

Final Project Report, 18551, Spring 2011

Super Hand

Group #3

Allison Kator [akator@andrew.cmu.edu](mailto:akator@andrew.cmu.edu)

Michelle Lin [mjlin@andrew.cmu.edu](mailto:mjlin@andrew.cmu.edu)

## Table of Contents

Introduction .....	3
Problem Statement .....	3
Background .....	3
Physiology .....	3
Previous Research.....	3
Previous 551 Work .....	4
Signal Flow.....	4
Data Acquisition .....	6
Experimental Set Up .....	6
Procedure .....	6
Algorithm .....	7
Preprocessing .....	7
Real-Time.....	7
Thresholding .....	7
Feature Extraction .....	7
Classification.....	8
kNN.....	8
Specific Cases .....	8
Code.....	9
DSK .....	9
Results & Discussion.....	10
Demo .....	10
Addressing Feedback.....	11
Potential Biometrics Work .....	12
Database/Data Collection.....	12
Results.....	12
Discussion.....	13
Division of Labor.....	13
Future Work.....	14
Acknowledgment.....	14
References .....	14
Appendix .....	15

## Introduction

### Problem Statement

Even though the number of people who need prosthetic arms has greatly increased, the arms that amputees are given have not really improved since World War I. These arms basically have a hook at the end and allow the user to do very few basic activities. Hands are capable to do many things and can move in so many ways that living without one restricts the amputee from going about his or her daily life. When somebody loses their hand, they are still able to send the impulses to their wrist, but the signals stop there since they have nowhere to go. As biomedical engineering majors, we both have an interest in using these biomedical signals to express desired actions through different directions. This will be achieved through extraction features and classification methods.

### Background

#### Physiology

Some of the most influential arm muscles used in movements consists of extensors and flexors. The different types of muscles coordinate combinations of reversed reactions in order to move accordingly. A set of muscles that need to contract will cause the other set of muscles to release. See appendix for the anatomy of the arm. All muscle cells communicate with each other through electrical impulses. When these electrical impulses are recorded, these signals are then called electromyograms (EMG).

Muscle	Function
Extensor digitorum	main extensor of the fore fingers
Flexor digitorum superficialis	main flexor of the fore fingers
Flexor digitorum profundus	moving finger tips
Abductor pollicis longus	extends and lifts the thumb up and away from the palm
Extensor indicis	Extends the index finger (for pointing)
Flexor Carpi Ulnaris	Flexes the wrist and bends it towards the midline of the body

Table 1: Table of the different arm muscles used during motion

From: Abrahams, Peter. "How The Body Works." *Amber Books LTD*, 2009 pg. 228-231.

### Previous Research

The National Taiwan University Robotics Laboratory worked on a DSP-based controller for a prosthetic hand. They classified between 8 different hand motions using a 30-400 Hz 6th order Butterworth bandpass filter with a 4th order 60 Hz Butterworth notch filter. This research group used three channels and placed their electrodes on specific locations of the arm (palmaris longus, extensor digitorum and flexor carpiulnaris).<sup>1</sup>

---

<sup>1</sup> Huang, 2000.

The Arts Lab in Italy analyzed many different features and classification methods in both research and clinical contexts as the purpose of this article was to review “the state of the art of EMG-based control of artificial hands and attempt to define the potentialities and limits of this approach.”<sup>2</sup>

Lastly a lab group in Iran investigated 19 different features and tested the effectiveness of those features. While different features provided different important information, this paper made this assessment based on three different criterions of classification accuracy, noise tolerance, and calculation complexity.<sup>3</sup>

All of these papers used the Waveform Length, Variance, and Willison (or Wilson) Amplitude as their features when processing these EMG signals. As a result, we decided to use these features as well since they gave promising results in these papers.

### Previous 551 Work

There are two groups that have done related-hand detection-type projects. One group was Group 3 from Spring 2003, called “Handtranslation.” This project looked at different webcam images and attempted to classify the appropriate alphabet letters within those images. The other group was Group 4 from Fall 2008, called “Handtroller.” This group also based their project off of webcam images, aiming toward gesture recognition in order to play a PC game. Our project is novel because instead of using images as the data, our data will be EMG signals. There has been no other groups who has used EMG signals let alone apply them to any type of classification.

### Signal Flow

We used surface EMG electrodes to collect the data from the arm. The EMG electrodes were connected to a device called the BioRadio150. The BioRadio150 wirelessly sent the signals to the PC, where the software BioRadio150 Capture Lite saved the data in a specific directory. The data then is accessible to Matlab, where the signal is prepared to be sent to the DSK via CodeComposer and Microsoft Visual C++. CodeComposer contains the code for the DSK side while Microsoft Visual C++ contains the code for the PC side. When the DSK completes extracting features from the data, it sends the features to the PC, where Matlab will have access to them. Matlab will then classify the features against the training set, displaying the result on a Matlab GUI.

The electrodes that were used were MVAP-II Electrodes with Hydro Gel. We selected to use these electrodes in conjunction with the BioRadio150 because collecting EMG signals is most seamless and cost effective this way. Through Carnegie Mellon University, Biomedical Engineering Department, the BioRadio150, accompanying software and electrodes were provided to make this capstone project possible.

We decided to use the DSK to perform the feature extraction because of the limits of the chip. Because the training consists of a lot of data, the classification process was left to be done on the PC. The DSK would expect an input of 960x7 and output a 21x7, which isn't much data to

---

<sup>2</sup> Zecca, 2002.

<sup>3</sup> Boostani and Moraeli, 2003.

transfer. Because the PC can handle greater amounts of data (larger memory) and perform the desired functions at a decent pace (greater processing power), everything else was processed on the PC.

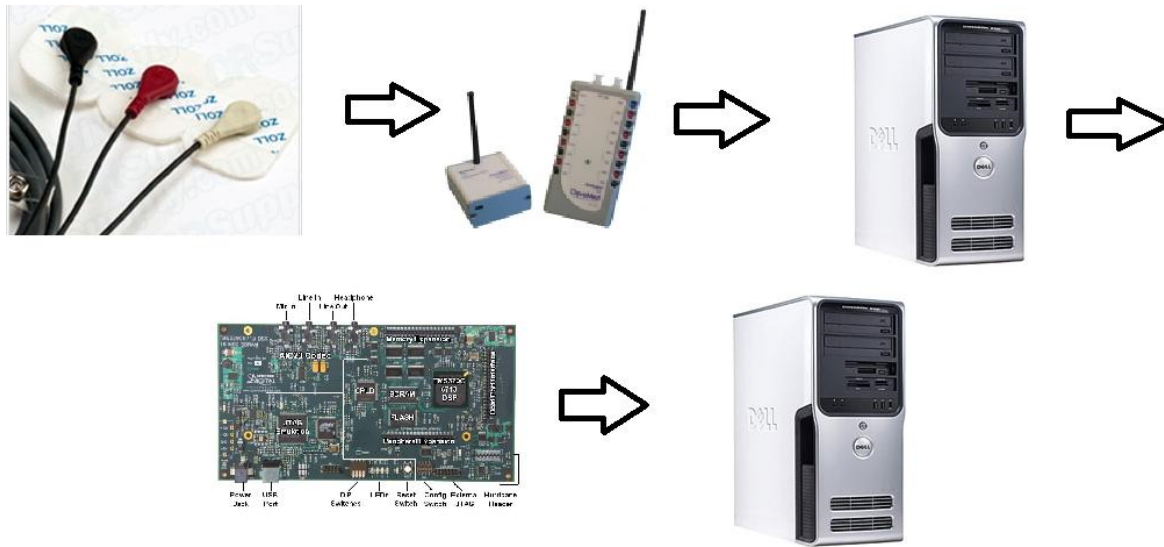


Figure 1: High Level flow graph of the hardware involved in this project

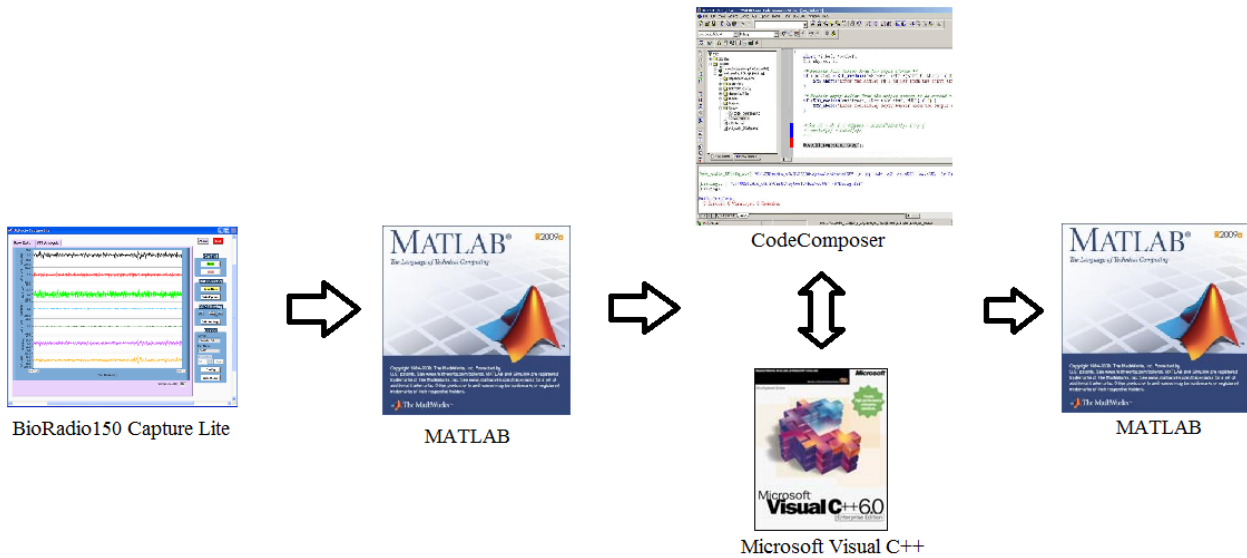


Figure 2: High Level flow graph of the software involved in this project

## Data Acquisition

### Experimental Set Up

To prepare data collection, the subject needs to prepare their skin for the surface EMG electrodes. The subject had to apply NuPrep, a gel that helps exfoliate the skin and remove all dead skin cells. This process, called pumicing, is to increase the clarity of the signals from the body to the surface EMG electrodes. The remainder is then wiped around with alcohol prep. To collect a holistic picture of the muscle signals in the arm, electrodes were placed all over the subject's arm in a circular rotation (see Figure 3).

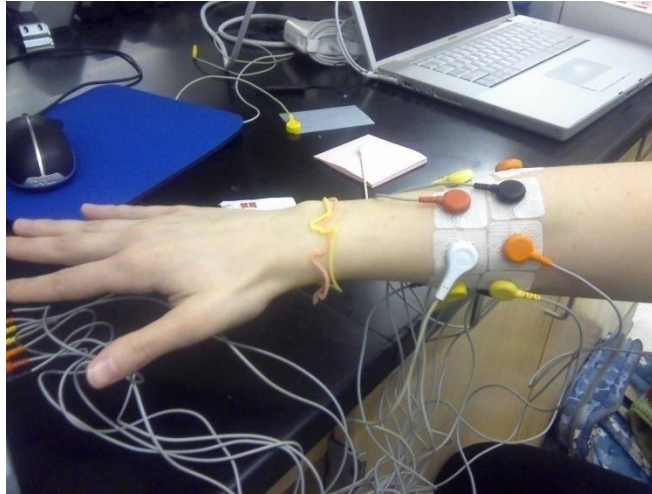


Figure 3: The arrangement of the electrodes wraps around the subject's arm.

On the software end, the BioRadio needs to be configured first. A channel has two electrodes (one for positive and the other for negative) where the differential is calculated. This gives a total of 14 electrodes, making seven channels. Two filters were applied: 6<sup>th</sup> order 30 Hz high pass filter and a 60 Hz notch filter. The high pass filter was applied to decrease the noise from motion artifacts. The notch filter was to get rid of the noise from the electrical outlets. The fastest sampling rate for the data to be collected was 960 Hz. The default setting for resolution is eight bits.

### Procedure

When Capture Lite begins saving the data, the subject performs a direction and holds it for five seconds. There were three directions that were desired: superior/inferior, abduct/adduct, and medial/lateral. Within each direction, there are different states that the hand can be in. For superior/inferior, there are two states: "neutral" and "apart." For abduct/adduct, there are five states: "up high", "up mid", "neutral", "down mid", "down high." Lastly for medial lateral, there are three states: "left", "neutral" and "right." These directions and their states are depicted in Table 2.




medial/lateral ("apart")	
superior/inferior ("wave")	
abduct/adduct ("abad")	

Table 2: All the arm states for each direction.

## Algorithm

The algorithm from the beginning of the semester till now has been modified many times. After much calculations and trial and error, the algorithm that will be detailed out will be the final algorithm that was used during the demo.

## Preprocessing

Because the BioRadio150 is set up to collect data based on the filters set, there were no other extra pre-processing stages other than squaring the data. Squaring of the data helps polarize any leftover noise from the signal desired.

## Real-Time

To simulate real-time, we implemented "changing windows." These windows were of a 1 second time frame (960 samples) and each new window would be incremented by 0.05 seconds (48 samples) later. Therefore all feature extractions and classification happens for every 0.05 seconds. Further discussion can be seen in "Addressing Feedback."

## Thresholding

With just using kNN, the algorithm had difficulty detecting any neutrals. Consequently a threshold classification was added. The mean of all data seen thus far was stored and compared to the mean of just the previous 960 samples. If the mean of the previous second was less than one half of the mean of all data seen, the thresholding part of the algorithm would classify the previous second as either neutral or apart depending on the results of the kNN algorithm, which will be discussed in "Specific Cases."

## Feature Extraction

After testing many different features, the feature that was selected was Waveform Length. Waveform Length is the sum of the differences between two consecutive samples for a tenth of a second (96 samples). This can be realized in this formula<sup>4</sup>:

$$WFL = \sum_{k=1}^N |\Delta x_k|$$

For thresholding, the feature used was average. Consequently, the features extracted from either MATLAB or the DSK were Waveform Length and average.

## Classification

### kNN

K-Nearest Neighbor is a very simple and rudimentary means of classifying signals where its class is unknown against signals where the class is known. kNN is possible through two different sets: training and testing set. The training set is put together to function as a standard. Any signal in testing set is then compared to the training set and determined what would be most similar. How the training set is built is described in more details in “Addressing Feedback.”

To determine how similar a training set is, the feature points are treated as a point of N-dimension (where N is the length of the features). The shorter the distances between the training point and testing point, the greater likelihood the class of the testing point matches the class of that training point. There are many different ways of calculating the distances but the one that worked best statistically was the Manhattan distance<sup>5</sup>:

$$d = |x - a| + |y - b|$$

Manhattan distance seems to perform the best probably because it does well in taking out outliers. With other distances like Euclidean, the shortest distance calculated can be offset by an outlier.

### Specific Cases

As was discussed before, a combination of thresholding and kNN were used to classify the signals. This worked fairly well, but seemed to classify apart as either up or left because the waves look similar, apart just has a smaller amplitude. Consequently, a specific case was added to classify apart. As was previously discussed, if the average of the last second was less than the average of all the data over two, the threshold classifier would suggest neutral. At this point, kNN was used and if it were to classify the previous second as up high or left, it was possible that the signal should actually be classified as apart. Because the only difference between the signals was a smaller amplitude for apart, another threshold was used. Both up high and left had high values in Channel 1 and Channel 2, while apart did not, so if Channel 1 was less than 100 microvolts and Channel 2 was less than 200 microvolts but kNN suggested up high or left, it would instead classify as apart. This works because up high and left are such high amplitude signals that if the kNN gives either of these classes but the threshold classifier says neutral, the signal cannot be in either up high or left, so it must either be neutral for all three classes or

---

<sup>4</sup> Huang, 2000.

<sup>5</sup> Ritz, 2006.



neutral for the first two, and apart for the last. However, both up high and left are different enough from neutral that kNN does not mix them up, and so if it were below the threshold and classified as one of these classes, it was probably in apart.

## Code

In data acquisition, parsing and saving data was done both in Matlab (csvwrite, testscan, save) and in C (loadArray, printArray). Feature extractions (average, variance, Wilsons Amplitude, Waveform Length) were first implemented in Matlab for testing purposes but then eventually translated into C for execution on the DSK (getFeatures and getFeature2).

Code for both PC-side and DSK-side transfer of data were used and modified from the 18-551 Lab 3 that was done in the beginning of the school semester. Lab 3 dealt with paging inputs and outputs and EDMA while our project does not since the inputs and outputs are small in size. The code was modified to include the function mentioned in the previous paragraph.

Classification (thresholding and kNN) was written in Matlab. The output was also displayed in a Matlab GUI which was done through the GUI layout editor (GUIDE, GUI Design Editor).

There are three different codes that were written for this project. The code in the “matlab” folder was the code that was ran for the demo, the “dsk” folder is our algorithm with the DSK incorporated, and the “dsk modified” folder is further analysis on the DSK.

## DSK

The DSK was used for feature extraction. Every time a new 48 samples were read into matlab, a new text file was created that had the previous 960 samples, and the number of the current sample, repeated 7 times, as the last row. Consequently, this file had 961 rows and 7 columns. This file was then read by the PC side of the C code. On the PC side, a 960 by 7 array was created using the first 960 rows, and it was then sent to the DSK for feature extraction. The DSK received this array and calculated the Waveform Length and average of it before sending it back to the PC side. The Waveform length is a 20 by 7 array and the average is a 1 by 7 array. After it was received, it was written into a text file, with the first 21 rows being the features from the DSK and the last row being the number of the current sample that was originally passed to it. The number of the current sample was passed to make sure that the MATLAB code waited for the DSK to process the current data set before classifying it. This is because when it was originally compared to the MATLAB results, the DSK results were one classification behind because the classification was being done too soon after the data had been sent to the DSK and the DSK didn't have enough time to extract features before using them. Consequently, the features it was using were the ones from the previous data set. However, adding the number of the last sample and waiting for that number to come back accounted for the delay and after this was added, the MATLAB and DSK gave the same features. To speed up the DSK code, instead of sending the 960 by 7 arrays to the getFeatures functions, pointers to these arrays were sent and used. The main reason for the delay was reading and writing text files, but this was the best way to integrate the code with MATLAB.

## Results & Discussion

The training accuracy is consisted of all the different directions and how well the features are able to classify accurately.

Train	all directions
euclidean	80.6%
manhattan	81.5%
cosine	73.9%

Table 3: Training accuracies against different distances. Threshold of average/4. Contains 2360 windows.

In training accuracy, there were 2360 windows that were classified. The threshold that was used for these accuracies was the average of the input data divided by 4.

The testing accuracy is broken up into its specific directions. For the wave classification, the threshold was the average divided by 4 and for the other directions, the threshold was the average divided by 2.

Test	wave	abad	apart
euclidean	60.7%	79.5%	82.4%
manhattan	66.4%	77.8%	83.8%
cosine	46.3%	81.3%	77.4%

Table 4: Testing accuracies against different distances.

In the direction for wave there were 2640 windows, 1280 windows for abduct/adduct direction and lastly 642 windows for apart/neutral. It can be seen, as foreshadowed in the Algorithm section, that Manhattan performed better overall for both training accuracy and testing accuracies.

The error that has occurred is probably due to multiple factors. One factor would be a labeling issue. When labeling the accurate labels for the data, transitional states were not marked as “don’t care.” This is important because to go from State A to State B is expressed in the signal however when we label the states, it’s purely binary.

The superior/inferior direction has lowest accuracy in comparison to the other directions most likely because it is classifying more states. The greatest inaccuracy that occurs is classifying are the less extreme states (“up mid” and “down mid”).

## Demo

In the demo, Allison Kator placed electrodes all over her arm as described in “Data Acquisition.” When Allison was hooked up to the BioRadio150, data was collected through BioRadio150 Capture Lite software.

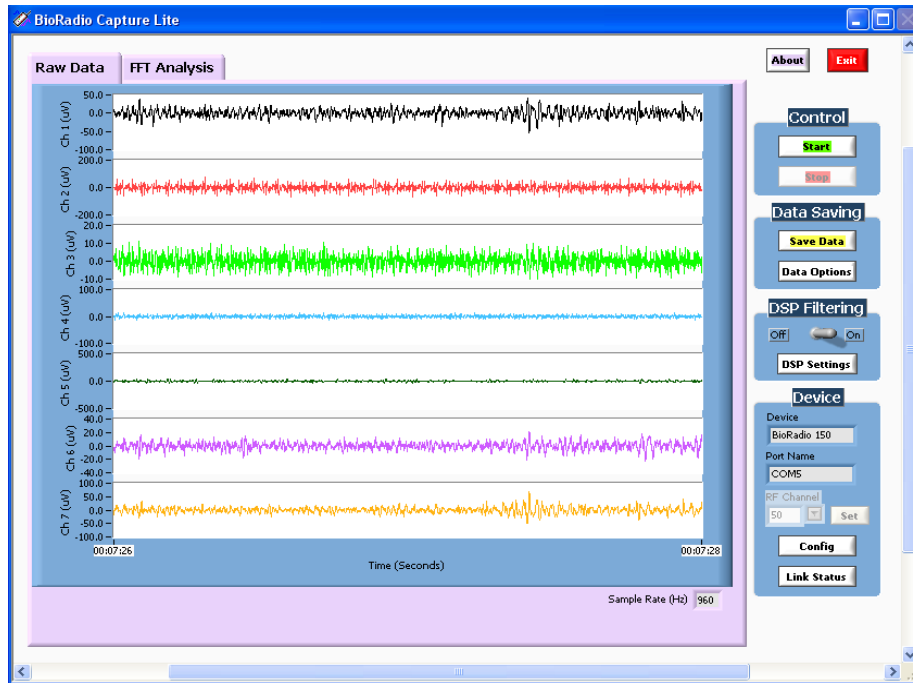


Figure 4: Screen shot of the CleveMeds software “BioRadio150 Capture Lite”

As the program is saving data, the matlab code is executed where feature extraction and classification happens. A GUI will appear with the results from the classification.

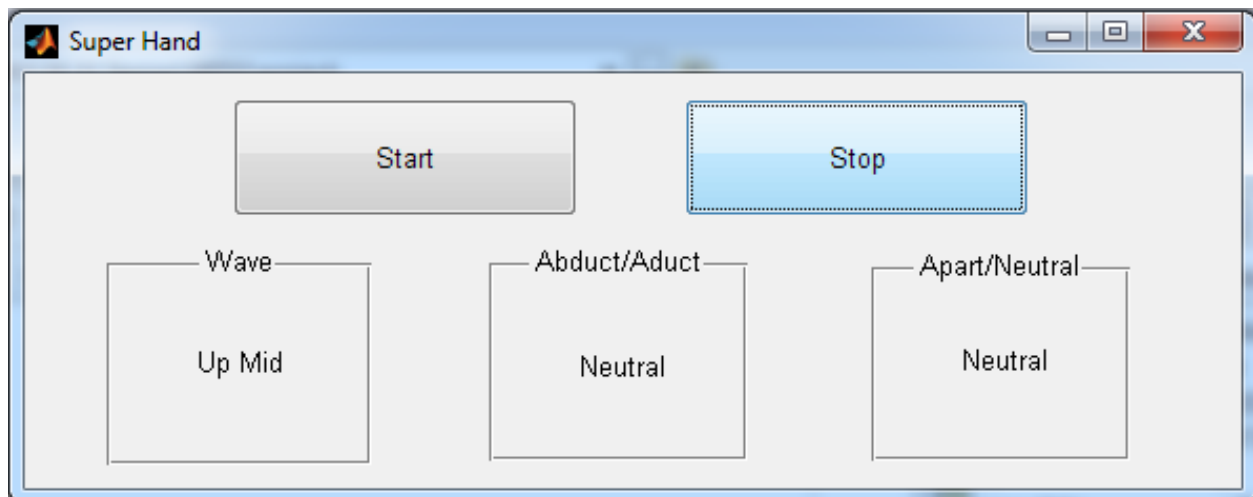


Figure 5: Screen shot of Matlab GUI

## Addressing Feedback

Initially all processing was done “offline.” It was offline in terms of taking an entire stream of data and performing calculations upon that data. However, with the purpose of the project being real-time, the way data was fed into the system and processed changed. Instead of viewing the entire data collected and processing everything that has happened, we implemented “changing windows.” These windows were of a 1 second time frame (960 samples). Once that window was processed, the next 1 second window would be data 0.05 seconds (48 samples) later. Therefore the data would overlap thus simulating a real-time system.

The training set first included the features of only the first second (out of five seconds) of each direction. However because the system switched to a real-time system, the classifier’s ability to identify a state in the middle of the five seconds (instead of only the first second) was extremely poor. As a result, the training set was redone to include features in different seconds of the active-motion, making the training set more robust and versatile.

Certain features were more effective than other features and therefore were suggested to change our classifier to weight the features based on importance. However after much testing, only certain features seemed to work best with the data. As a result, only one feature (Waveform Length) was used for the final demo, avoiding the issue all together. This helped increase the speed of the system as well as avoid issues such as paging for transfer of data to and from the DSK.

### Potential Biometrics Work

As inquired during presentations, a short analysis has been done to investigate the possibility of biometric work with EMG signals. Using the exact same training features from the main project, the question that we attempted to resolve is if another person’s EMG signals would be classified correctly.

### Database/Data Collection

The two subjects were Allison Kator and Michelle Lin. With the same set up as the project (see Data Acquisition), each person recorded these series of actions: neutral, up, neutral, down, neutral, left, neutral, right, neutral, apart, and neutral. The number of windows for Allison’s data amounted to 605 while Michelle’s data had 576 windows.

### Results

The original training features contained only Allison’s EMG features. There were three different statistics that were calculated. There is an overall accuracy rate – across all directions, specific direction accuracies for each subject and lastly a correlation confusion matrix. The confusion matrix is a 7x7 matrix, 7 for the total channels. Each channel of one subject was normalized and cross correlated with the other subject’s channel. The more correlated the signals are, the higher the correlation values (closes to 1 since it is normalized). Across the columns are Allison’s channels and across the rows are Michelle’s channels.

OVERALL	Allison	Michelle
euclidean	62.6%	50.5%
manhattan	64.5%	49.7%
cosine	60.3%	47.4%

Table #: Overall accuracy of Allison and Michelle’s testing data against original training set

Allison	Up High	Down High	Neutral	Left	Right	Apart
euclidean	12.2%	100.0%	90.5%	19.0%	0.0%	0.0%
manhattan	32.7%	100.0%	90.5%	20.7%	0.0%	0.0%
cosine	14.3%	69.4%	90.5%	19.0%	0.0%	0.0%

Table #: Broken down accuracies of Allison’s data tested against Allison’s training data.

Michelle	Up High	Down High	Neutral	Left	Right	Apart
euclidean	0.0%	53.8%	84.5%	14.0%	0.0%	0.0%
manhattan	0.0%	57.7%	84.5%	0.0%	0.0%	0.0%
cosine	12.5%	13.5%	84.5%	2.0%	0.0%	3.3%

Table #: Broken down accuracies of Michelle's data tested against Allison's training data

0.300	0.242	0.294	0.214	0.225	0.191	0.229
0.272	0.396	0.283	0.264	0.250	0.204	0.494
0.282	0.370	0.284	0.265	0.248	0.207	0.381
0.237	0.310	0.256	0.240	0.206	0.217	0.340
0.252	0.259	0.261	0.224	0.299	0.249	0.315
<b>0.414</b>	0.212	<b>0.416</b>	0.242	<b>0.360</b>	0.280	0.209
0.276	<b>0.470</b>	0.316	<b>0.343</b>	0.282	<b>0.289</b>	<b>0.513</b>

Table #: Confusion matrix of correlation values across channels.

## Discussion

When testing Allison's new test data against the original training data, the results yielded overall were better than when Michelle's test data was tested against the training set. This makes sense because a good factor to high accuracy with the classification deals with the placement of the electrodes. Fortunately, as stated in Data Acquisition, Allison's electrode placements were generally constant because of the physical biomarker she has on her arm (a freckle). Therefore there is greater consistency when testing with the data. When Michelle's data is tested against Allison's training data, there is greater discrepancy because the placement of electrode will not be exactly the same. When viewing the broken down accuracies of the directions, overall it can be seen that Michelle's testing data performed far worse than Allison's data.

In an ideal case, the best correlation for a channel should be with its own channel, therefore resulting in a diagonal – from top left to bottom right – of bolded numbers. When observing the confusion matrix, Michelle's channels 6 and 7 correlate most with all of Allison's channels (see Table #). This could be due to placement of the electrodes. However if it truly was a displacement of electrodes, the deviations would be shifted by a channel or two because the electrodes were placed in order and in the same direction as Allison's electrodes. Since the result is in no way a diagonal formation, there might be some component of the signal being unique to each subject.

## Division of Labor

Date	Tasks	Both	Primary	Secondary
2/13 - 2/20	Final selection of arm muscle location and arm movements		A	M
2/21 - 2/27	Data Collection	x		
2/28 - 3/4	Selecting best features for data	x		
2/28 - 3/4	Selecting best classifier for data	x		
3/14 - 3/20	Coding algorithms for feature extraction in MATLAB		A	M

3/14 - 3/20	Coding algorithms for classification in MATLAB		M	A
3/21 - 3/30	Midproject Oral Presentations	x		
3/30 - 4/2	Additional changes to feature extraction and classifier	x		
4/2 - 4/9	Final testing with new changes to features and classifiers	x		
4/10-4/17	Hooking up all hardware and software parts to lab computers	x		
4/18 - 4/23	Coding of feature extraction on DSK in C	x		
4/18 - 4/23	Writing the GUI to load results		M	A
4/23 - 4/25	Transferring data to and from DSK	x		
4/26	Lab Demo	x		
4/27 – 5/1	Further analysis on DSK and performance issues		A	M
4/27 – 5/1	New data collection and analysis for potential biometrics work		M	A

## Future Work

In the future, it would be nice to get more classes of movements. Originally it was a goal to classify finger movements, but there was not enough time. So in the future, it would be nice to do this. Additionally, it would be nice to see if sEMG data has a biometric component.

A lot of time was spent trying to make the Matlab software work with the BioRadio150 software. Originally, an SDK was used to have everything running in Matlab, but this worked a lot worse than just recording data in the BioRadio software and transferring it over. A lot of time was spent trying to figure out what filters worked the best.

## Acknowledgment

This project was possible because of the resources allocated through the Biomedical Engineering Department. We would like to thank Conrad M. Zapanta, Ph.D. And Yu-li Wang, Ph.D. for allowing us to use the hardware and software for the BioRadio, along with the Biomedical Engineering Laboratory to collect data.

## References

- Dr. Scott Day, “Important Factors in Surface EMG Measurement”, Bortec Biomedical Ltd, 225, 604-1st ST SW, Calgary, AB, T2P 1M7  
<http://www.bortec.ca/Images/pdf/EMG%20measurement%20and%20recording.pdf>
- De Lucca, G., “Fundamental Concepts in EMG Signal Acquisition”, Delsys Inc, 2003  
<http://www.udc.es/inef/profesores/Miguel%20del%20Olmo/Documentos/cursos%20doctorado%20valoracion/III.pdf>
- Han-Pang Huang & Chiang, Chun-Ying, "DSP-based controller for a multi-degree prosthetic hand," Robotics and Automation, 2000. *Proceedings. ICRA '00. IEEE*

*International Conference on*, vol.2, no., pp.1378-1383 vol.2, 2000

<http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=844790&isnumber=18246>>.

- Abrahams, Peter. "How The Body Works." *Amber Books LTD*, 2009 pg. 228-231.
- Zecca, M. *et al.* "Control of multifunctional prosthetic hands by processing the electromyographic signal." *Critical Review Biomedical Engineering*. 2002, vol.30, pg. 459–485
- R. Boostani and M. H. Moradi, "Evaluation of the forearm EMG signal features for the control of a prosthetic hand," *Physiological Measurement*, vol. 24, no. 2, pp. 309–319, May 2003.
- D. Nishikawa, W. Yu, H. Yokoi, and Y. Kakazu, "On-line learning method for EMG prosthetic hand control," *Electronics Communications in Japan*, vol. 84, no. 10, pt. 3, pp. 1510–1519, Nov. 2001.
- Ritz, Anna. "Generating Normalized Cluster Centers with KMedians," 2006. *SDM '06*. <http://www.cs.brown.edu/~aritz/files/SDMpresentation.pdf>>.

## Appendix

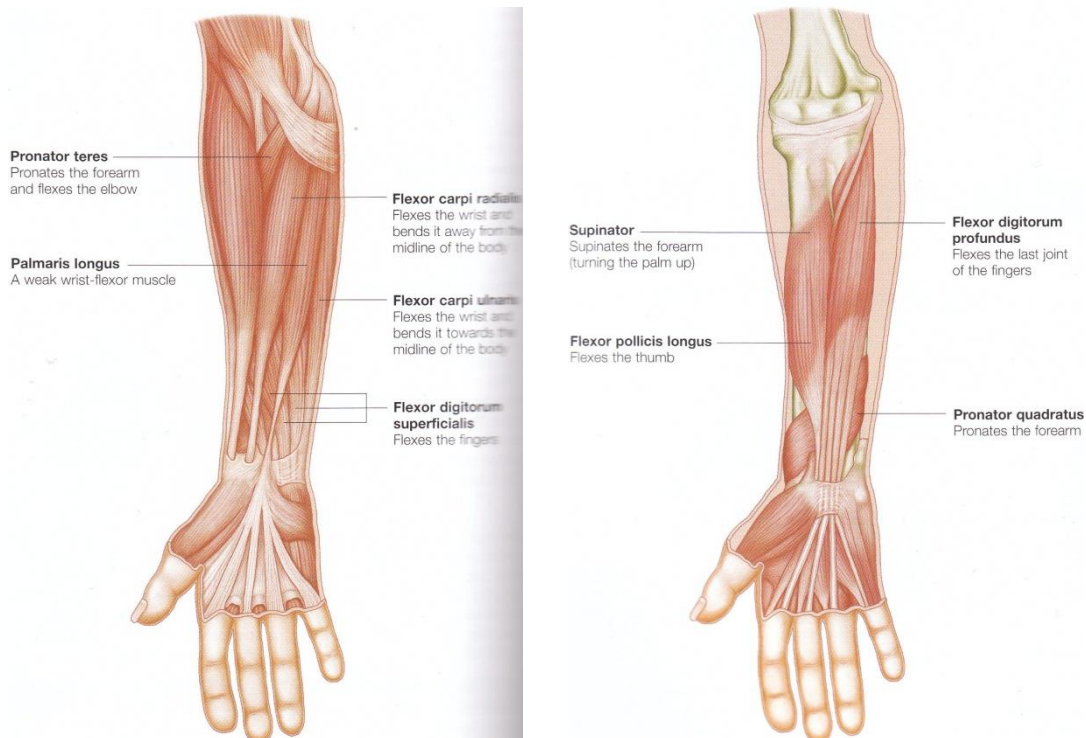


Figure 2: Anatomy of forearm muscles. Left: superficial flexor muscles Right: deep flexor muscles<sup>6</sup>

<sup>6</sup> Abrahams, 2009.



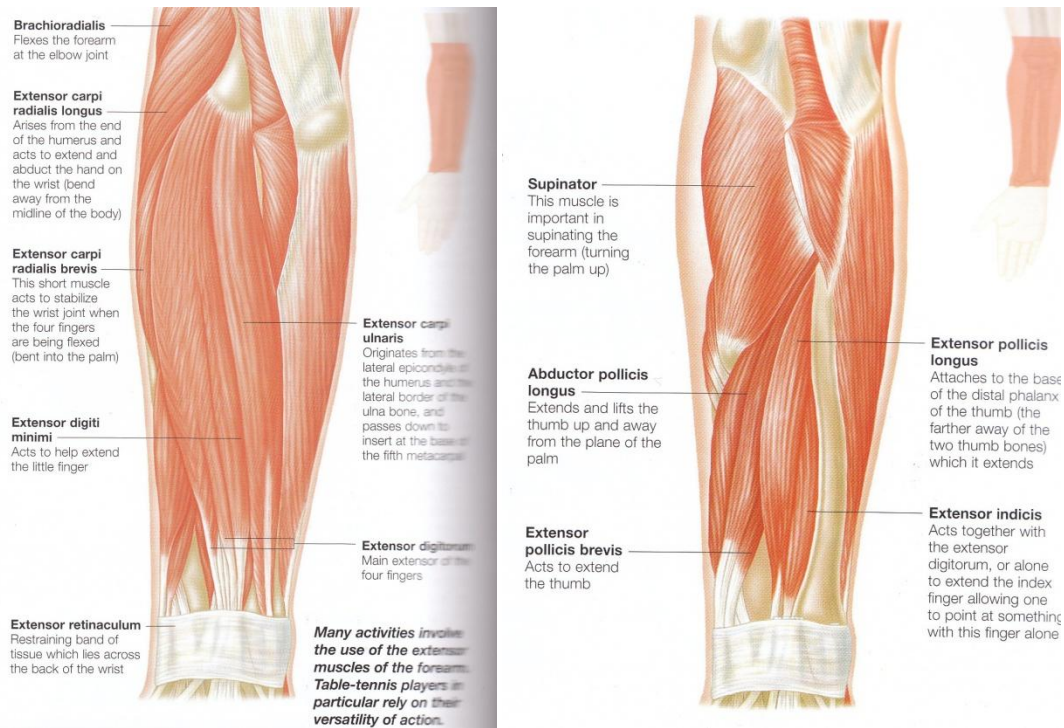


Figure 4: Anatomy of forearm muscles. Left: superficial extensor muscles Right: deep extensor muscles<sup>7</sup>

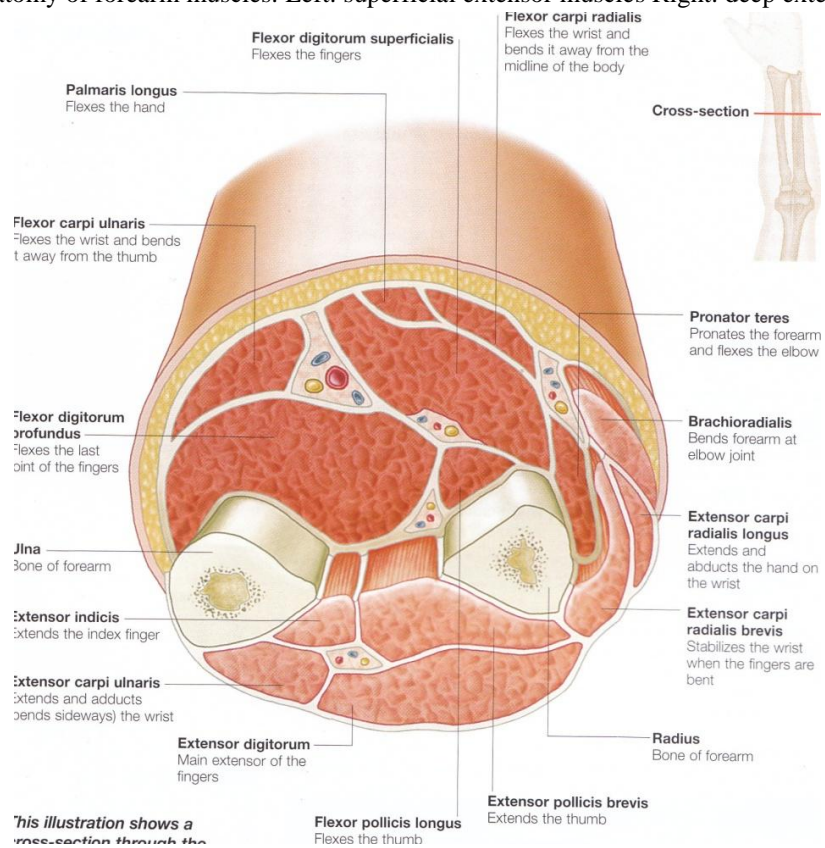


Figure 3: Cross section of the forearm<sup>8</sup>

<sup>7</sup> Abrahams, 2009.

<sup>8</sup> Abrahams, 2009.