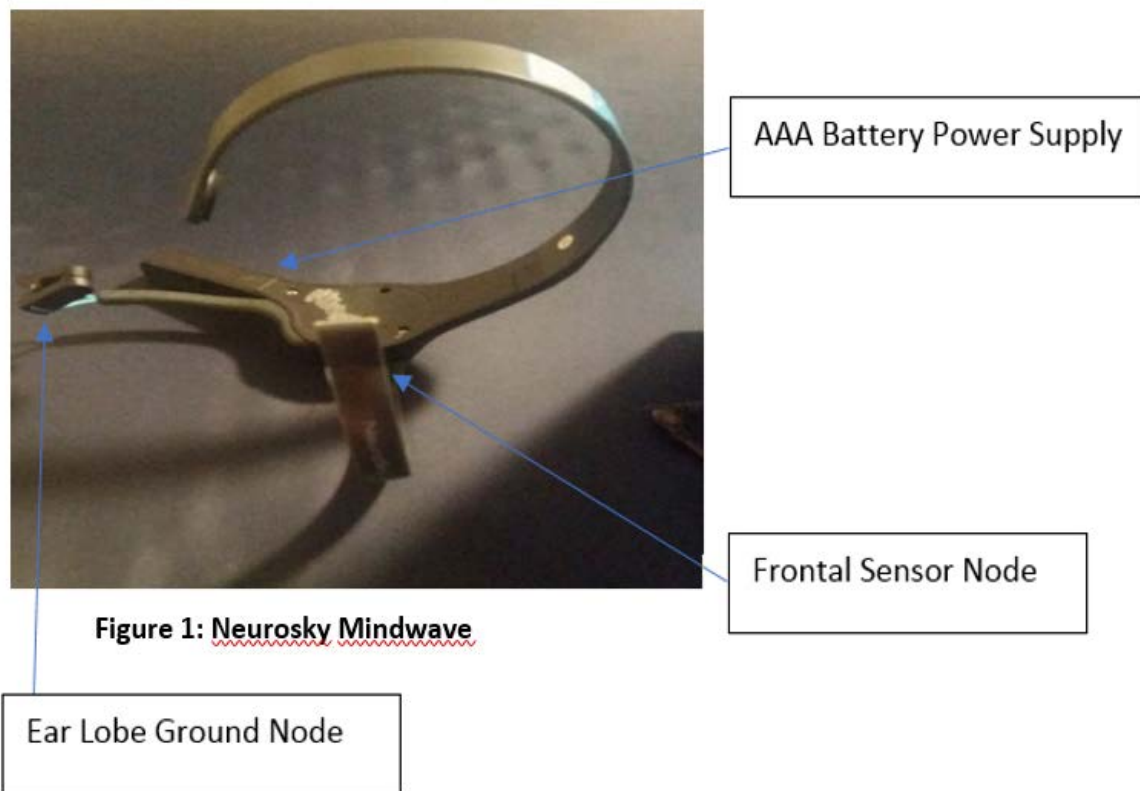
As with any analytical approach, the more sources of information that are available, the more conviction can be placed upon the results. As stated, the Neurosky Mindwave possesses only the two nodes. Depending on the depth of study, additional nodes may be required for improved precision and accuracy of readings. Different authors' approach to the number of nodes can be found in the Literature Review section. Not only does the number of nodes affect the quality of readings, but a lack of sufficient nodes can severely limit the technical approaches that can be taken to utilizing EEG technology. There are two main ways to implement EEG technology. One possible approach is to create commands according to which regions of the brain are activated by specific thought processes [7]. These commands can be established by associating certain mental states with desired inputs to a system based on the type of cognitive thinking required to

activate each mental state. Such states are created via a statistical classifier and would

need to be customized for each user and calibrated before each trial [8]. An example of

this is controlling a robotic car through a maze and other directed control applications [7].

Another potential method is to hard code commands that rely on the relative strengths of

brainwaves, which was the approach utilized by this work. The benefit of this approach is

that it does not require a piece of software to associate different mental states with

command inputs. This approach is still rather unique, as most applications tend to favor

the idea of associating mental states [Millán, 14]. This style of approach is also limited by

the number of different brainwaves and their relative magnitudes in association with one

another. Mental states is instead limited by the number of regions that can be

independently activated by a user's mental activity. When deciding between these two

approaches, it is important to weigh these limitations.

Based on extensive background research, the largest caveat to this project was the

general belief that everyday applications of EEG technology would be difficult due to the

rapid thought processing of the brain, and the amount of noise generated during thought

processes [12]. From Foerster's study of brain computer interfaces, 8 sensory nodes were

utilized in addition to a ground node and a reference node [2].

Methods

For this application, a Neurosky Mindwave EEG sensor was first used as a

preliminary tool to record and interpret brain wavelengths. These wavelengths were then

broadcasted to a computer via Bluetooth connectivity and visually monitored. This data

was collaborated into Brainwave Visualizer, a software developed by Neurosky that specialized in visual management of brainwave data. From the sensor input, the Brainwave Visualizer software was able to show the relative magnitude of the various brain wavelengths. This software was able to demonstrate the following brain wavelengths: High Alpha, Low Alpha, High Beta, Low Beta, High Gamma, Low Gamma, Delta, and Theta. These wavelengths were all measured through an electrode placed on the forehead of the user (Figure 2). These initial readings of wavelengths were then imported into a Microsoft Excel file for preliminary data analysis (Figure 2).
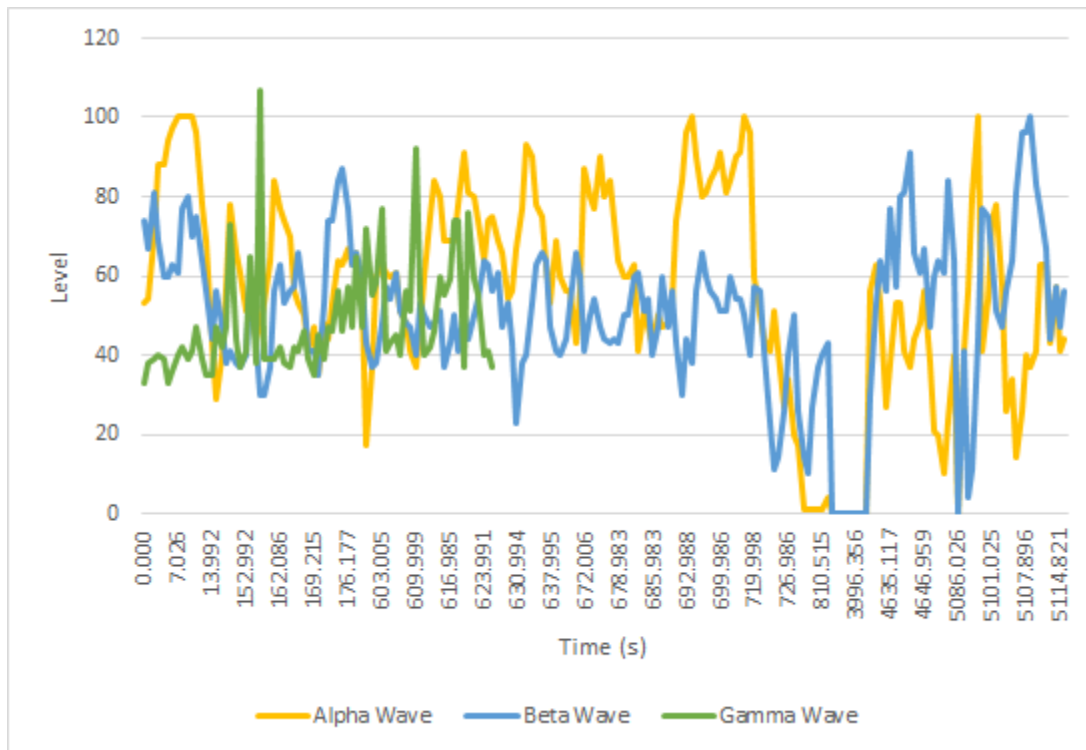

Figure 2: Preliminary Sensor Data

Of all the data collected, alpha, beta, and gamma waves were considered as they are the result of more easily replicative brain processes. For example, alpha waves are associated with wakeful rest and could dominate the input to the computer simply by having the user close their eyes and imagine a relaxing scene [5,12]. Beta waves are more closely associated with concentration, so any task that actively required focused attention or brainpower could cause beta waves to be the dominant signal to the sensor [14]. Gamma waves are commonly associated with mental states related to prayer, meditation, and other methods of deep contemplation. These waves can be seen visually in Figure 2. The relative magnitude of each wave can also be seen on the y axis of the graph.

This initial testing method, although beneficial for being able to take multiple sets of data very rapidly, was far from ideal. The goal of this experiment was to demonstrate using EEG data as potential inputs for a mechanical system, so the fact that each wavelength varied intensely was problematic from the perspective of obtaining usable data in order to establish a proof of concept. To further improve upon the data quality and utility, a method that could automatically apply a moving average filter to the data. A moving average filter is useful because it takes the average value of all samples within a specified interval range [4]. This average value reduces variance in the data and would be a more proper representation of the data set as a whole. The equation for a moving average can be seen below:

$$x_{MA} = \frac{1}{n} \sum_{i=0}^{n-1} x_M - i$$

(Equation 1)

Where n is the number of terms within a given interval, x_M is the value of the given

wavelength. This equation allows for the calculation of a new sum at every point,

dropping the oldest term in exchange for the newest one and calculating the new sum

each time.

During experimentation, an Arduino Uno was used in order to automatically collect

brainwave data. To complement the Arduino microcontroller, an RN-41 Bluetooth

Module was integrated into the circuit to be able to pair with the Neurosky Mindwave

(Figure 3).
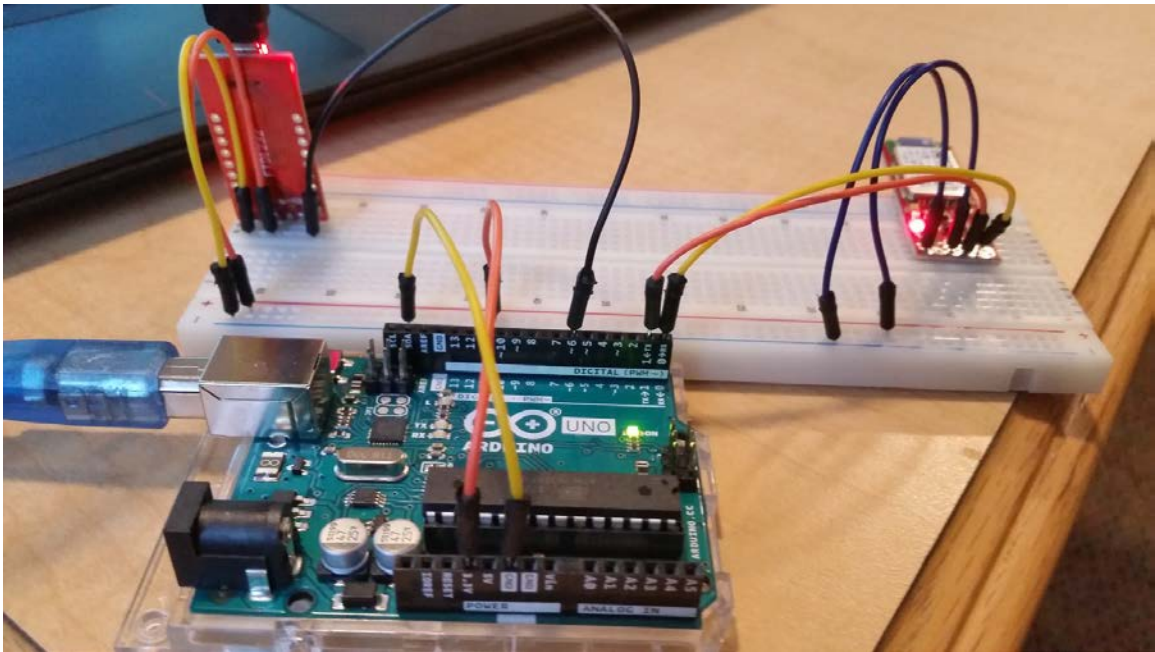

Figure 3: Arduino Circuit Controller Setup

The function of the circuit is for the EEG sensor to pass data to the Bluetooth Module.

The Bluetooth Module then incorporates the serial monitor from the Arduino software to

display the amplitude of each type of brainwave. The Arduino is also passed the code to

be able to apply a moving average filter to the data as it comes in (Appendix).

Results

Over the course of the project, a lot was discovered in the realm of initializing an electroencephalographic sensor into a new area of application. The largest and most impactful result of this experiment was the utility and suitability of the Neurosky Mindwave for a potentially commercial application. As stated previously, the number of electrodes plays a large role in the quality of data readings. It was found that the Neurosky Mindwave was not suitable for an Arduino based application. The Neurosky had numerous connectivity issues regarding both the RN-41 Bluetooth Module and the Arduino software interface. Despite the Bluetooth Module recognizing and connecting the Neurosky Mindwave EEG sensor and the Arduino successfully implementing code and recognizes the Bluetooth Module in its port, the Arduino does not successfully display output the data that should be recorded via the Mindwave. This was potentially a compatibility issue with the Neurosky Mindwave and the Arduino. This issue presented complications when it came to post data analysis. Due to troubles with the EEG sensor, the Arduino code associated with automatically recording all active brainwave measurements and applying a moving average filter to each brainwave was not able to be properly implemented. The brainwave readings that suffered from technical difficulties can be seen below (Figure 4).
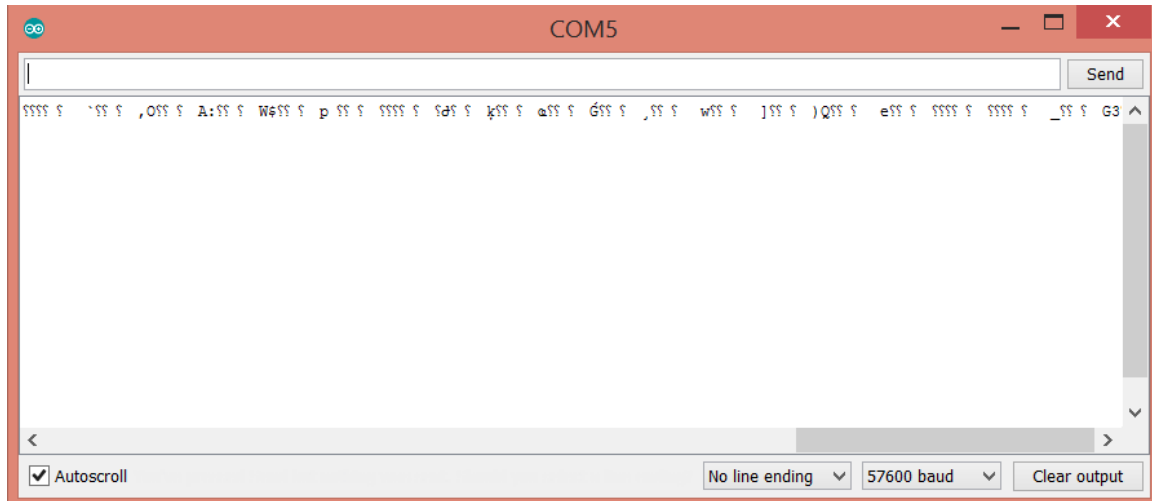
Figure 4: Erroneous Serial Output of EEG Sensor via Arduino

Discussion

This work serves as a prime example of the fundamental challenges of applying EEG technology to simple life tasks today. Despite how simple it may appear to attempt to establish a conceptual design that utilizes EEG sensors, the simple fact is that the technology may still not be ready for off the shelf commercial applications. As such, it is recommended that alternative approaches still be considered for EEG applications. Alternative methods may offer more troubleshooting capabilities and better availability of

Conclusion

Despite the lack of results of this approach, this experiment served to provide many benefits to the electroencephalographic industry. For example, it was learned that strategies that utilize mental state were proven to be very successful as seen in Millán's and Foersters experiments with brain computer interfaces [2,6-8]. Brain computer

interfaces (BCI's) supplement the capabilities of EEG technologies by bringing additional processing power to the situation. Computers can be used to automatically create mental states based on specific types of thinking that activate different regions of the brain [5,6,8,10]. Such examples could include multiplication, mentally solving a Rubix cube, crushing a box, or relaxation. These mental states can then be associated with commands to be used as inputs into a mechanical system. Such brain computer interfaces can be more easily implemented as it takes part of the burden of operation off the user and utilizes additional software to create criteria for input commands. Going forward, it would be beneficial to incorporate a sensor with more electrodes to read and send signals to a brain computer interface. It would also be beneficial to utilize additional software capable of creating mental states and linking mental associations with desired mechanical outputs to mitigate the burden on potential users. Only when these obstacles presented are removed can electroencephalographic technology be applied to a robotic feeding arm to the point where any user can successfully utilize such technology to a high degree of accuracy. Once such a method is implemented can an approach begin to integrate EEG technology into a new realm of commercial application.

Works Cited

[1] Escolano, C., Antelis, J. M., & Minguez, J. (2012). A telepresence mobile robot controlled with a noninvasive brain-computer interface. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *42*(3), 793–804. https://doi.org/10.1109/TSMCB.2011.2177968

[2] Foerster, M., Bonnet, S., Van Langhenhove, A., Porcherot, J., & Charvet, G. (2013). A synchronization method for wireless acquisition systems, application to brain computer interfaces. *Engineering in Medicine and Biology Society (EMBC), 2013 35th Annual International Conference of the IEEE*, 830–833.

[3] Kim, Y. S., Baek, H. J., Kim, J. S., Lee, H. B., Choi, J. M., & Park, K. S. (2009). Helmet-based physiological signal monitoring system. *European Journal of Applied Physiology*, *105*(3), 365–372. https://doi.org/10.1007/s00421-008-0912-6

[4] Mayur Agarwal, Ashutosh Mishra, Swapna Banerjee, "VLSI architecture for IEEE single precision floating point moving average calculator", *Power and Advanced Computing Technologies (i-PACT) 2017 Innovations in*, pp. 1-4, 2017.

[5] Member, S., Antelis, J. M., Andrea, K., & Minguez, J. (2009). A Noninvasive Brain-Actuated Wheelchair Based on a P300 Neurophysiological Protocol and Automated Navigation, *25*(3), 614–627. https://doi.org/10.1109/tro.2009.2020347

[6] Millán, J. D. R., Ferrez, P. W., Galán, F., Lew, E., & Chavarriaga, R. (2007). Non-invasive brain-actuated interaction. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *4729 LNCS*, 438–447. https://doi.org/10.1007/978-3-540-75555-5_42

[7] Millán, J. D. R., Renkens, F., Mouriño, J., & Gerstner, W. (2004). Noninvasive brain-actuated control of a mobile robot by human EEG. *IEEE Transactions on Biomedical Engineering*, *51*(6), 1026–1033. https://doi.org/10.1109/TBME.2004.827086

[8] Millán, J. del R., Renkens, F., Mouriño, J., & Gerstner, W. (2004). Brain-actuated interaction. *Artificial Intelligence*, *159*(1–2), 241–259. https://doi.org/10.1016/j.artint.2004.05.008

[9] Oehler, M., Neumann, P., Becker, M., Curio, G., & Schilling, M. (2008). Extraction of SSVEP signals of a capacitive EEG helmet for human machine interface. *Conference Proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, *2008*(factor 2), 4495–4498. https://doi.org/10.1109/IEMBS.2008.4650211

[10] P, M. L. A., & Rob, M. G. (2013). Analysis and Classification of Electroencephalographic Signals (EEG) to Identify Arm Movements, 138–143.

[11] Rosen, Christine. "The Myth of Multitasking." *The New Atlantis*, no. 20, 2008, pp. 105–110., www.jstor.org/stable/43152412.

[12] Rosenberg, W. V. O. N., & Member, S. (2016). Smart Helmet : Wearable Multichannel ECG and EEG. *IEEE Journal of Translational Engineering in Health and Medicine*, *4*(August). https://doi.org/10.1109/JTEHM.2016.2609927

[13] Soleymani, M. (2013). Imperial College London , UK Imperial College London , UK University of Twente , the Netherlands, *231287*(231287).

[14] Teplan, M. (2002). Fundamentals of EEG measurement. *Measurement Science Review*, *2*(2), 1–11. https://doi.org/10.1021/pr070350l

[14] Yosephine, V. (n.d.). LEGO Robotics Based Project for Industrial Engineering

Education. The International Journal of Engineering Education.

Appendix

Arduino Code:

```
/*

  Brain wave interface

  Connections:

  Bluetooth silver mate to Uno: Rx --> Tx

                    Tx --> Rx

  Once configured, Neurosky mindwave connects to the bluetooth module automatically.

https://learn.sparkfun.com/tutorials/hackers-in-residence---hacking-mindwave-

      mobile?_ga=1.192828522.1192467422.1459118391

  Software serial is used to push the data serially.

  --------------------------------------------------------------------------------

  This example code is in the public domain.

  modified 7 Dec 2017

  by Manoj Sharma

  imanoj_sharma@yahoo.com

  https://twitter.com/irahulone

  --------------------------------------------------------------------------------



*/



//////////////////////////////////////////////////////////////////////
```

```
#include <SoftwareSerial.h>     // library for software serial

SoftwareSerial mySerial(5, 6);  // RX, TX

int LED = 13; // onboard led connected to pin #13

int BAUDRATE = 57600;

// checksum variables

byte payloadChecksum = 0;

byte CalculatedChecksum;

byte checksum = 0;          //data type byte stores an 8-bit unsigned number, from 0 to
        255

int payloadLength = 0;

byte payloadData[64] = {0};

byte poorQuality = 0;

int attention = 0;

int meditation = 0;

// system variables

long lastReceivedPacket = 0;

boolean bigPacket = false;

boolean brainwave = false;

int attTh = 60;   // Set the threshold b/w 0 & 100

void setup()

{

  pinMode(LED, OUTPUT);

  digitalWrite(LED, HIGH);   // hello sequence
```

```
  //delay(100);

  digitalWrite(LED, LOW);

  //delay(100);

  Serial.begin(57600);       // Bluetooth

  delay(500);

  mySerial.begin(57600);      // software serial

  delay(500);

  mySerial.print("Communicating... ");

  mySerial.println();

}

byte ReadOneByte()

{

  int ByteRead;

 // Wait until there is data

  while(!Serial.available()); //mySerial.println(millis());

  //Get the number of bytes (characters) available for reading from the serial port.

  //This is data that's already arrived and stored in the serial receive buffer (which holds

       64 bytes)

  ByteRead = Serial.read(); //mySerial.println(ByteRead);

  return ByteRead; // read incoming serial data

  Serial.flush();

  }

unsigned int delta_wave = 0;
```

```
unsigned int theta_wave = 0;

unsigned int low_alpha_wave = 0;

unsigned int high_alpha_wave = 0;

unsigned int low_beta_wave = 0;

unsigned int high_beta_wave = 0;

unsigned int low_gamma_wave = 0;

unsigned int mid_gamma_wave = 0;

void read_waves(int i) {

  delta_wave = read_3byte_int(i);

  i+=3;

  theta_wave = read_3byte_int(i);

  i+=3;

  low_alpha_wave = read_3byte_int(i);

  i+=3;

  high_alpha_wave = read_3byte_int(i);

  i+=3;

  low_beta_wave = read_3byte_int(i);

  i+=3;

  high_beta_wave = read_3byte_int(i);

  i+=3;

  low_gamma_wave = read_3byte_int(i);

  i+=3;

  mid_gamma_wave = read_3byte_int(i);
```

```
}

int read_3byte_int(int i) {

  return ((payloadData[i] << 16) + (payloadData[i+1] << 8) + payloadData[i+2]);

}

void(* resetFunc) (void) = 0;

void loop()

{

 // Look for sync bytes

  // Byte order: 0xAA, 0xAA, payloadLength, payloadData,

  // Checksum (sum all the bytes of payload, take lowest 8 bits, then bit inverse on lowest

if(ReadOneByte() == 0xAA) {

if(ReadOneByte() == 0xAA) {

payloadLength = ReadOneByte();

if(payloadLength > 169) //Payload length can not be greater than 169

return;

payloadChecksum = 0;

    for(int i = 0; i < payloadLength; i++) {      //loop until payload length is complete

      payloadData[i] = ReadOneByte();           //Read payload

      payloadChecksum += payloadData[i];

    }

    checksum = ReadOneByte();                 //Read checksum byte from stream

    payloadChecksum = 255 - payloadChecksum;     //Take one's compliment of

        generated checksum
```

```
        if(checksum == payloadChecksum) {

          poorQuality = 200;

          //attention = 0;

          //meditation = 0;

    }

      brainwave = false;

    for(int i = 0; i < payloadLength; i++) { // Parse the payload

        switch (payloadData[i]) {

        case 02:

          i++;

          poorQuality = payloadData[i];

          bigPacket = true;

          break;

        case 04:

          i++;

          attention = payloadData[i];

          break;

        case 05:

          i++;

          meditation = payloadData[i];

          break;

        case 0x80:

          i = i + 3;
```

```
    break;

  case 0x83:                    // ASIC EEG POWER INT

    i++;

    brainwave = true;

    byte vlen = payloadData[i];

    //mySerial.print(vlen, DEC);

    //mySerial.println();

    read_waves(i+1);

    i += vlen; // i = i + vlen

    break;

  }                          // switch

}                          // for loop

if(bigPacket) {

  if(poorQuality == 0){

  }

  else{                    // do nothing

  }

}

    if(brainwave && attention > 0 && attention < 100)

    {

      // Here the values are sent ove the software serial

      mySerial.print(attention); mySerial.print(", ");

      mySerial.print(meditation); mySerial.print(", ");
```

```
        mySerial.println();

    }

    }

    }
//////////////////////////////////////////////////

    if(attention >= attTh)

    {

      digitalWrite(LED, 1);

    }

    else

      digitalWrite(LED, LOW);

    //digitalWrite(LED, 0); delay(100);

    //digitalWrite(LED, 1); delay(100);
//////////////////////////////////////////////////

}
```

Moving Average Filter.

Example to implement a real time moving average filter, using a potentiometer

as an input to the system.

Connections:

Connect the output of a potentiometer to pin A0.

This example code is in the public domain.

modified 9 Feb 2018

```
  by Manoj Sharma

  imanoj_sharma@yahoo.com

  @irahulone

  --------------------------------------------------------------------------------
*/

const int window_size = 100;  // define the window size.

const int refPosPin = A0;    // input pin

int refPosAr[window_size];

float ref = 0;

void setup()

{

  Serial.begin(9600);

}

void loop()

{

  int p = sizeof(refPosAr)/sizeof(int);  // window size

  refPosAr[p] = analogRead(refPosPin);

  float a = 0;

  for(int i = 1; i <= p; i++)

  {

    a += refPosAr[i];

  }

  a = a/p;
```

```
  ref = a;

  for(int i = 2; i <= p; i++)

   {

    refPosAr[i-1] = refPosAr[i];

   }

  Serial.println(ref);

 }
```