WIRELESS CONTROL OF A ROBOTIC ARM USING 3D MOTION TRACKING SENSORS AND ARTIFICIAL NEURAL NETWORKS

Fernando Ríos, Georgia Southern University; Rocío Alba-Flores, Georgia Southern University;
Imani Augusma, Georgia Southern University

Abstract

In this paper, the authors describe the hardware and software components of an intelligent system that is able to wirelessly control the movements of a robotic arm for mimicking human arm gestures. For the implementation of the system, a laptop computer, 3D wireless motion tracking sensors, an artificial neural network (ANN) classifier, and a microcontroller were used to drive the six-degree-of-freedom robotic arm. Results demonstrated that the robotic arm is capable of mimicking motions of the human arm. The overall accuracy of the ANN classification system was 88.8%. Due to limitations of non-continuous rotation servos, some movements had to be limited or changed in order for the robotic arm to perform as an equivalent to a human arm.

Introduction

Robotic technologies have played and will continue to play important roles in helping to solve real-life problems. One of the most important fields in the development of successful robotic systems is the human-machine interaction (HMI). In this paper, the authors describe the development of a system that uses an ANN classifier to control a robotic arm that is able to mimic the movements of a human arm. In this study, the user was able to directly control a six-degree-of-freedom (6-DOF) robotic arm by performing arm motions with his/her own arm. The system uses inertial measurement units to sense the movements of the human arm.

Alternative approaches that have been used to develop human-machine interaction include the use of electromyography (EMG) signals to capture and analyze electrical activity in human muscle tissue [1, 2]. However, due to the electrical signals being minuscule, processing the data using this method is difficult. Other techniques that have been used include gyroscopes and accelerometers. For example, Sekhar et al. [3] developed a low-cost wireless motion sensing control unit using three sensors: accelerometer, gyroscope, and magnetometer. They used a three-degree-of-freedom robotic arm to control the elbow and wrist positions. Matlab software was used to process the signals coming from the sensors and generate the pulse width modulation (PWM) signals to control the servomotors; the accuracy of the developed system was not specified. An alternate approach that recently has started to gain popularity among researchers is to track muscle activity using inertial measurement units (IMUs) and air pressure sensors [4, 5]. IMUs integrate an accelerometer, a gyroscope, and a magnetometer together to measure three-directional static and dynamic movements. Malegam and D'Silva [6] developed a mimicking robotic hand-arm using flex sensors for individual fingers and multiple three-axis accelerometers. Using four encoders, they divided individual processing units for the fingers and arm to increase the processing speed. They also used a high-speed microcontroller to control the input and output processing, then developed a glove to house all of the components for a user to wear.

Tracking System Operation

In this current study, the authors designed and developed a wireless control system to give commands to a robotic arm. The commands were given by a human subject wearing two IMUs on his/her arm. Figure 1 shows the selected IMU location. The IMU contained an accelerometer, a gyroscope and a filter in a small unit [7]. The robotic arm had six degrees of freedom and could perform elbow, wrist, and shoulder joint movements. Figure 2 shows the robotic arm used in this study. Kalman filtering was also integrated into the IMU software to reduce potential noise and to produce smooth signal data.

Figure 1. Subject Wearing the Two Inertial Measurement Units (IMUs)

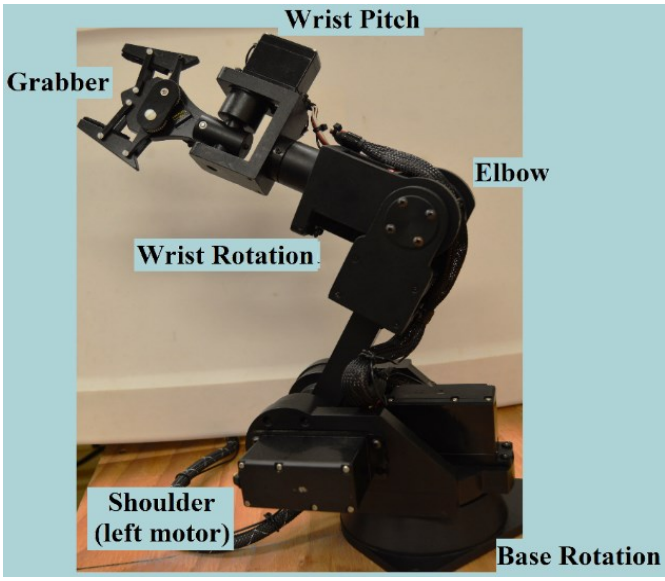


Figure 2. Six-DOF Robotic Manipulator

After obtaining movement activity information from the IMUs, the data were then fed into a trained ANN. An ANN is an adaptive and powerful artificial intelligence (AI) technique that is used to classify the inputs of a biological system. The ANN has the ability to recognize both linear and nonlinear relationships between input and output data, similar to the human brain. Because of this, ANNs are widely used for data classification and pattern recognition. Figure 3 shows the basic structure of an ANN.

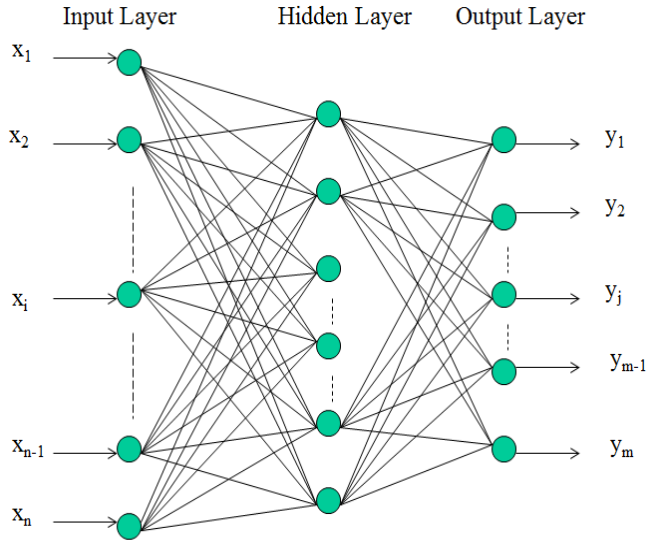


Figure 3. Schematic Diagram of a Multilayer Feed-Forward Neural Network

The ANN processes information using various layers that are linked together: the input layer, the hidden layer, and the output layer. Each layer is composed of interconnected nodes that represent neurons. Data are fed into the input layer, which connects to the hidden layer, and the hidden layer connects that output to the output layer. All of the connections are weighted, and individual weights are modified as the network is trained. The ANN learns by example, using an algorithm called backpropagation, also known as the backwards propagation error. The ANN receives input data repeatedly and then makes a guess about the corresponding output and then compares it to the actual output. The hidden layer computes an error that will be fed back into the network to adjust the weights. Each input and hidden layer neuron's value is multiplied by a predetermined weight. The weights are meant to minimize the error as much as possible to minimize misclassifications.

The weighted input layer and the weighted hidden layer are then summed together. If the summation does not equal one, then adjustments will occur during each cycle, or "epoch," until the summation is as close to one as possible, which means the error cannot be minimized further and this input corresponds to an output. This is called training the ANN. Rote memorization can occur if the network is trained to recognize only one type of input. This is called over-training the system. For an ANN to work properly, it must be trained with various types of input data to a desired output. After training the system, it will be able to see new input data and adjust the weights accordingly in order to produce an accurate output [8-11]. The ANN then determines the corresponding movement that is performed by the user, based on the test set that was used to train the ANN. After the network decides the movement, this information is sent to the robotic arm to emulate the human arm motion.

Main System Components

Figure 1 shows the 3D wireless motion tracking sensors IMUs [7] that were placed on the human subject's arm at two locations—the wrist and the upper arm. At a sampling rate of 100 Hz, the sensors tracked the XYZ-coordinates, inertial data, and the Euler angles of the subject's arm as he/she performed a specific movement. Nine pre-defined arm motions were selected for detection in this study. The raw data (XYZ-coordinates from the IMUs) were processed computing the root mean square (RMS) and the average rectified value (ARV). Normalized data were then used to train a multilayer, feed-forward ANN to classify the arm motions. The Matlab Neural Network toolbox software was used for the design and implementation of the ANN classifier. An Arduino microcontroller was used to control the servo motors in the robotic arm. The Arduino was directly interfaced to the laptop computer implementing the ANN classifier. The data set used for training the ANN in this

study consisted of 180 data vectors (from four different subjects, each of which performed each motion five times). Seventy percent of the data was used for training the ANN, 10% for validation, and 20% for testing. A totally independent set of arm motions (from a 5th subject) was used to determine the accuracy of the ANN classification system.

The main components used in the implementation of the system were: a) an IMU board composed of a digital three-axis accelerometer and a digital three-axis gyroscope [7]; b) a ZigBee RF wireless communication module [12] to transmit and receive data; c) a low-cost microcontroller, Arduino Mega [13], to control the input and output processing; d) a 6-DOF robotic arm that used servo motors to control the joint positions (see again Figure 2)—the servos were controlled using PWM signals; and, e) a Kalman filter that was used to reduce noise and have smooth signal data from the accelerometers and the gyroscope. Figure 4 shows the interaction of these components.

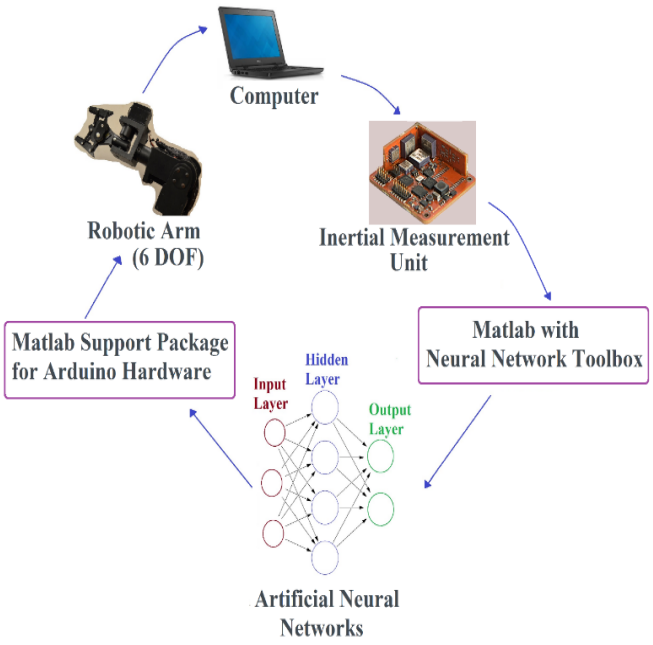


Figure 4. Main Components of the Wireless Robotic Control System

Methods

The human subject performed one of the pre-defined arm movements and Xsens Technologies' [7] software captured the waveform of the acceleration data. These data were then exported into an Excel spreadsheet and imported into Matlab. The sampling rate of the sensors was 100 Hz and each arm movement took approximately three seconds to perform. After the sensor data were collected, Matlab functions were used to calculate the RMS and ARV values. These values are then fed into the trained ANN. The ANN then determined the corresponding arm movement that was performed by the human subject. The corresponding arm movement was then performed by the robotic arm.

Description of the Arm Movement

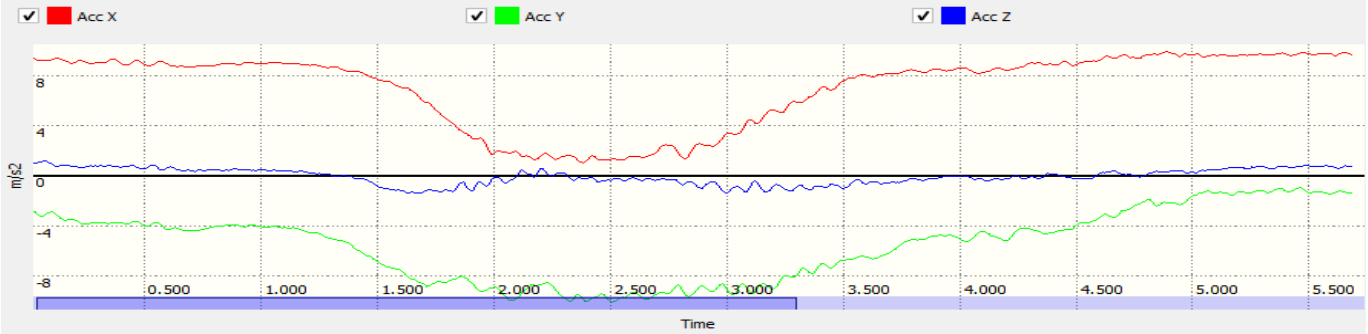
To capture the arm's movements, two IMUs were used. Figure 2 shows that sensor 1 was placed on the person's upper arm and sensor 2 was placed closest to the wrist. Sensor 1 signifies the unit on the person's upper arm and sensor 2 signifies the unit on the person's wrist. There was a total of nine pre-defined arm movements that the ANN could identify and the robotic arm could mimic—arm extension, arm raise, arm raise elbow bend, clockwise windmill, counterclockwise windmill, shoulder touch, side arm raise, wipe right, and wrist rotation. Figure 1 shows that the arm movement used the same initial position—straight down by the person's side, fingers pointing to the floor. For all of the following descriptions of arm movements, the initial starting position was for the person to be standing, the arm and hand fully extended on the side of the body, and the palm facing the body.

Motion 1: Arm Extension. When performing the arm extension movement from the starting position, the person first bends the elbow until the forearm is parallel to the floor, then extends the full arm all the way forward, maintaining it parallel to the floor, and then returns to the initial position. Figure 5 shows the acceleration waveforms obtained during the arm extension motion.

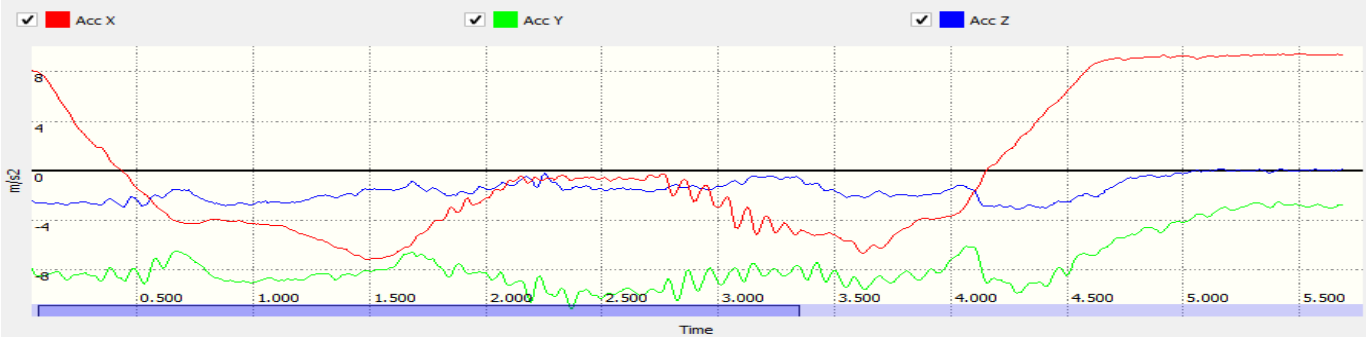
Motion 2: Arm Raise. When performing the arm raise movement, the person starts in the initial position. Without bending the elbow, the person raises the arm up to shoulder height and then returns to the initial position. Figure 6 shows the acceleration waveforms obtained during the arm raise motion.

Motion 3: Arm Raise Elbow Bend. When performing the arm raise elbow bend movement, the person starts from the initial position, next raises the full arm to shoulder height, bends the elbow inwards towards the body until the forearm touches the biceps, and then returns to the initial position. Figure 7 shows the acceleration waveforms obtained during the arm raise elbow bend motion.

Motion 4: Clockwise Windmill. When performing the clockwise windmill from the initial position, the person rotates the shoulder 360 degrees clockwise and then returns to initial position. Figure 8 shows the acceleration waveforms obtained during the clockwise windmill motion.

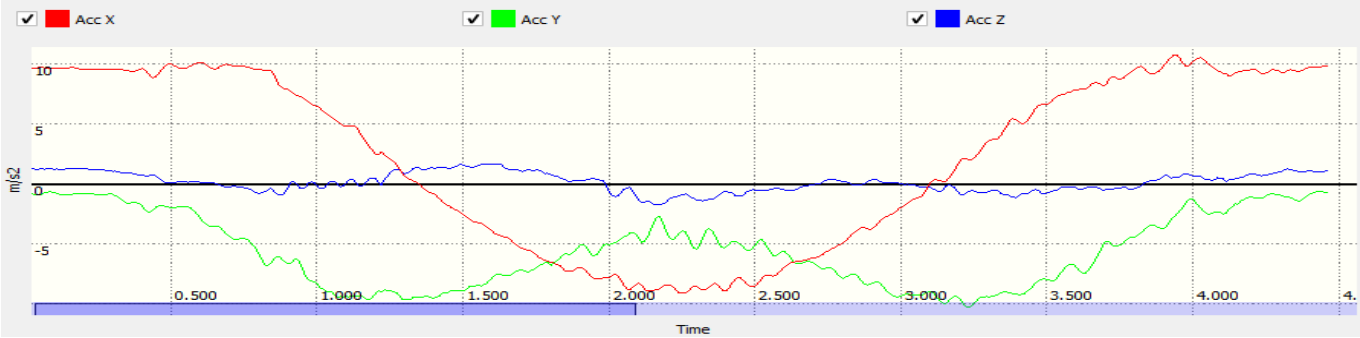


(a) Sensor 1

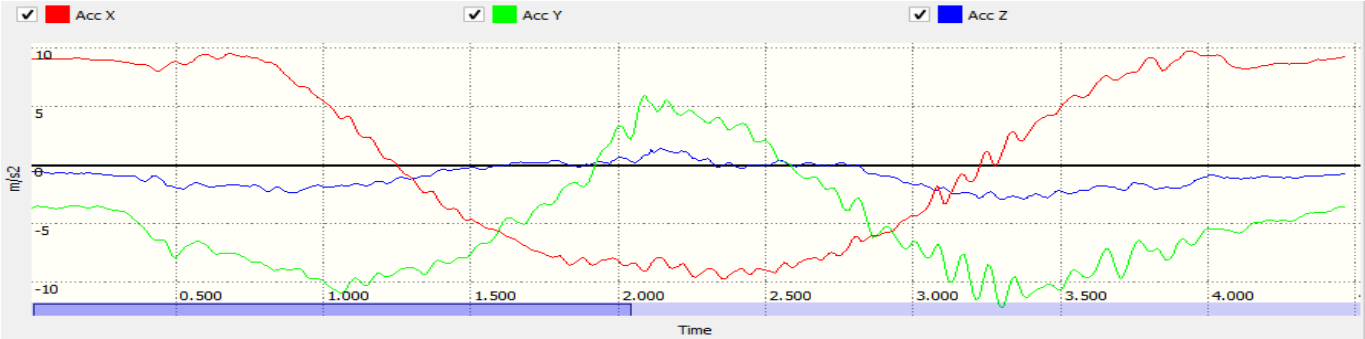


(b) Sensor 2

Figure 5. Acceleration Waveforms for Arm Extension

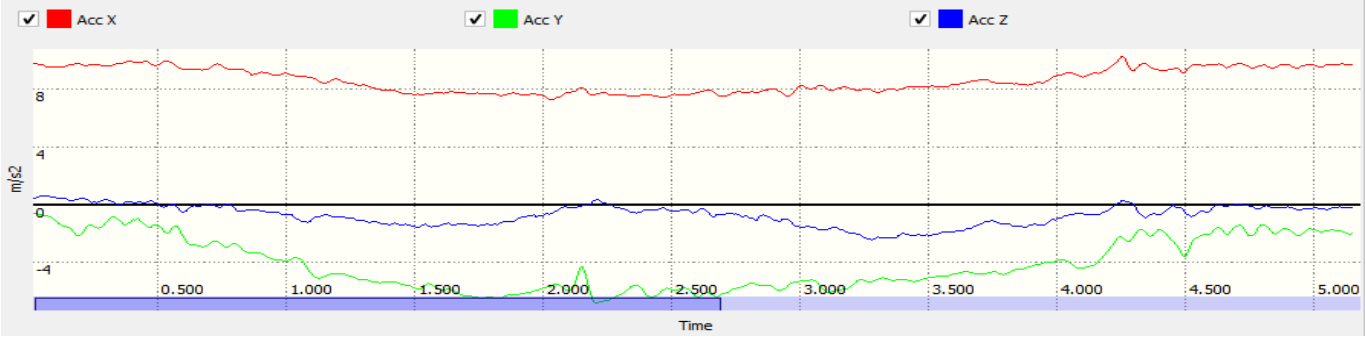


(a) Sensor 1

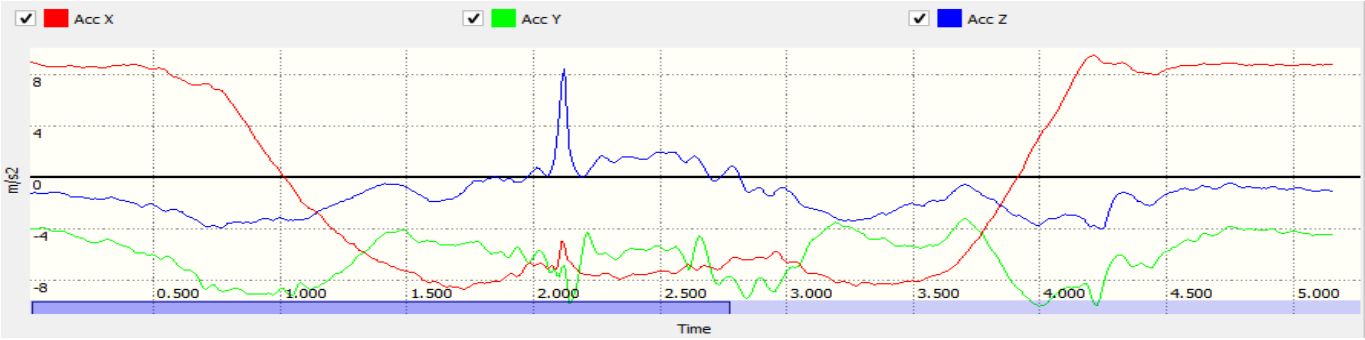


(b) Sensor 2

Figure 6. Acceleration Waveforms for Arm Raise

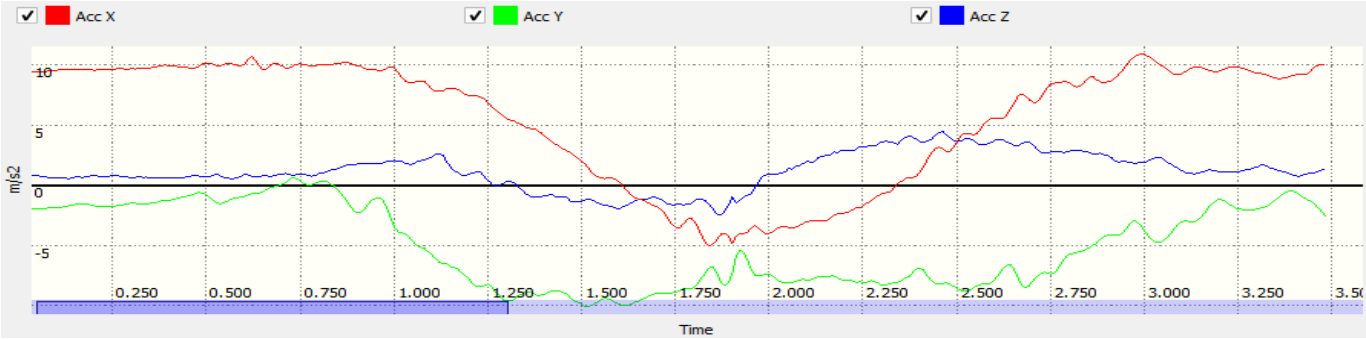


(a) Sensor 1

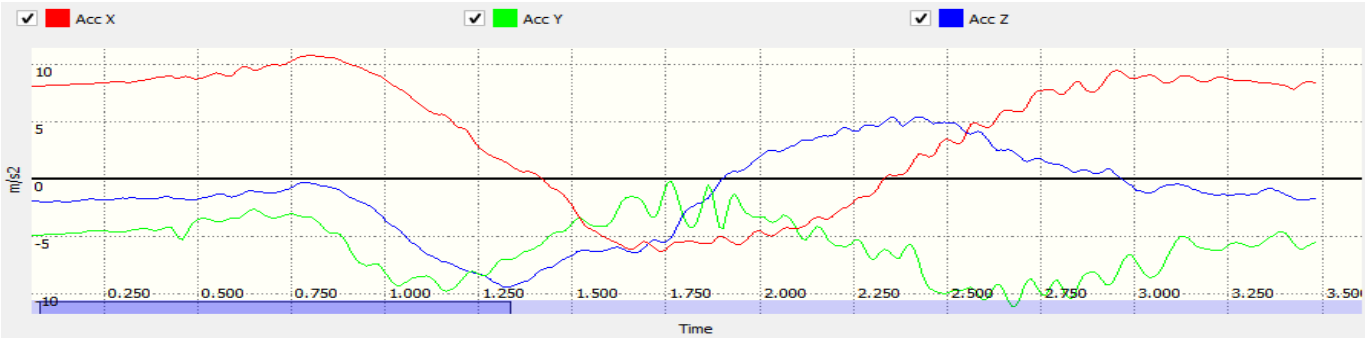


(b) Sensor 2

Figure 7. Acceleration Waveforms for Arm Raise Elbow Bend



(a) Sensor 1



(b) Sensor 2

Figure 8. Acceleration Waveforms for Clockwise Windmill

Motion 5: Counterclockwise Windmill. When performing the counterclockwise windmill from the initial position, the person rotates the shoulder 360 degrees counterclockwise and then returns to the initial position. Figure 9 shows the acceleration waveforms obtained during the counterclockwise windmill motion.

Motion 6: Shoulder Touch. When performing the shoulder touch movement from the initial position, the person bends the elbow until the hand touches the shoulder and then returns to the initial position. Figure 10 shows the acceleration waveforms obtained during the shoulder touch motion.

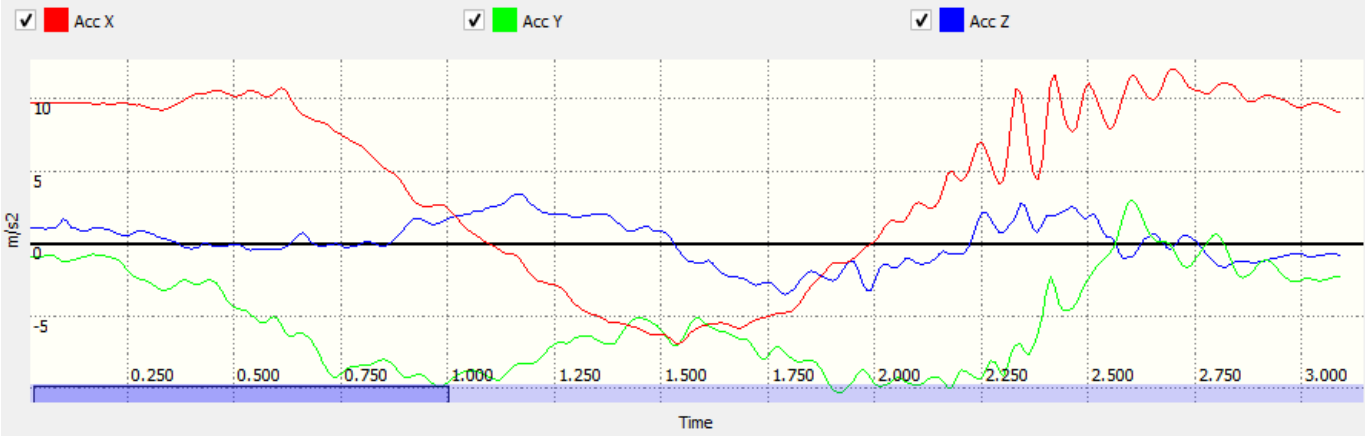
Motion 7: Side Arm Raise. When performing the side arm raise from the initial position, the person lifts the arm straight from the side 90 degrees until it is parallel to the floor, then returns to the initial position. Figure 11 shows the acceleration waveforms obtained during the side arm raise motion.

Motion 8: Wipe Right. When performing the wipe right movement from the initial position, the person bends the elbow until the arm is parallel to the floor, then rotates the forearm to the right as far as possible and then returns to the initial position. Figure 12 shows the acceleration waveforms obtained during the wipe right motion.

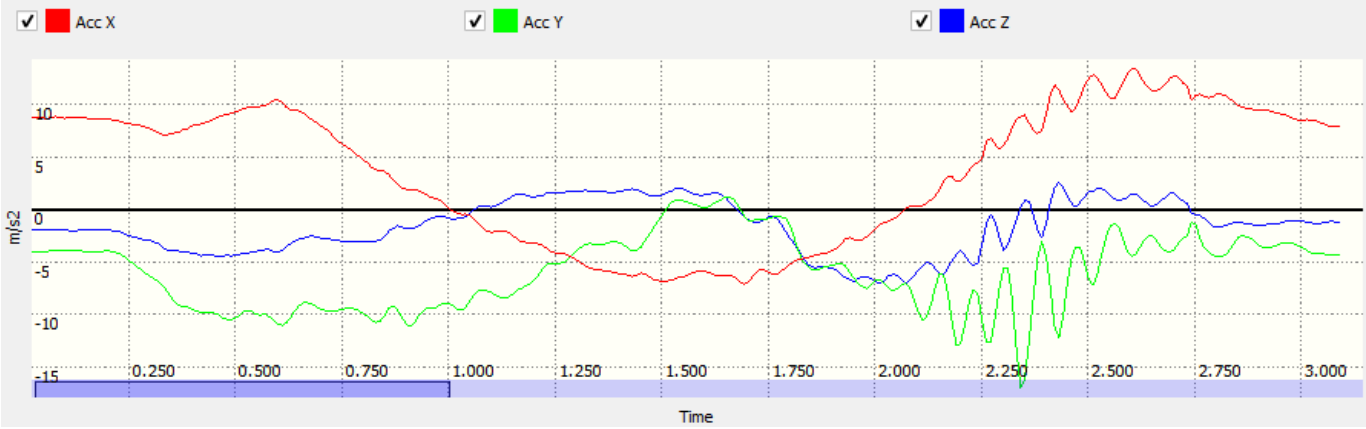
Motion 9: Wrist Rotation. When performing the wrist rotation from the initial position, the person bends the elbow until parallel to the floor, rotates the wrist inward and back again, and then returns to the initial position.

Artificial Neural Network

The robotic arm's controller had to be trained to distinguish among the arm's various movements. To make this possible, an ANN was used as a classifier to identify the specific arm movement to perform. The ANN was trained using the backpropagation algorithm and the triaxle acceler-

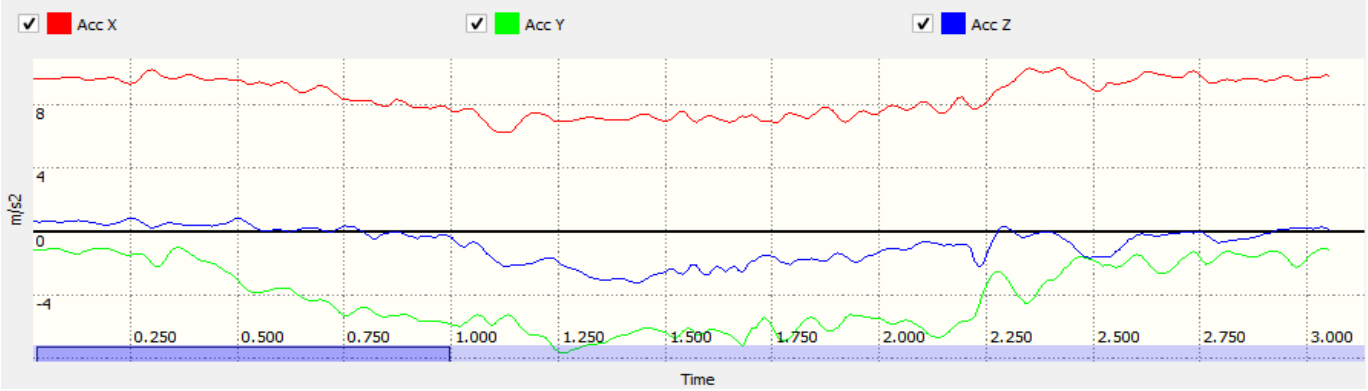


(a) Sensor 1

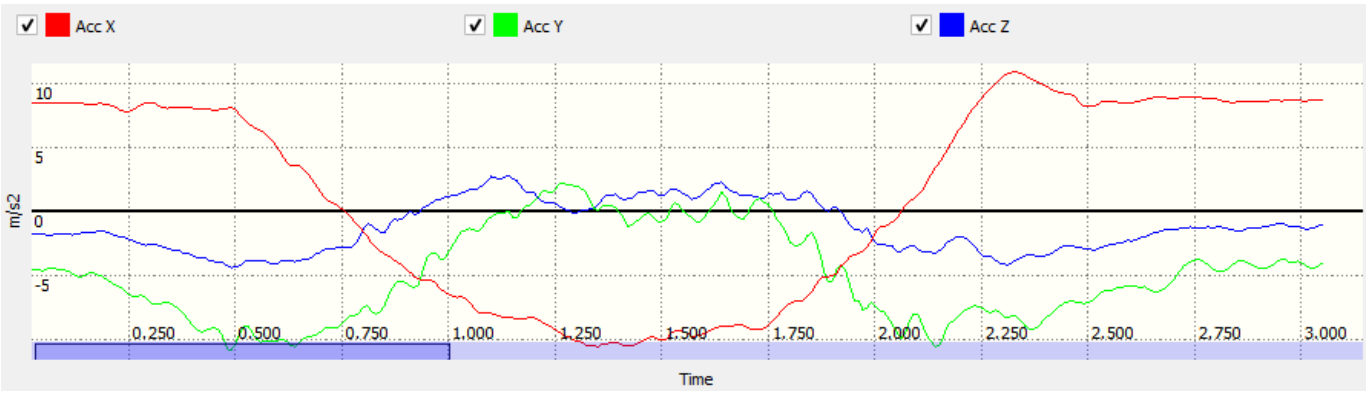


(b) Sensor 2

Figure 9. Acceleration Waveforms for Counterclockwise Windmill



(a) Sensor 1



(b) Sensor 2

Figure 10. Acceleration Waveforms for Shoulder Touch

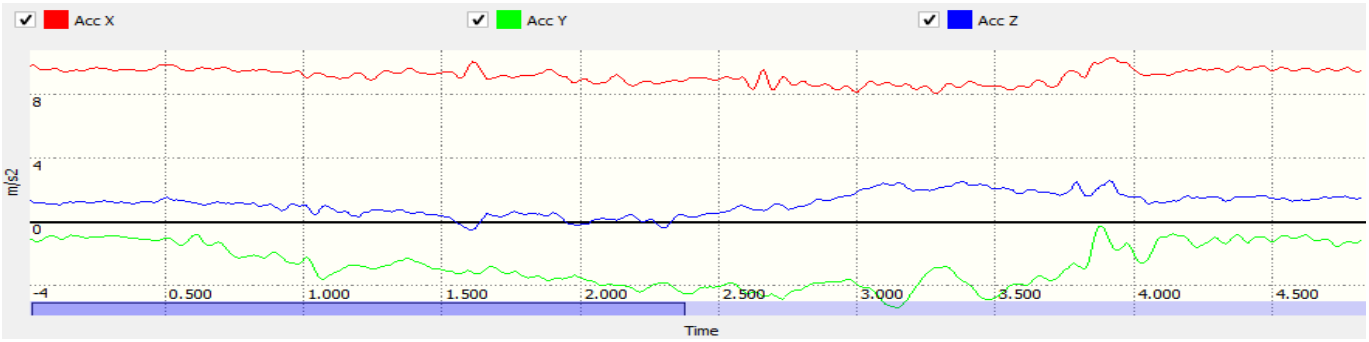
ation sampling data obtained from the IMUs. There was a total of 20 data sets for each motion. As mentioned previously, the IMU sampling rate was 100 Hz and each arm motion took approximately 3 to 5 seconds to complete. Each data-capture trial contained hundreds of individual data points. To reduce the size of the data sets, and be able to utilize the sets to train the ANN, the data sets were condensed into two statistical measurements for each sensor: the average rectified value (ARV) that can be calculated using Equation (1), and the root mean square (RMS) value, that can be calculated using Equation (2). The ARV is the average of the absolute values in the data set, whereas the RMS is the square root of the average. This would create a characteristic value that could be used as an input for the ANN.

$$x_{ARV} = \frac{1}{n}\left(|x_1| + |x_2| + \ldots + |x_n|\right) \quad (1)$$

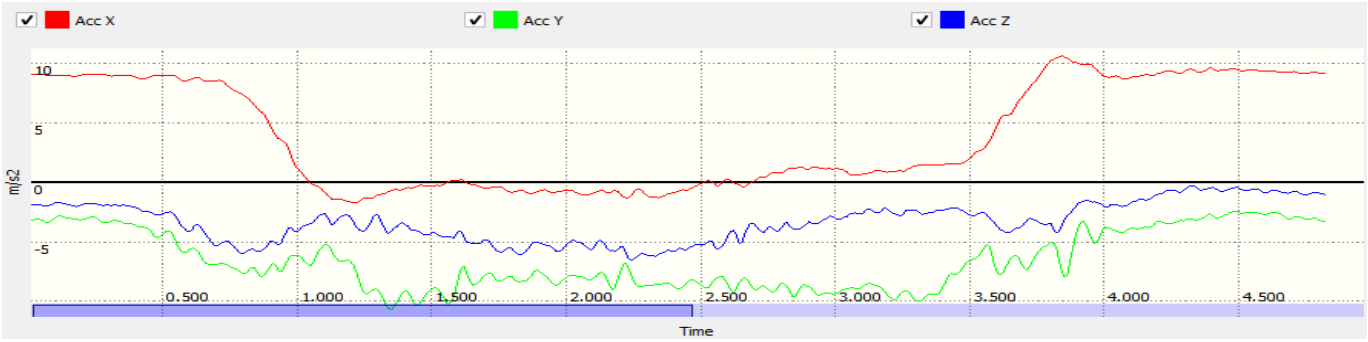
$$x_{RMS} = \sqrt{\frac{1}{n}\left(x_1^2 + x_2^2 + \ldots + x_n^2\right)} \quad (2)$$

The size of the final training matrix was 12x265. This corresponds to having 12 input nodes that are signals coming from the X-Acc, Y-Acc, Z-Acc coordinates, from sensor 1 and sensor 2, each having the computed ARV and RMS values (3x2x2). The length of each vector was 265 samples; this was selected by looking at the 3-5 second signals and selecting enough samples to have reliable information from the IMU. A target set matrix (output) with 9 rows by 265 columns was constructed to train the ANN with the correct output values that should be learned. Each of the 9 rows corresponded to a particular motion, while the length of the matrix mirrored the training set. Each movement's row had a 1 in its output cell, if the current trail matched, otherwise it had a 0.

The data set was divided into three batches: training, validation, and testing. During the training stage, the network ran through epochs, or iterations, and attempted to minimize the error until it could not progress further. The training set used 70% of the total database, 20% for the validation process, and 10% for the testing stage.

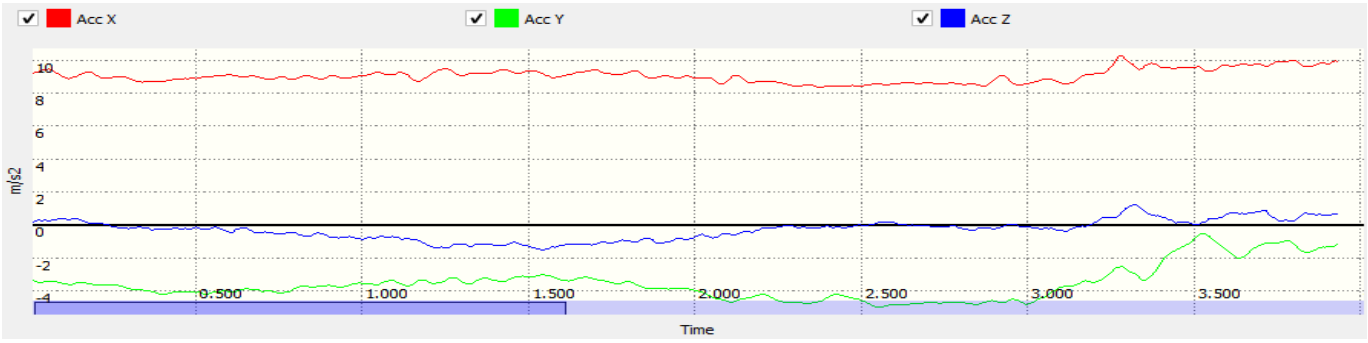


(a) Sensor 1

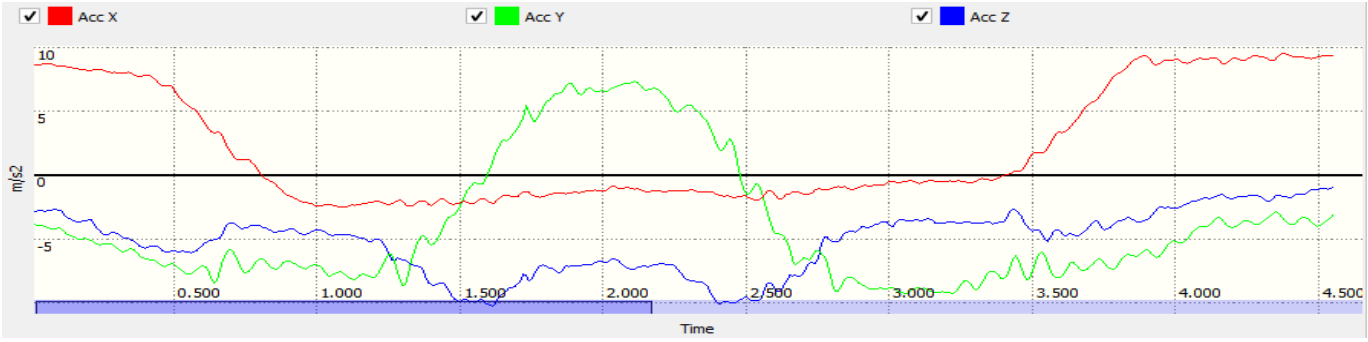


(b) Sensor 2

Figure 11. Acceleration Waveforms for Side Arm Raise



(a) Sensor 1



(b) Sensor 2

Figure 12. Acceleration Waveforms for Wipe Right

The validation stage was used to test the network progress, and signal when to stop the training, while the testing stage was used to measure the accuracy of the trained network. The database for the construction of the ANN consisted of 180 arm movements collected from four different subjects. Each subject performed each motion five times. The network architecture was constructed by selecting the amount of hidden neurons to place in the hidden layer of the network. This number can be tuned, but the general rule of thumb that was followed was to select a number somewhere between the number of inputs and outputs. For this study, there were 12 inputs and 9 outputs, so 10 neurons were placed in the hidden layer. The ANN was trained until the mean squared error of the output vector was minimized.

Robotic Arm

Figure 2 shows that the robotic arm used in this study had six degrees of freedom and used seven servo motors. Each servo had an individual signal port, with the exception of the two shoulder joint servos, which were driven in tandem. The six servos controlled base rotation, shoulder rotation, elbow rotation, writs rotation, wrist pitch, and grabber. Initially, a separate program was developed to model the desired movements using user-input angle writes. This allowed the authors to see how the arm would need to move in order to execute the proper movement. The robotic arm was programmed to receive input voltages (PWM signal) to each servo, corresponding to the particular motion classified by the ANN system. Then the robotic arm executed the movements in a predetermined sequence when the commanded motion had been completed: the robotic arm paused for about 5 seconds and then returned to the starting position.

The movements were set up so that the motors were stepped through a range of angles until reaching the desired position. This was accomplished using "for loops" and time delays to make the movements smooth rather than abrupt. The implementation of the ANN was performed using the ANN toolbox from Matlab. Once the ANN identified the specific arm movement, it would automatically transition to a program in which each servo motor was assigned a corresponding control signal (PWM). This was possible by interfacing the computer running the Matlab software with an Arduino microcontroller.

Testing the Arm Extension Motion

The robotic arm was capable of mimicking the motions of a human arm in real time. As an example, consider the arm extension motion, which entails bending the elbow until the arm is parallel to the floor, then extending the arm all the way forward. Figure 13 shows the initial position, then the subject begins to bend his/her elbow until the arm is parallel to the floor, as shown in Figure 14.

Figure 13. Subject in Initial Position

Figure 14. User Bends Elbow until Parallel to the Floor

Figure 15 shows that, after the person's elbow is parallel with the floor, he/she then begins to slowly extend the arm until it is fully extended, as shown in Figure 16. While the person performs these movements, the XYZ acceleration data are acquired and imported into Matlab, which computes the RMS and ARV values for each coordinate. This information is then fed into the ANN. The network outputs the corresponding commands that the robotic arm should do to perform the arm extension movements.

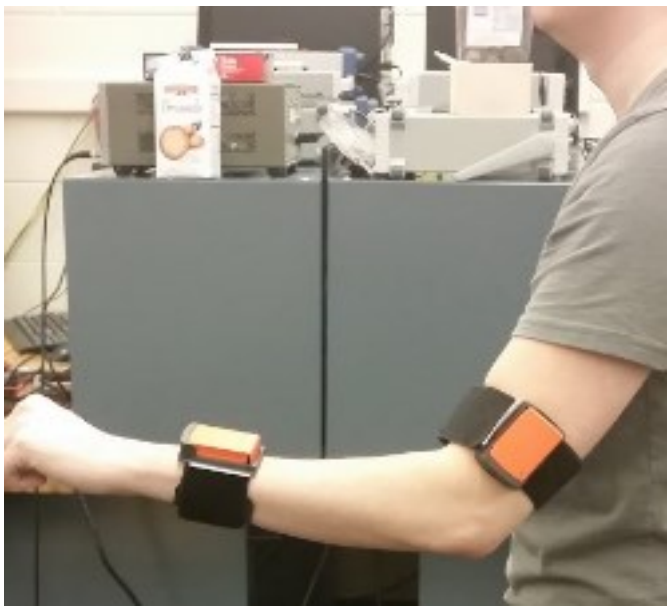


Figure 15. User Slowly Begins to Extend Arm

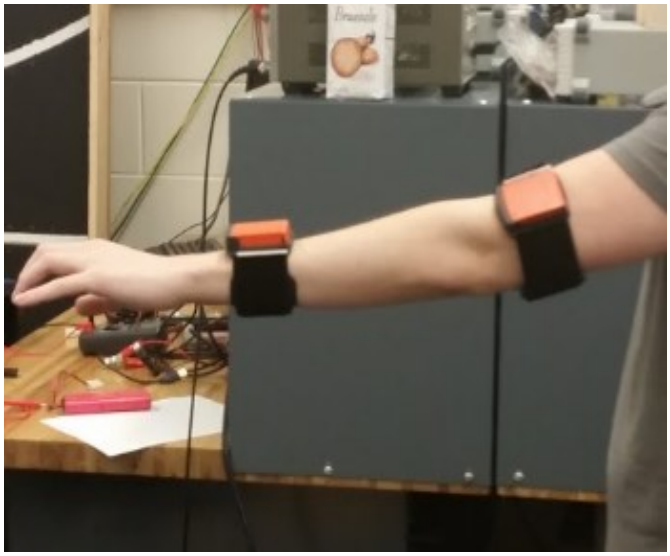


Figure 16. User Extends Arm

When the robotic arm is powered up, the arm assumes the initial position shown in Figure 17, and holds it until further input is received. Figures 18-20 show the corresponding arm extension motions performed by the robotic arm, mimicking the human arm movement described above. In particular for the arm extension motion, Figure 17 shows the robotic arm at the initial position that corresponds to a position of 10 degrees on the shoulder's servo motor, and 180 degrees angle on the elbow's servo motor. Then, the robotic arm begins to bend the elbow's joint until parallel to the floor (90 degrees on the elbow's servo motor), as shown in Figure 18.

Figure 17. Robotic Arm in Initial Position

Figure 18. Robotic Arm Bends Elbow until Parallel to the Floor

Figure 19 shows that, after the robotic arm is parallel to the floor, it begins to extend itself by moving the shoulder's servo approximately 55 degrees. Figure 20 shows that, as the arm extends the angles for the shoulder and elbow joints, they have to be adjusted so that the arm maintains a horizontal position with reference to the floor. Once the robotic arm has performed the complete set of movements for the arm extension motion, the robotic arm returns to the initial position and waits for the next command.

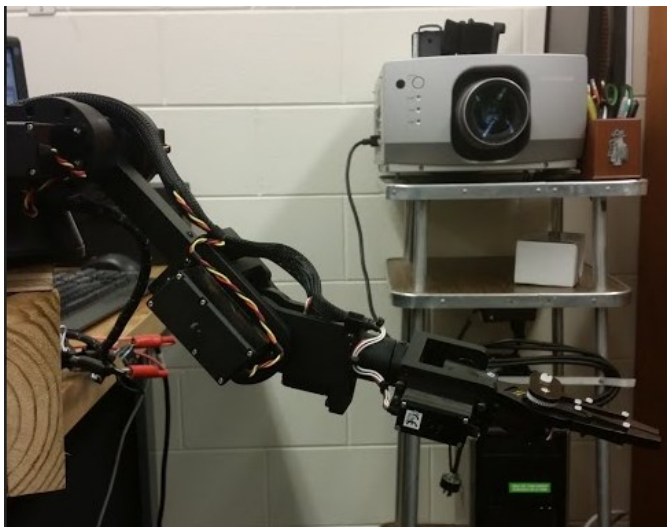


Figure 19. Robotic Arm Slowly Begins to Extend Its Arm

Figure 20. Robotic Arm Fully Extended

Results

To determine the accuracy of the ANN classification system, an independent volunteer was asked to perform each of the nine motions. The accuracy of the classification was 88.8% (one motion was misclassified by the ANN). Due to the mechanical limitations of the robotic arm used in this study, not all of the nine motions were able to be performed exactly as originally intended. The main reason for this was that the robotic arm was designed to be mounted and operate in a horizontal position, as shown in Figure 20. Thus, in order to better mimic human arm movements, it would be necessary to use a robotic arm that can be mounted and operate in a vertical position. Due to this limitation, there were three human arm movements that the robotic arm was not able to perform exactly, namely: the wipe right, the counter-clockwise windmill, and the clockwise windmill.

Conclusions

The overall system was able to perform the commanded movements in real time, with a small delay of about three seconds, due to the signal processing time required on the computer. This delay can be reduced by interfacing the Xsens Technologies' software directly with Matlab, so that the intermediate step of importing the signals captured by the Xsens technology into an Excel spreadsheet is removed. The robotic arm mimicking system was successful, but currently unilateral. The ANN performed very well. In the test with the independent subject, the ANN was able to correctly identify eight of the nine motions (88.8%). This accuracy can be improved by expanding the database that was used to train, validate, and test the ANN. Only motions from four subjects were used to train, validate, and test the ANN, and motions from a fifth subject were used as independent motions to compute the accuracy of the system. Currently the system is unilateral; that is, the human subject is the one that sends signals to the robot. Adding haptic feedback, however, the robotic arm would be able to send signals to the human subject, making a bilateral system that could expand the possible applications.

Potential applications of robotic mimicking include the manufacturing and medical industries. Manufacturing companies can use such a system in a way that a person can teach a robotic structure specific actions to perform without having an expert programmer. Robotic limbs can be integrated with this system to help amputees or people with disabilities, so that this system can help these individuals in providing arm movements or improve physical therapy and the improvement of motor skills.

References

[1] Lobov, S., Mironov, V., Kastalskiy, I., & Kazantsev, V. (2015). Combined Use of Command-Proportional Control of External Robotic Devices Based on Electromyography Signals. *Medical Technologies in Medicine*, *7*(4), 30-37.

[2] Hickman, S., Mirzakhani, A., Pabon, J., & Alba-Flores, R. (2015). A Case Study on Tuning Artificial Neural Networks to Recognize Signal Patterns of Hand Motions. *Proceedings of the IEEE SouthEastCon,* (pp. 1-4). Fort Lauderdale, FL. DOI: 10.1109/SECON.2015.7132893

[3] Sekhar, R., Musalay, R., Krishnamurthy, Y., & Shreenivas, B. (2012). Inertial Sensor Based Wireless Control of a Robotic Arm. *IEEE International Conference on Emerging Signal Processing Applications*, (pp. 87-90). Las Vegas, NV. DOI: 10.1109/ESPA.2012.6152452

[4] Georgi, M., Amma, C., & Schultz, T. (2015). Recognizing Hand and Finger Gestures with IMU based Motion and EMG based Muscle Activity Sensing. *Proceedings of the International Conference on Bio-inspired Systems and Signal Processing*, (pp. 99-108). Lisbon, Portugal. DOI: 10.5220/0005276900990108

[5] Jung, P., Lim, G., Kim, S., & Kong, K. (2015). A Wearable Gesture Recognition Device for Detecting Muscular Activities Based on Air-Pressure Sensors. *IEEE Transactions on Industrial Informatics, 11*(2), 485-494.

[6] Malegam, M., & D'Silva, M. (2011). Mimicking Robotic Hand-Arm, *Annual IEEE India Conference,* (pp. 1-5). Hyderabad, India. *DOI*: 10.1109/ INDCON.2011.6139365

[7] Wireless Motion Tracker XSENS. (n.d.). Retrieved from https://www.xsens.com/products/mtw-development-kit

[8] Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*, Cambridge University Press.

[9] Stergiou, C., & Siganos, D. (1996). Neural Networks. Retrieved from http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

[10] Shiffman, D. (2017). The Nature of Code: Chapter 10. Neural Networks. Retrieved from http://natureofcode.com/book/chapter-10-neural-networks/

[11] Artificial Neural Networks Technology, University of Toronto, Department of Psychology. (n.d.). Retrieved from http://www.psych.utoronto.ca/users/reingold/courses/ai/cache/neural2.htm

[12] XBee Pro, 2.4GHz, XBee 802.15.4 RF modules from Digi. (n.d.). Retrieved from http://www.digi.com/pdf/ds_xbeemultipointmodules.pdf

[13] Arduino Mega Microcontroller. (n.d.). Retrieved from https://www.arduino.cc/en/Main/arduinoBoardMega

Biographies

FERNANDO RIOS is an associate professor in the Electrical Engineering Department at Georgia Southern University. His areas of research include robotics, neural networks, fuzzy logic, and embedded systems. He has over 20 years of experience in higher education. Dr. Rios may be reached at frios@georgiasouthern.edu

ROCIO ALBA-FLORES is an associate professor in the Electrical Engineering Department at Georgia Southern University. Her areas of research include control systems, biomedical applications, and robotics. Dr. Alba has over 20 years of experience in higher education. Dr. Alba may be reached at ralba@georgiasouthern.edu

IMANI AUGUSMA is a graduate student in the Master of Science in Applied Engineering Program at Georgia Southern University. Her areas of interest include intelligent vehicles, mechatronics, and engines. Ms. Augusma may be reached at ia00336@georgiasouthern.edu