Software Design Document

Assistive Limbs: Using MYO Armband

Amr Hamdi, Hassan Hamdy,Philip Naguib, Hossam Badr
Supervised by Dr. Sarah Nabil and Eng. Radwa Samy

February 14, 2018

1 Introduction

1.1 Purpose

This Software Design Document provides the design details of Assistive Limbs: Using MYO Armband, which detect EMG Signals from the patient's arm and send it to the unity to apply it on the virtual arm. this document contains the software Architecture , the design choices and the traceability matrix of the system.

1.2 Scope

This document contains the complete description for our system design which is important for experts and developers to understand our system architecture. This software system will control the assistive arm and help the amputees living ordinary life.
we use machine learning techniques to deal with our own dataset. Our system has an Creative design to get the patient movement to increase our dataset. Increasing our accuracy,the software has feedback system to allow the patient to contact us if there any problem with applied movements.

1.3 Overview

This is the overview of Software Design Document. the next section provides the system overview and how the system works. then the system Architecture section have Architectural Design,Sequence Diagram,Decomposition Description (Class Diagram) and Design Rational.
after that the Data Design section which have Data Description and Data Dictionary.

then the Component Design section which have the details of the feature extraction and classification Algorithms used in the System. The next section provides the Human Interface Design which have Screen Images of the System. Finally, the last section provides Requirements Matrix.

1.4 Definitions and Acronyms

Term	Definition
Software Design Document (SDD)	Used as the primary medium for communicating software design information.
MVC	Model view Controller
SVM	Support Vector Machines
MAV	Multi Algorithm Voting
MYO	Myocardium (Armband device which detect EMG signals)
EMG	Electromyography signals
CNN	Convolutional Neural Network
KNN	K-nearest neighbor
ANN	Artificial Neural Network
RMS	Root Mean Square
FNN	Feedforward Neural Network
ES	Evaluation Strategies
LPF	low-pass filter
LPD	Low pass Diffrential filter

2 System Overview

Real time moving and observing of virtual arm and enhancement of classification system. The MYO armband got 8 channels can detect signals through each channel for acceleration and orientation. After MYO Armband reading signals to be filtered from noise and preprocessed sends it to fire-base cloud to be stored. Firebase cloud send the readings to processing unit to detect the movement and be classified and send the signals generated to the cloud. The classification system should mainly aim to provide accurate understanding of intended movements to help amputees achieving their daily life tasks easily. Cloud sends detailed signals and action for arm to be rendered to in unity3d arm the system will get the model from the cloud and select what intended movement should be applied. If misclassification occurred the user sends feedback to the cloud. This document proposes MIU-X-Prosthetic which is a system that performs detection and identification of users EMG signals using MYO armband device. the user will get an account when he/she downloads the application account to allow him uploading his dataset files which are CSV files to make it customizable. the user will be able to send feedback if the movements where wrong through log files and he/she must also send his intent of the movement the system then should extract the needed features then add train the classifier with the intended movement, the user can add new movements by selecting which fingers he need to move and adding the node also. then an alarm will be sent to the admin to check if the movement can be added so he will add it in the next update, if there are new update includes new movements a notification will be sent to all users to make them update their application to the latest version.

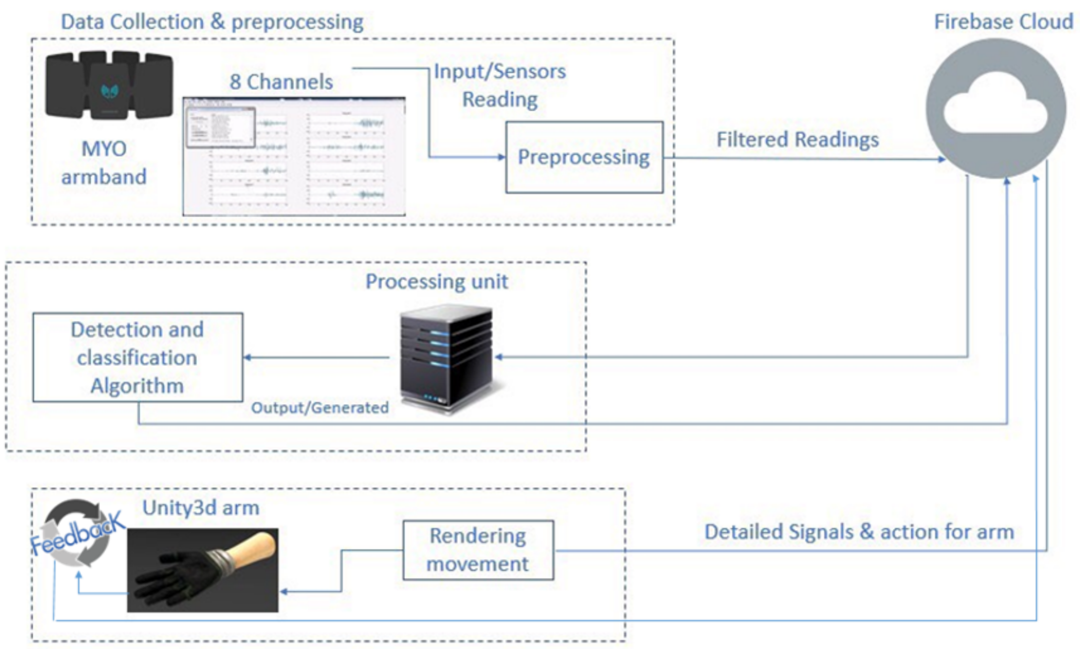


Figure 1: System Overview

3 System Architecture

3.1 Architectural Design

The system is designed in MVC architecture.
The Architectural provides three sections Model,View and Controller as shown in fiqure [2].

3.1.1 View

This represents the user interface which is divided into two interfaces as Follow: Web Interface where the User can interact with the System. the web App provides training the user dataset for the 10 Movement which we deal with them.

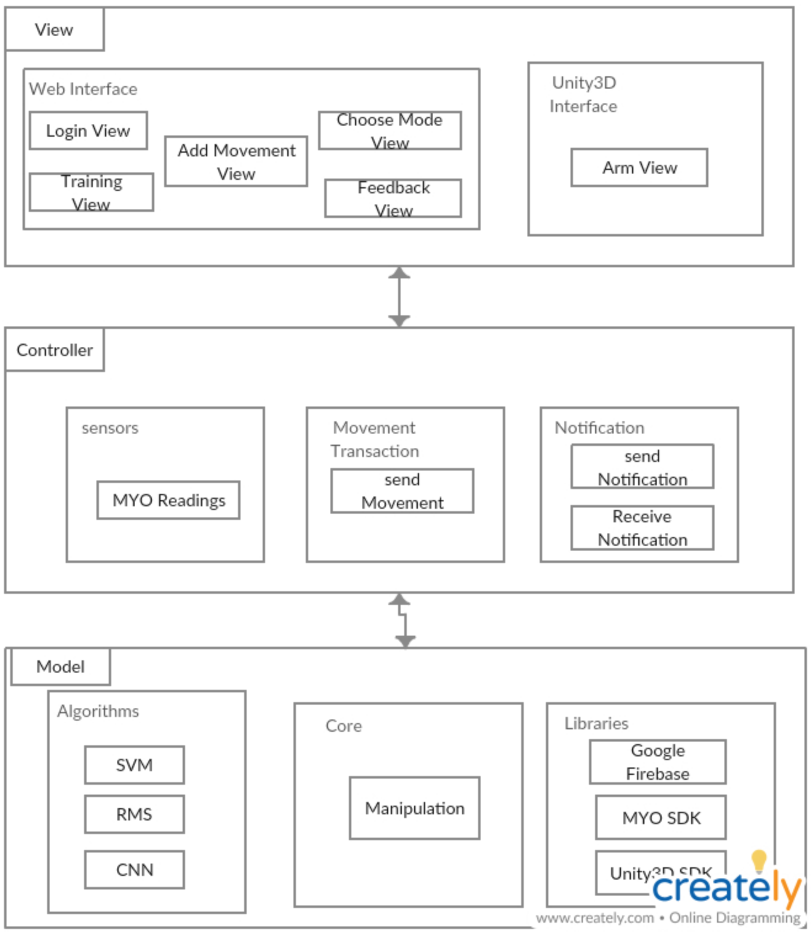


Figure 2: Architectural Design

then the user can Add new Movement to the System. Finally, the user can send his/her Feedback if he/she found any problem with the system.
Unity3D Interface :contains the Render of the Arm Movements.

3.1.2 Controller

This binds the View with the Model. Takes input from the View and sends it to the database and retrieves data from the Model and sends it to the view to be previewed there.
it also handles the EMG readings from the MYO sensors. it is responsible for Classified Movement Transaction to the Unity3D Render.

3.1.3 Model

Core:
Manipulation:
the signals pass through an equation to remove noise found in the Signals such as gravity to improve interpretation and computation on EMG Signals.
then the data is segmented to the size of the window that we set the data in that window is the data being used for the manipulation at that time.
When moving on to new data for the computation, make overlapping 50% then put segments on rmse feature extraction to pass it to classifier.
Algorithms:
contains Classification and Feature Extraction Algorithms as: RMS : stands for Root Mean Square which used for get most important features in Signals.
SVM : stands for Support Vector Machine which used for classifying the Signals to Movments samples.
CNN : stands for Convolutional Neural Network which will used for classifying and learn new Movements.
Libraries:
Google Firebase:
MYO SDK:
Unity3D SDK:

3.2 Model pipeline

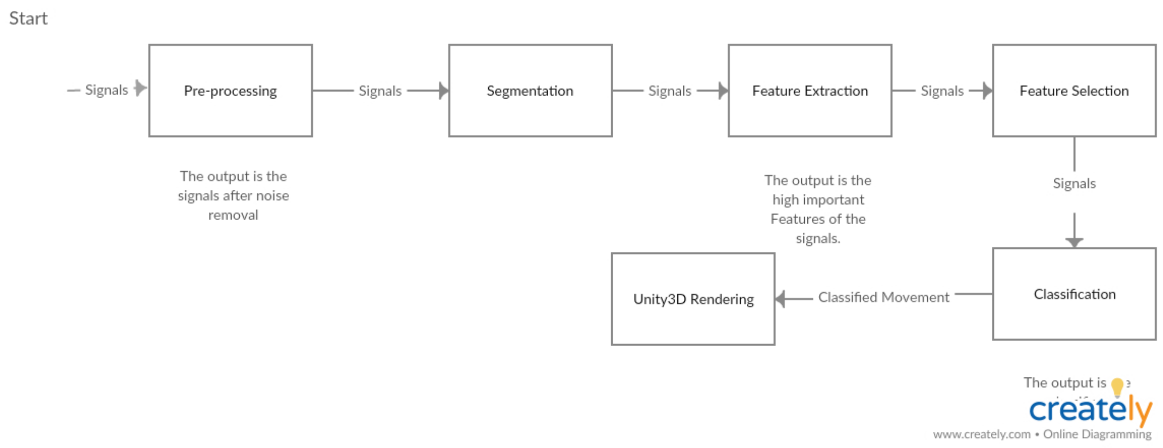


Figure 3: Architectural Design

3.3 Decomposition Description

3.3.1 Class Diagram

Class diagram with MVC Pattern.

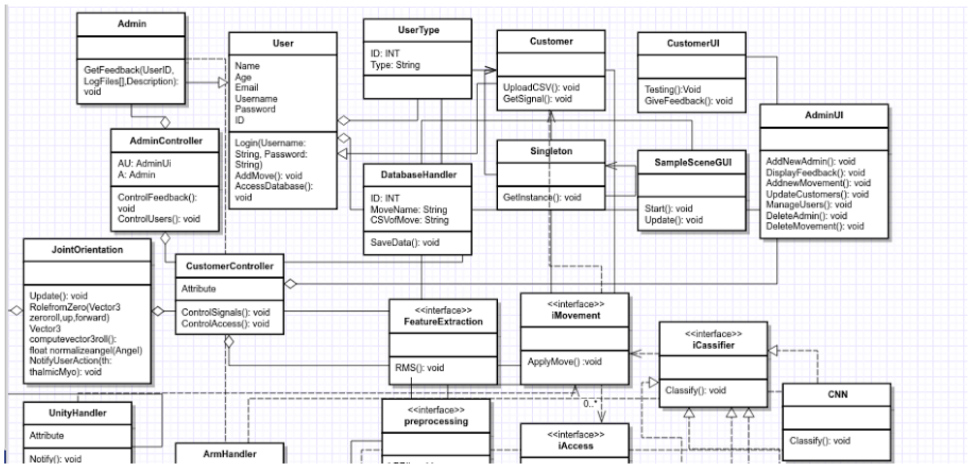


Figure 4: Class Digram

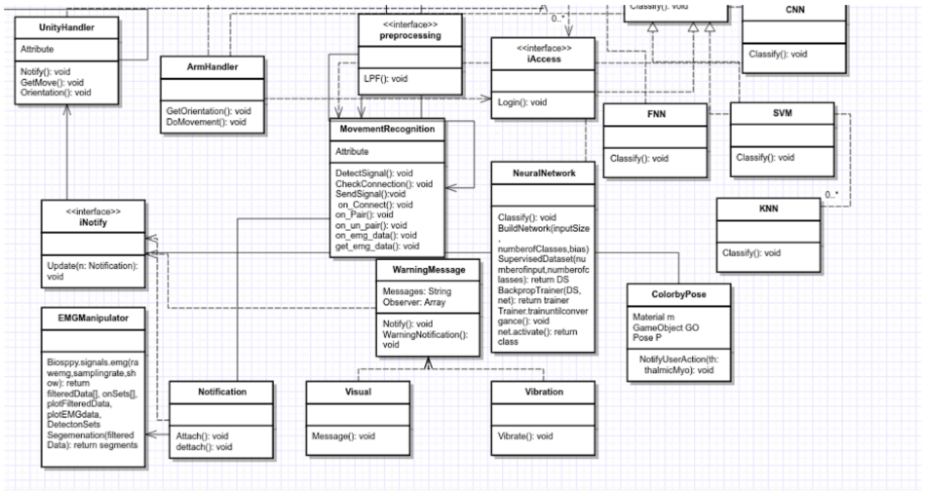


Figure 5: Class Digram

View Classes:

SampleSceneGUI: this class is a connection between the two classes colorbyPose and the jointrOrientation which they can control the arm movement,

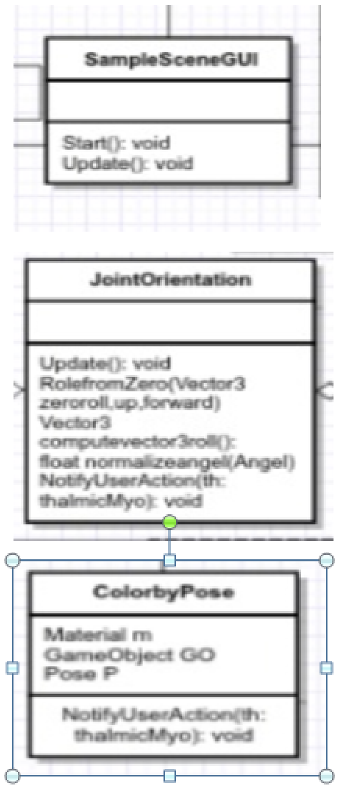

SampleSceneGUI
Start(): void
Update(): void
JointOrientation
Update(): void
RolefromZero(Vector3 zeroroll,up,forward)
Vector3 computevector3roll():
float normalizeangel(Angel)
NotifyUserAction(th: thalmicMyo): void
ColorbyPose
Material m
GameObject GO
Pose P
NotifyUserAction(th: thalmicMyo): void

Figure 6: Class Digram

color and orientation, the myo device will send the accelerometer, orientation and Euler orientation to the unity which then the system calculate in the world coordinates how the arm will be rendered according to this data while, also function normalize Angle will translate the angle from -180 to 180 in the world coordinates, the RolefromZero method will calculate when the angle of which the arm of the user has rotated between 180 and -180 by computing the cousin angle between the up vector and the zero vector which tells the system how much the MYO has rotated around, and the NotifyUserAction method will tell get an instance from the MYO object class and it will lock MYO if the movement of the user is recognized so it will not accept any new movement till the old one is rendered , while the colorByPose class is getting the game object from the scene by adding it from the editor window or we can hardcoded it by telling it the id of the cube while the material the system need to know if the user have fired a gesture the material will change according to his gesture this is an important issue as the system need to know the user intended movement while the sampleSceneGUI class first the system checks if the unity connected and synced if the MYO is synced then a message will appear on the screen to tell the user the state of the device currently of the device is not connected then the user can force it reconnect by pressing the q button on the keyboard, the pose object will get the pose from the MYO and the system will change the material according to it.

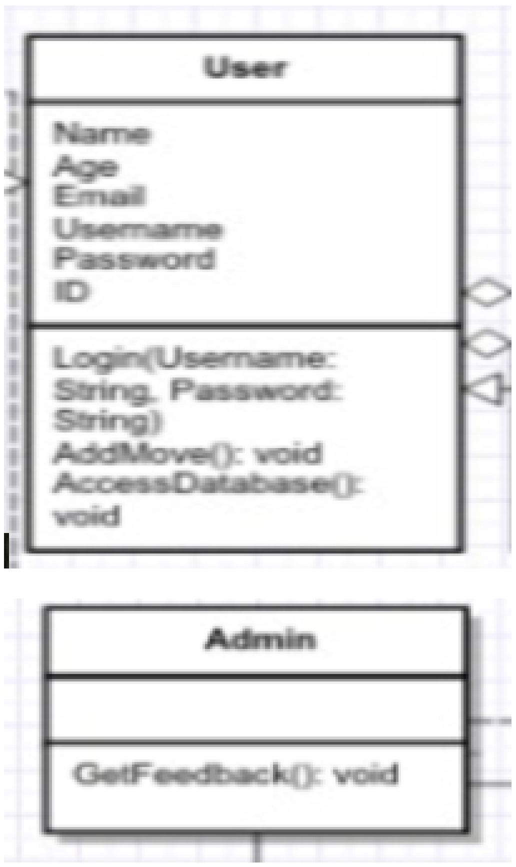


Figure 7: Class Digram

The user and the admin model classes are used for holding of the data that the system will use for saving the user data and sending them to the database each user will get password while creating his account while the admin will be able to check the userid and the required movement and the logfiles.

The adminUI class will be the page for viewing the admin and he/she can create new admin , remove admin also can add new movement or remove movement also they can delete admin the adminUI will hold some text fields and buttons for creating, editing and removing either of movement, user and admin.

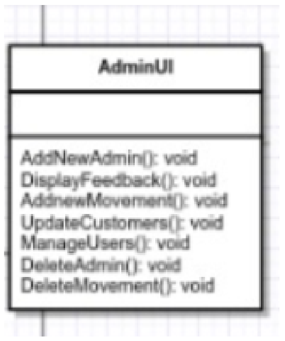


Figure 8: Class Digram

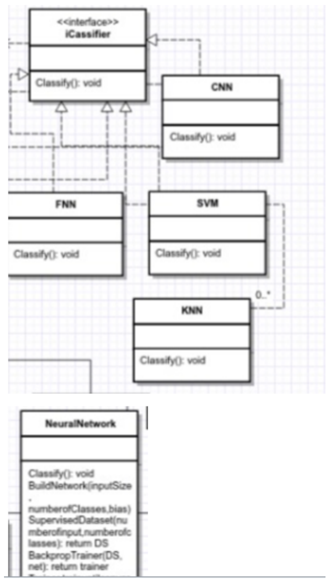


Figure 9: Class Digram

We have implanted the template design pattern for choosing between multiple classifier we have implemented the KNN which calculate the distance between each value and the other that would take a lot of time consumption it has accuracy about 88% with the best give distance Euclidean distance while the Manhattan about 84% accuracy , while the CNN (Convolution Neural Network) which the algorithm takes the segments and update the network layer until it reach convergence while SVM is the co-ordinates of individuals observation, which after that used to do some calculations on it to get the values between each class, while the FNN(feedforward neural network) we are still trying to implement it right now to if its better than the CNN or not, finally the NN is a network used by pybrain library which use buildnetwork method, and takes number of inputs and number of classes, also the supervisedDataset takes tha input and the number of classes, while the addSample is adding new input for classification, while the activate method is like predict method that returns in which class the value belongs .

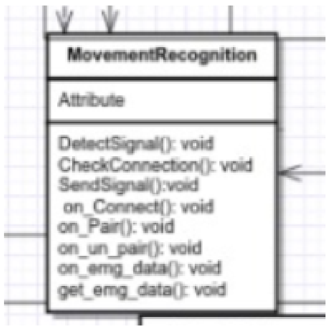


Figure 10: Class Digram

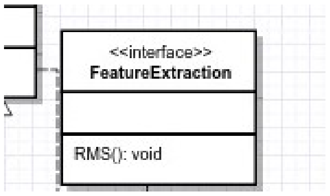


Figure 11: Class Digram

Feature extraction:
This class responsible to apply the feature extraction algorithms like rms which extract the feature from the segments that the function take .in this function can take more than one feature extraction algorithm to increase the accuracy and give many important feactures to make the classifier classify easier .

EMG manipulator:
In pre-processing class make noise removing from the original raw emg signals using LPD(low pass differential) and normalize the signals to present it in a straight line , also detect the onsets of the signals and can use that in segmentation by onsets . in pre processing class make segmentation with time windowing every 5 sec and make 50% overlapping and plot every segment in different plot. The preprocess in the system implemented by biosppy.emg library which responsible to handle many kinds of bio signals like emg , ecg and eeg.

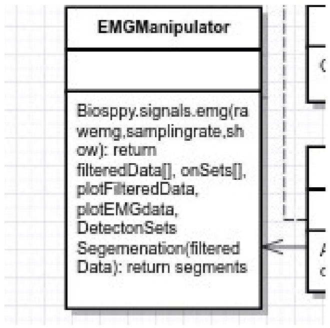

EMGManipulator
Biosppy.signals.emg(rawemg,samplingrate,show): return filteredData[], onSets[], plotFilteredData, plotEMGdata, DetectonSets
Segemenation(filtered Data): return segments

Figure 12: Class Digram

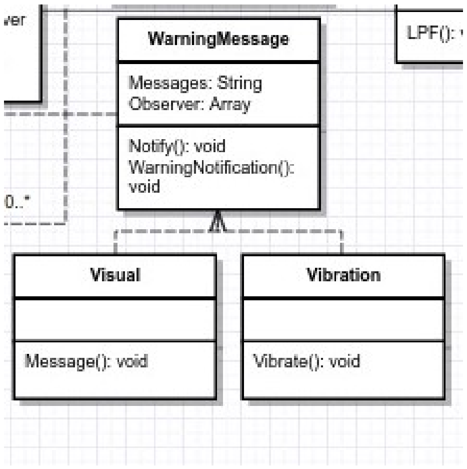


Figure 13: Class Digram

Warning message:
This class for the messages that the users receive when new update available or warn when any harm affect the system or violate the safety of the system or the user like any wrong classification and make some action like vibrating or generating visuals.

3.3.2 Sequence Diagram

This is a sequence diagram showing the sequence of Sending signals from User to the System fiqure[14].

This is a sequence diagram showing the sequence of Feedback system fiqure[15].

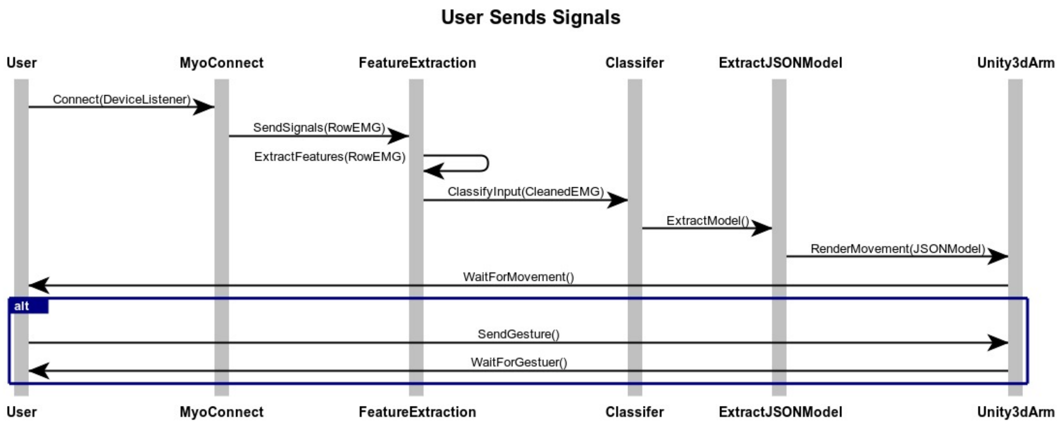


Figure 14: Sending Signals Sequence Diagram

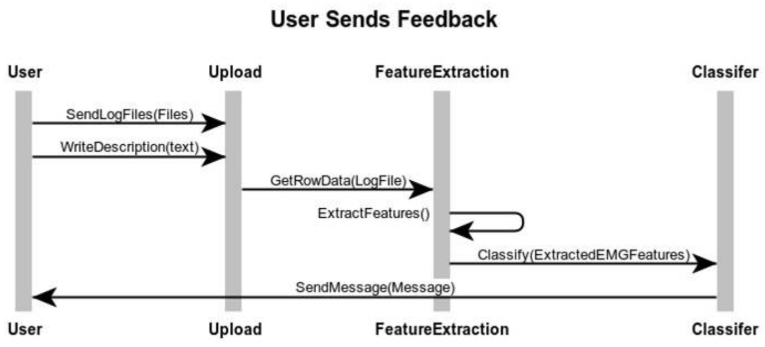


Figure 15: Feedback Sequence Diagram

3.4 Design Rationale

The system have two architecture :
First the MVC architecture
Finally the pipeline architecture
We have chose the MVC because it will help in separate the information layer and the presentation layer also we have unity3d arm which is considered as a GUI so we need MVC design architecture .
Also we have pipeline architecture since we need to classify data in sequence also in preprocessing.

3.4.1 CNN Algorithm

Convolutional neural network is a deep-learning algorithm. CNN is used for images and signals classification.
the design of CNN as follow:
Convolutional.
Pooling.
Fully connected.
Weights.
as shown in figure[5],the CNN architecture on an image.

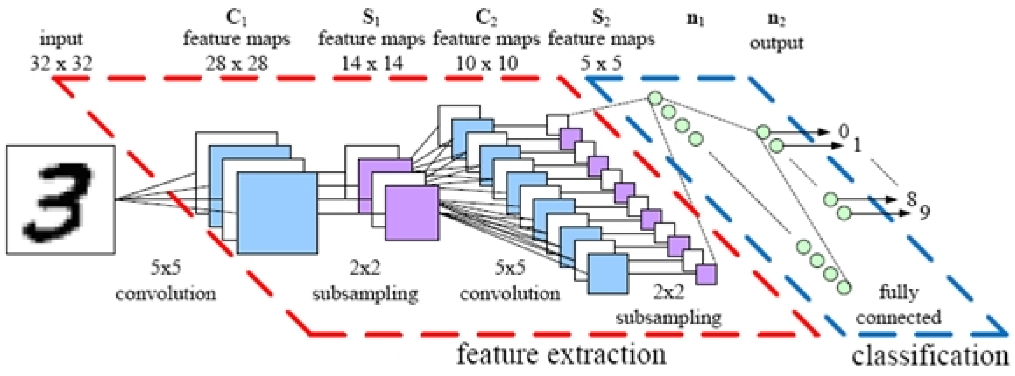


Figure 16: CNN Architecture

3.4.2 KNN Algorithm

K-nearest neighbor is a classification algorithm where a set of examples is collected and each one has a known class.
This set is used as a training dataset for the algorithm so when a new data is fed to it a comparison takes place between the training dataset and the new data.
in figure[17], how KNN Algorithm works.
KNN has different calculation methods; they are euclidean, Manhattan and cosine similarity distances[2].

3.4.3 SVM Algorithm

Support vector machine is a classification algorithm where a set of examples is collected and each one has a known class. the algorithm divides the classes by line and put each value to his class.

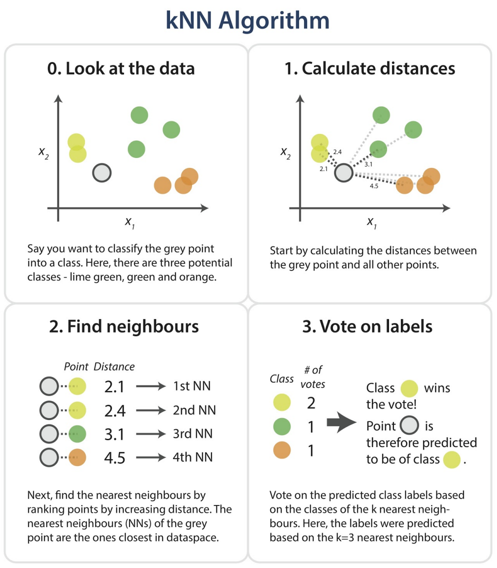

kNN Algorithm
0. Look at the data
x_2
x_1
Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.
1. Calculate distances
2.4
2.1
3.1
4.5
Start by calculating the distances between the grey point and all other points.
2. Find neighbours
Point Distance
2.1 → 1st NN
2.4 → 2nd NN
3.1 → 3rd NN
4.5 → 4th NN
Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.
3. Vote on labels
Class
# of votes
2
1
1
Class wins the vote!
Point is therefore predicted to be of class .
Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the k=3 nearest neighbours.

Figure 17: KNN Architecture

SVM algorithm not only for linear classification but it supports Multi Classes. in figure[3], how SVM works:

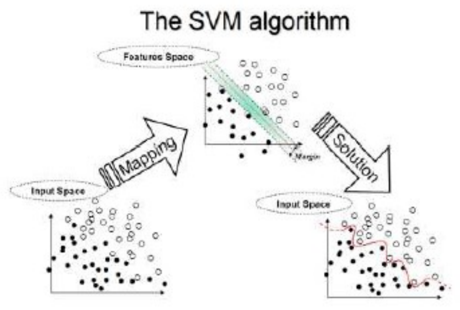


Figure 18: Linear SVM

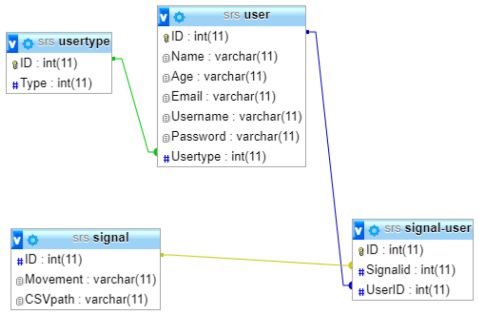


Figure 19: Linear SVM

4 Data Design

4.1 Data Description

4.2 Data Dictionary

Usertype:
In table usertype indicates to the user type, user can be customer or admin every type has id and type name.

User:
The table user carries all the info of users like name, username, password, age, email and ID. this table used to profile the user and get access in his account also connected by user type by user type id to know the user type if admin or customer and every time has his access in the system and has a different functionality.

Signal:

Table signals , the system store the csv files of raw Emg signals in this table and every user has his own signals , ths table store the raw emg signals and its name to can identify them.

5 Component Design

5.1 RTM: Real Time Mointering

n This Phase,myo-python, in realtime processing the system first checks the connection between the MYO and the processing unit with the on connect port

which connected to the CPU using blue-tooth device then the system collect the row EMG data from the MYO APIs which return the current reading of each sensor.
the listener class takes two parameters the first one is the port and the second one is the deviceListenerClass which contain on_connect which returns true if the MYO is connected, on pair this method checks if the MYO device is warmed up and working probably, on_unpair this method checks is the device is disconnected.
on_EMG_data this function returns the EMG. these are the methods used to connect with MYO for realtime processing the system read 50 array each array contains 8 sensor values from each reading and waits for 2 seconds between each read, after the system collects data.
in real-time the system pass it for pre-processing the system apply root mean square algorithm (RMS) after the value is returned the system then takes segments on for the RMS values returned then itś sent to the classifier we have tried many of them CNN, SVM, KNN, FNN, we have found that SVM accuracy is acceptable and it has some convenient speed at run-time it has accuracy of 95 % after the classifier finishes the extracted model is sent to the unity3d render to render the arm with the intended movement while the CNN has accuracy of 98% but it takes much more time to calculate the movement which is not accepted.

5.2 preproccessing

In this phase, In preprocessing, read the data and plot it then make noise removing on the raw data and plot it . take the filtered data and detect onsets on it to can make segmentation by onsets. take the filter data and segment it by time windowing and make overlapping 50% then put segments on RMSE feature extraction to pass it to classifier.
Biosspy:
Biosspy is a library can handle many kinds of bio-signals. the system uses this library to noise removing and detecting onsets of the filtered data. This library make noise removing by skip all the corrupted signals and normalize it to can represent it in straight line. Also, this library detects the onsets by make threshold (0.5) and (- 0.5) and the library also show the data after making all this functions by plotting.

5.3 RMS : Root Mean Square

RMS is an Feature extraction algorithm. RMS used to get the main features from our dataset to start working on it in the next phase.

In the case of a set of *n* values $\{x_1, x_2, \dots, x_n\}$, the RMS

$$x_{\mathrm{rms}} = \sqrt{\frac{1}{n}\left(x_1^2 + x_2^2 + \cdots + x_n^2\right)}.$$

The corresponding formula for a continuous function (or waveform) *f(t)* defined over the interval $T_1 \le t \le T_2$ is

$$f_{\mathrm{rms}} = \sqrt{\frac{1}{T_2 - T_1}\int_{T_1}^{T_2} [f(t)]^2\, dt},$$

and the RMS for a function over all time is

$$f_{\mathrm{rms}} = \lim_{T\to\infty} \sqrt{\frac{1}{T}\int_0^{T} [f(t)]^2\, dt}.$$

Figure 20: RMS Equations

5.4 SVM : Support vector machine

Support vector machine is a classification algorithm where a set of examples is collected and each one has a known class. The algorithm divides the classes by line and put each value to his class as figure[3]. as for instance, it does not suffer limitations of data dimensionality and limited samples [2][4]. the disadvantages of SVM is it is less effective on noisier datasets with overlapping classes.

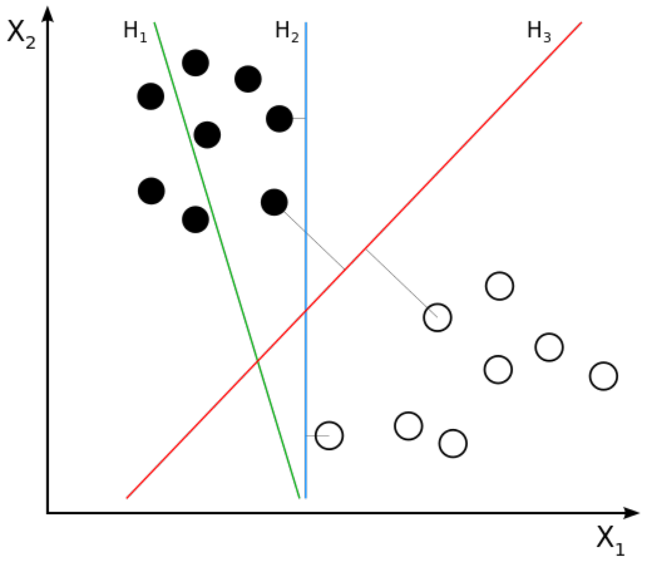


Figure 21: Linear SVM

Multi SVM is Multi-class algorithm where the labels are drawn from a finite set of several elements. The system uses Multi SVM to classify our 10 Movements on our system. Multi SVM deal as binary classifier by 2 samples and start with one of them with another one to start a new binary classification and complete to the end with these concept as shown in figure [4].

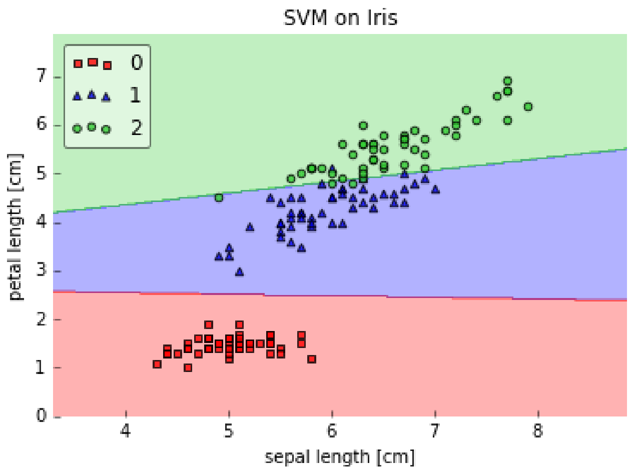


Figure 22: Multi SVM

Libraries: [from sklearn import svm],this Library for import predefiend SVM [from sklearn.svm import SVC],this Library to import SVC function

-Functions: SVC() function provides the Type of Kernal and C. RFE() function provides the number of feature selection.

6 Humnan Interface Design

6.1 Overview of User Interface

The system interface is going to divided in two main components a Desktop application and unity3d arm which is going to control signals coming from user and the desktop application is going only to manage user files and signals by logging in our system the user will have privileges like deleting his files or uploading new one then he will be able to download his files to train the system based on his profile.

6.2 Screen Images

6.3 Screen Objects and Actions

6.3.1 Training Window

in this window, the user learn how to use our system and start train the system with his own dataset.

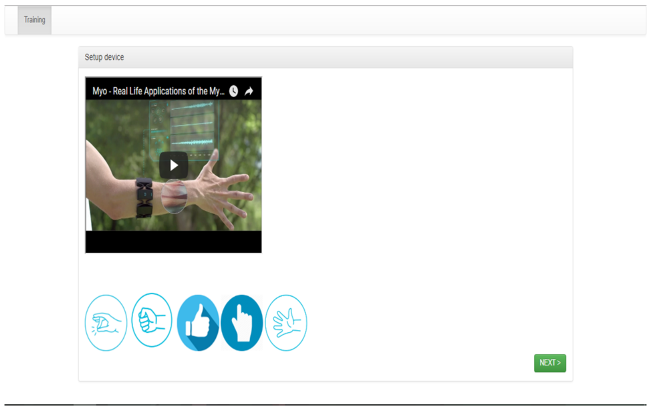


Figure 23: Training Window

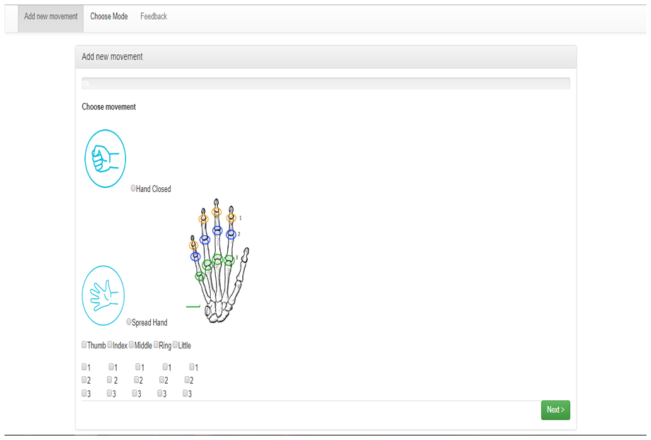


Figure 24: Add New Movement

6.3.2 Add New Movement

In this window, the user can add new Movement to our system by choose each Mode and the finger he/she want to move.

6.3.3 Feedback

In this window, User can send his feedback if he found any problem in the system.user writes a description of the problem and choose the file log.

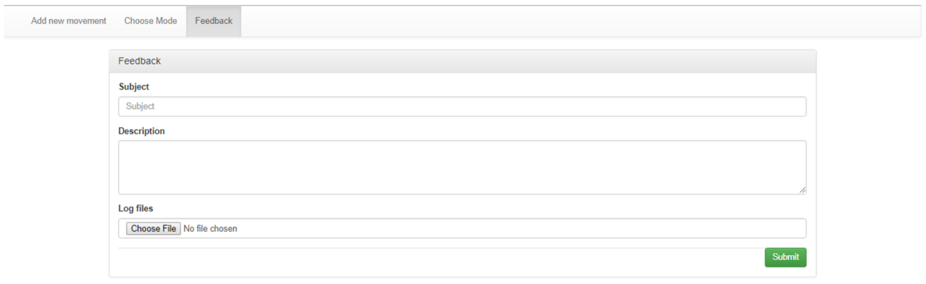


Figure 25: Feedback

7 Requirements Matrix

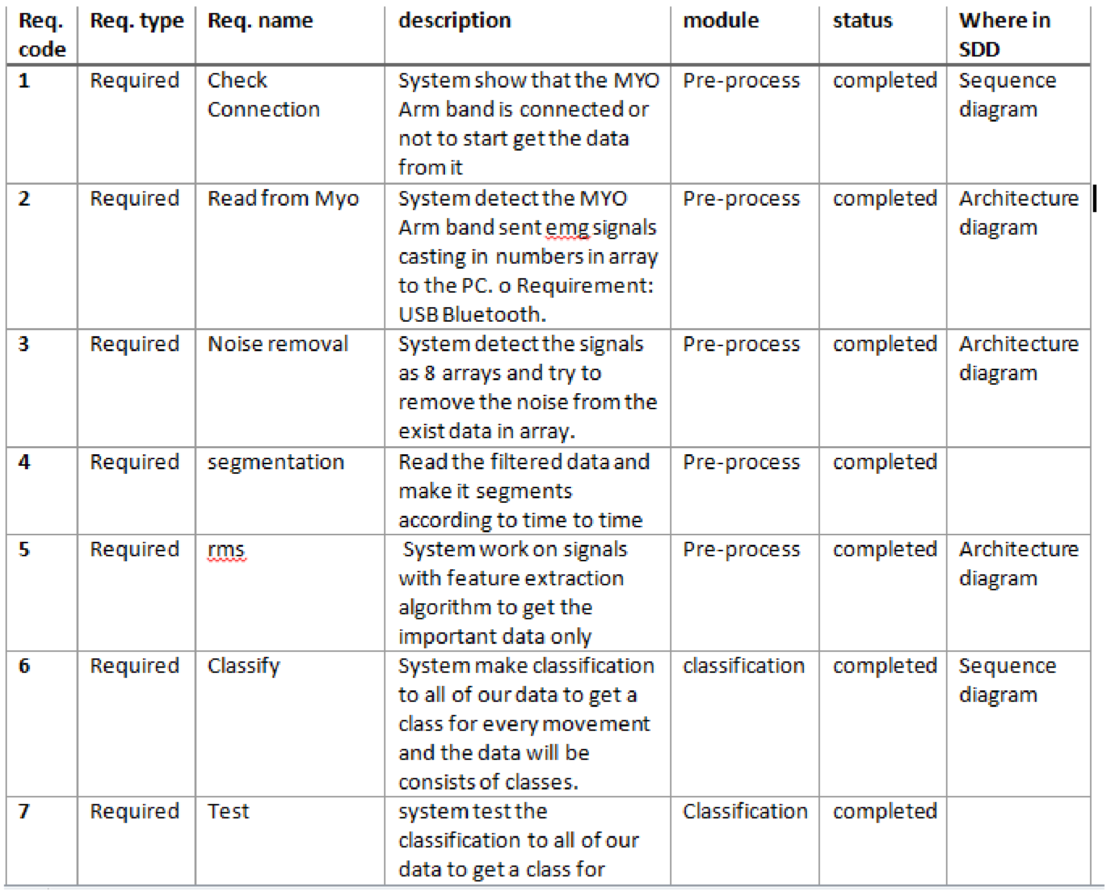
8 References

[1] K Srivastava Durgesh and B Lekha. Data classification using support vector machine. In: Journal of Theoretical and Applied Information Technology 12.1 (2010), pp. 17.

[2] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on Computational learning theory. ACM. 1992, pp. 144152.

[3] Sadegh Bafandeh Imandoust and Mohammad Bolandraftar. Application of k-nearest neighbor (knn) approach for predicting economic events: Theoretical background. In: International Journal of Engineering Research and Applications 3.5 (2013), pp. 605610.

[4] Vladimir Vapnik. The nature of statistical learning theory. Springer science and business media, 2013.



Req. code	**Req. type**	**Req. name**	**description**	**module**	**status**	**Where in SDD**
1	Required	Check Connection	System show that the MYO Arm band is connected or not to start get the data from it	Pre-process	completed	Sequence diagram
2	Required	Read from Myo	System detect the MYO Arm band sent emg signals casting in numbers in array to the PC. o Requirement: USB Bluetooth.	Pre-process	completed	Architecture diagram
3	Required	Noise removal	System detect the signals as 8 arrays and try to remove the noise from the exist data in array.	Pre-process	completed	Architecture diagram
4	Required	segmentation	Read the filtered data and make it segments according to time to time	Pre-process	completed	
5	Required	rms	System work on signals with feature extraction algorithm to get the important data only	Pre-process	completed	Architecture diagram
6	Required	Classify	System make classification to all of our data to get a class for every movement and the data will be consists of classes.	classification	completed	Sequence diagram
7	Required	Test	system test the classification to all of our data to get a class for	Classification	completed	

Figure 26: Requirements Matrix

8	Required	Split test	split dataset to make training and testing	Classification	completed	
9	Required	notify	Get notified with new movement the user add it and testing it	customer		Architecture diagram
10	Required	Send classified model	System sent the data to the arm to do it.	classification		Architecture diagram
11	Required	Draw movement	System compare the action to other action on arm handler to know which movement must execute	rendering	completed	Sequence diagram
12	Required	signup	Send user data to the database	customer		Architecture diagram
13	Required	login	Connect to server and retrieve user data	Customer		Architecture diagram
14	Required	Add new move	the user add new movement according to his needs which is specific to him	Customer		Architecture diagram
15	Required	Delete move	remove the movement from dataset	Customer		Class diagram
16	Required	Send feedback	submit and send the feedback.	Customer		Class diagram
17	Required	Upload csv	Send files to the server	Customer		Class diagram
18	Required	Choose mode	the classification is set according to the user selection	Customer		Architecture diagram

Figure 27: Requirements Matrix

19	Required	Add admin	Send user data to the server	Admin		Class diagram
20	Required	Delete admin	Deletes an admin from the database	Admin		Class diagram
21	Required	Update admin	Send user data to the server	Admin		Class diagram
22	Required	Delete move	Admin can remove records from data set of users.	Admin		Class diagram
23	Required	Real-time movement	Read data from myo in real time	Admin	completed	Class diagram

Figure 28: Requirements Matrix