Chapter 16

Artificial Human Arm Driven by EMG Signal

Mohammed Z. Al-Faiz and Abbas H. Miry

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/48531

1. Introduction

This chapter presents the anatomy of Electromyography (EMG) signal, measurement, analysis, and it's processing. EMG is the detection of the electrical activity associated with muscle contraction. It is obtained by measurement of the electrical activity of a muscle during contraction. EMG signals are directly linked to the desire of movement of the person.

Robot arms are versatile tools found in a wide range of applications. While the user moves his arm, (EMG) activity is recorded from selected muscles, using surface EMG electrodes. By a decoding procedure the muscular activity is transformed to kinematic variables that are used to control the robot arm. EMG signals have been used as control signals for robotics devices in the past. EMG signals, which are measured at the skin surface, are the electrical manifestations of the activity of muscles. It provides an important access to the human neuromuscular system. It has been well recognized as an effective tool to generate control commands for prosthetic devices and human-assisting manipulators. Up to the present, a number of EMG-based human interfaces have been proposed as a means for elderly people and the disabled to control powered prosthetic limbs, wheelchairs, teleoperated robots, and so on. The core part of these human–robot interfaces is a pattern classification process, where motions or intentions of motions are classified according to features extracted from EMG signals. Commands for device control are then generated from the classified motions (Bu et al., 2009).

It has been proposed that the EMG signals from the body's intact musculature can be used to identify motion commands for the control of an externally powered prosthesis. Information extracted from EMG signals, represented in a feature vector, is chosen to minimize the control error. In order to achieve this, a feature set must be chosen which maximally separates the desired output classes. The extraction of accurate features from the EMG signals is the main kernel of classification systems and is essential to the motion command identification (Park &Lee, 1998).

2. EMG signal fundamentals

EMG is the recording of the electrical activity produced within the muscle fibers. The relation of surface EMG to torque makes EMG an attractive alternative to direct muscle tension measurements, necessary in many physical assessments. However, the complexity of the EMG signal origin has been a barrier for developing a quantitative description of this relation. The EMG signal origin and character is necessary background to understand the difficulty of establishing a relationship between surface EMG and torque.

The nervous system controls the voluntary movement of various body parts in humans by contracting and relaxing various skeletal muscles. To instantiate a contraction, a neuron generates a small electrical potential on the surface of the muscle fiber. This electrical potential causes depolarization of the muscle fiber tissue and a following depolarization waveform. This waveform travels the length of the muscle fiber and is known as the Action Potential (AP). Fig. 1 depicts the generation of electric fields in muscle fibers.

Muscle fibers are excited by nerve branches by one motoneuron in groups known as motor units. These motor units are defined as the fundamental unit of contraction and can range from a few muscle fibers for small muscles such as those in the hand and fingers, to thousands of muscle fibers in large muscles such as those in skeletal muscle. Because each motor unit contains a number of muscle fibers that are attached to the motor neuron at various points, the electrical signal of a motor unit is the summation of the action potential of each muscle fiber, which may be phase shifted from the other muscle fibers in that unit (Perry & Bekey,1981).

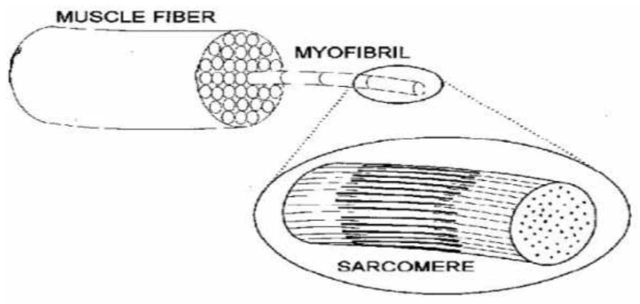


Figure 1. Muscle Fibers Composition

This notion is reinforced in Fig. 2. The electrical potential due to contraction of all fibers in a motor unit during a single activation is referred to as the Motor Unit Action Potential (MUAP). This MUAP can be recorded by using electrodes placed on the surface of the skin above the muscle. Also, a muscle is not typically excited by only one action potential. In order to hold a contraction for any length of time, the motor units must be repeatedly activated. This repeated activation gives rise to a series of MUAPs that can be modeled as a pulse train in classical signal processing terms. This series of MUAPs that is produced is referred to as a Motor Unit Action Potential Train (MUAPT). When the electromyography measured using a surface electrode, the electromyography can be defined as the superposition of numerous MUAPTs firing asynchronously. Fig. 3 reinforces the notion that the superposition of motor unit action potentials gives rise to surface EMG. The surface electromyography signal typically does not exceed 5-10 mV in amplitude with the majority of signal information being contained between the frequencies of 15 and 400 Hz. As a result, the amplitude of the EMG contains a great deal of the signal information which can be modeled as a Gaussian random process. The EMG amplitude can thus be defined as the time-varying standard deviation of the EMG signal and is a measure of the activity level of the muscle under observation.

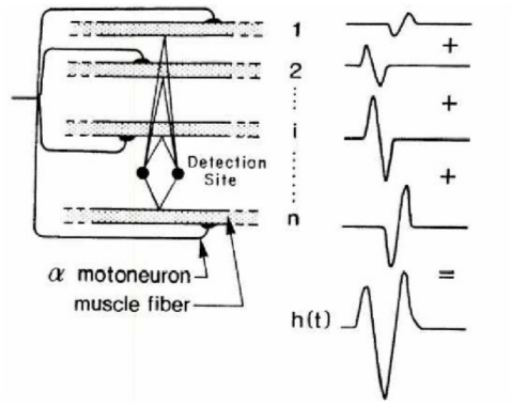


Figure 2. MUAP with Phase shifted from the other muscle fibers

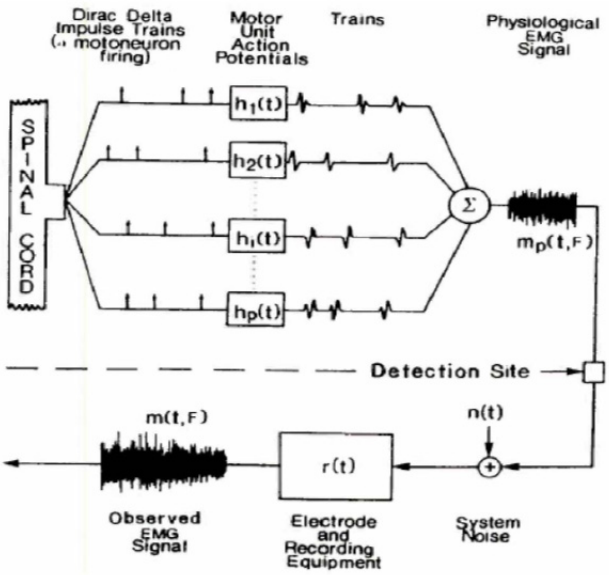


Figure 3. Superposition of Motor Unit Action Potential Gives Rise to Surface EM

3. Prosthetic limb developments

As previously discussed EMG signals provide a non-invasive measure of ongoing muscle activity. Therefore, EMG signals can be potentially used for controlling robotic prosthetic devices. Most prosthetic devices that are currently available usually only have one degree-of-freedom. As a result, these devices provide nowhere near the amount of control as the original limb which they are intending to replace. Through clinical research, it has been shown that amputees and partially paralyzed individuals typically have intact muscles that they can exercise varying degrees of control over. As a result, research is being conducted in regards to utilizing the signals from these intact muscles to control robotic devices with multiple degrees of freedom (Beau 2005). The EMG has been used in two manners in the area of prosthetic limb development. The first approach is for the subject to exert a force with a particular muscle. This force results in a steady-state EMG signal amplitude estimate. A degree of freedom of a robotic limb is then moved in proportion to the EMG amplitude. This described approach is used in the control of a standard prosthetic gripper that has one degree-of-freedom (Beau 2005).

The second manner that EMG signals are used involves discrete actions. When a discrete action is performed, such as the quick movement of the hand or arm, the surface EMG is obtained from various muscle cites. The temporal structure of the transient EMG activity is then analyzed. Upon analyzing the transient EMG activity, various movements can be classified. Hence, EMG signals can be used in the development of advanced prosthetic devices that have various degrees-of-freedom.

4. Muscle anatomy

Agonist-antagonist muscles exist in many human joint. Such human joint is usually activated by many muscles .The following is a summary of the muscles that are responsible for the movement of the arm, wrist, and hand. Abduction of the arm is performed by the deltoid. Human elbow is mainly actuated by two antagonist muscles: biceps and triceps, although it consists of more muscles. Consequently, biceps and a part triceps are bi-particular muscles. By adjusting the amount of force generated by these muscles, the elbow angle and impedance can be arbitrary controlled (Kiguchi et al., 2001). Contraction of the biceps brachii flexes the elbow. Contraction of triceps brachii extends the elbow. Most of the muscles that move the forearm and hand originate within the forearm. The extensor carpi radialis produces extension and abduction of the wrist. The flexor carpi ulnaris flexes and adducts the wrist (Elliott,1998).

5. Feature parameters

EMG classification is one of the most difficult pattern recognition problems because there exist large variations in EMG features. Especially, it is difficult to extract useful features from the residual muscle of an amputee. So far, many researches proposed many kinds of EMG feature to classify posture and they showed good performance. However, how to select a feature subset with the Best discrimination ability from those features is still an issue for classifying EMG signals (Huang et al., 2003). The success of any pattern classification system depends almost entirely on the choice of features used to represent the raw signals. It is desirable to use multiple feature parameters for EMG pattern classification since it is very difficult to extract a feature parameter which reflects the unique feature of the measured signals to a motion command perfectly. But the inclusion of an additional feature parameter with a small separability may degrade overall pattern recognition performance. The feature parameters of EMG signal are listed in Table .1 .

Feature Parameters (Phinyomark & Baraani ,2009)	
1. Integrated EMG	$IEMG = \sum_{n=1}^{N} \lvert h_n \rvert$
2. Mean Absolute Value	$MAV = \frac{1}{N}\sum_{n=1}^{N} \lvert h_n \rvert$
3. Modified Mean Absolute Value	$MMAV = \frac{1}{N}\sum_{n=1}^{N} \lvert h_n \rvert W_n \qquad W_x = \begin{Bmatrix} 1 & 0.25N \le n \le 0.75N \\ 0.5 & otherwise \end{Bmatrix}$
4. Variance of EMG	$VAR = \frac{1}{N-1}\sum_{n=1}^{N} h_n^2$
5. Waveform Length	$WL = \sum_{n=1}^{N-1} \lvert h_{n+1} - h_n \rvert$
6. Wilson Amplitude(WAMP)	$WAMP = \sum_{n=1}^{N-1} f(\lvert h_{n+1} - h_n \rvert),\ f(x) = \begin{Bmatrix} 1 & x \ge threshold \\ 0 & otherwise \end{Bmatrix}$

Table 1. Feature Parameters of EMG signal

The MATLAB code of this action is:

```
%% program of Building Data Base
function data_base =build_data_base
global mscl1 mscl2 mscl3 mscl4 mscl5
ths=50;interval=1600;
IIEMGT=IEMGT';IIEMGT=IIEMGT(:);
VVARR=VARR';VVARR=VVARR(:);
MMAV=MAV';MMAV=MMAV(:);
MMMAVT1=MMAV1';MMMAVT1=MMMAVT1(:);
WWAMPT=(WAMP(ths))';WWAMPT=WWAMPT(:);
WLTT=WL'; WLTT=WLTT(:);
data_base =[IIEMGT';VVARR';MMAV';MMMAVT1';WWAMPT';WLTT'];
```

6. K-nearest neighbor (KNN) algorithm

The K-nearest neighbor (KNN) classification rule is one of the most well-known and widely used nonparametric pattern classification methods. Its simplicity and effectiveness have led it to be widely used in a large number of classification problems. When there is little or no prior knowledge about the distribution of the data, the KNN method should be one of the first choices for classification. It is a powerful non-parametric classification system which bypasses the problem of probability densities completely.

Nearest Neighbor (NN) is a "lazy" learning method because training data is not preprocessed in any way. The class assigned to a pattern is the class of the nearest pattern known to the system, measured in terms of a distance defined on the feature (attribute) space. On this space, each pattern defines a region (called its Voronoi region). When distance is the classical Euclidean distance, Voronoi regions are delimited by linear borders. To improve over 1-NN classification, more than one neighbor may be used to determine the class of a pattern (K-NN) or distances other than the Euclidean may be used. The KNN rule classifies χ by assigning it the label most frequently represented among the K nearest samples; this means that, a decision is made by examining the labels on the K-nearest neighbors and taking a vote KNN classification was developed from the need to perform discriminate analysis when reliable parametric estimates of probability densities are unknown or difficult to determine (Parvin et al., 2008).

K-NN is the most usable classification algorithm. This algorithm operation is based on comparing a given new record with training records and finding training records that are similar to it. It searches the space for the k training records that are nearest to the new record as the new record neighbors. In this algorithm nearest is defined in terms of a distance metric such as Euclidean distance. Euclidean distance between two records (or two points in n-dimensional space) is defined by:

If χ_1=(χ_{11}, χ_{12},..., χ_{1n}) and χ_2=(χ_{21}, χ_{22},..., χ_{2n}) then

$$dist(\chi_1,\chi_2)=\sqrt{\sum_{i=1}^{n}(\chi_{1i}-\chi_{2i})^2} \tag{1}$$

Where χ_1 and χ_2 are two records with n attributes. This Formula measures the distance between two patterns χ_1 and χ_2 (Moradian & Baraani, 2009). The K-nearest neighbor classifier is a supervised learning algorithm where the result of a new instance query is classified based on majority of the K-nearest neighbor category. The training samples are described by n-dimensional numeric attributes. Each sample represents a point in an n-dimensional pattern space. In this way, all of the training samples are stored in an n-dimensional pattern space.

The following discussion introduces an example demonstrating the general concept of this algorithm in detail. The K nearest neighbor algorithm is very simple. It works based on minimum distance from the query instance to the training samples to determine the nearest neighbors. After we gather K nearest neighbors, we take simple majority of these K-nearest neighbors to be the prediction of the query instance. The data for KNN algorithm consists of several multivariate attributes names that will be used to classify the object Y. Suppose that the K factor is set to be equal to 8 (there are 8 nearest neighbors) as a parameter of this algorithm. Then the distance between the query instance and all the training samples is computed. Because there are only quantitative , the next step is to find the K-nearest neighbors. All training samples are included as nearest neighbors if the distance of this training sample to the query is less than or equal to the Kth smallest distance. In other words, the distances are sorted of all training samples to the query and determine the Kth minimum distance. The unknown sample is assigned the most common class among its k nearest neighbors. As illustrated above, it is necessary to find the distances between the query and all training samples. These K training samples are the closest k nearest neighbors for the unknown sample. Closeness is defined in terms of Euclidean distance (Bawaneh et al., 2008).

Let us consider a set of patterns $\chi = \{\chi_1, ..., \chi_N\} \subseteq R^P$ of known classification where each pattern belongs to one of the classes $CR = \{CR_1, CR_2, ..., CR_S\}$. The nearest neighbor (NN) classification rule assigns a pattern Z of unknown classification to the class of its nearest neighbor, where $\chi_i \in \chi$ is the nearest neighbor to Z if

$$Dist(\chi_i, Z) = \min\left\{Dist(\chi_l, Z) \quad l = 1, 2, ..., N\right\} \tag{2}$$

Dist is the Euclidean distance between two patterns in R^P. This scheme is called the 1-NN rule since it classifies a pattern based on only one neighbor of Z. The k-NN rule considers the k-nearest neighbors of Z and uses the majority rule. Let t_l where l=1, 2,..,s be the number of neighbors from class l in the k-nearest neighbors of Z (Pal & Ghosh, 2001).

$$\sum_{l=1}^{s} t_l = k \tag{3}$$

Then Z is assigned to class j if

$$\underbrace{\mathrm{t{=}Max}\{\mathrm{t}_1\}}_{1} \tag{4}$$

Here is step by step on how to compute KNN algorithm:

1. Determine parameter K = number of nearest neighbors.
2. Calculate the distance between the query-instance and all the training samples.
3. Sort the distance and determine nearest neighbors based on the K-th minimum distance.
4. Gather the category Y of the nearest neighbors.
5. Use simple majority of the category of nearest neighbors as the prediction value of the query instance.These steps are summarized in Fig.4.

The MATLAB program of K-NN is :

```
%%%%  main program of KNN
Dst=dist(IF,DB);nn=6;[a ,b]=sort(Dst);
st=floor(b/nn)+1;
kk=15;y=st(1:kk)';x=ones(length(y),1);
a1= y ==1;b1=sum(x(a1));
a2= y ==2;b2=sum(x(a2));
a3= y ==3;b3=sum(x(a3));
a4= y ==4;b4=sum(x(a4));
a5= y ==5;b5=sum(x(a5));
bb=[b1 b2 b3 b4 b5];
[a ,b]=sort(bb);af=fliplr(a);bf=fliplr(b);
result=bf(1)
```

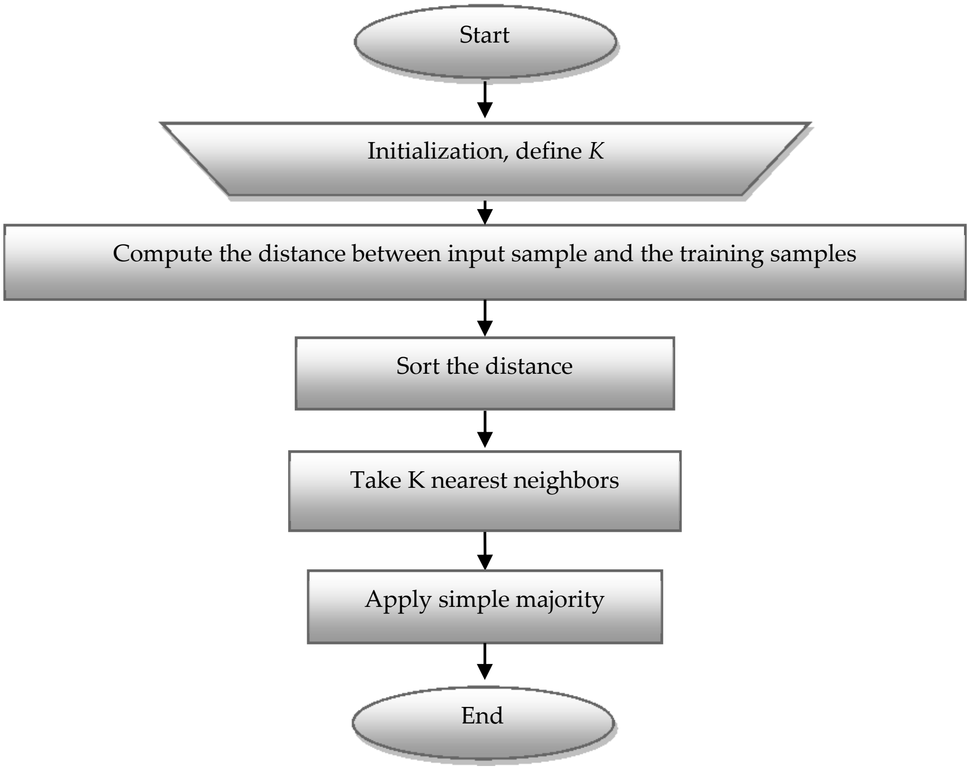

Start
Initialization, define *K*
Compute the distance between input sample and the training samples
Sort the distance
Take K nearest neighbors
Apply simple majority
End

Figure 4. Flowchart for the K- nearest neighbors

7. Measurement of Real EMG signal

EMG signal has two main sources, one of them is measured real EMG and the other one is generated EMG by EMG Simulator. Every one of them has advantage and disadvantage with respect to accuracy and reliability. There are many problems in real measurement system as follows: The first stage is the sensor of EMG signal (electrodes), there are two problems related with the considered structure. The first problem related with the electrode types, that is a needle type. This problem can be summarized as follows: the procedure for using this type by inserting the needle in proper placement on muscle to touch the fiber and sensing the EMG signal. One disadvantage of this needle type is caused high pain to human and it has side affect if using for long times, and it is used only with up normal muscle to check the activity or response of nerve that supply this muscle. Another problem is fixing the needle on the skin, where at any movement of muscle the needle will go out. To overcome the above mentioned problems, the needle may be replaced by either surface electrode or integral surface electrode. The main advantage for using the needle electrode is the amplitude of measured signal where it is better than using another types. In the needle case, the EMG signal is measured directly from MU. The second problem of obtaining EMG signals measured by a needle is only one channel exist for measuring in the same time. This is huge problem in the design where it needs number of channels equal to number of muscles that caused the required recognized movements. If one likes to recognize the wrist joint movement he needs four channels. To overcome the problems in real measuring system, the user can use the generation of EMG signal by EMG simulator.

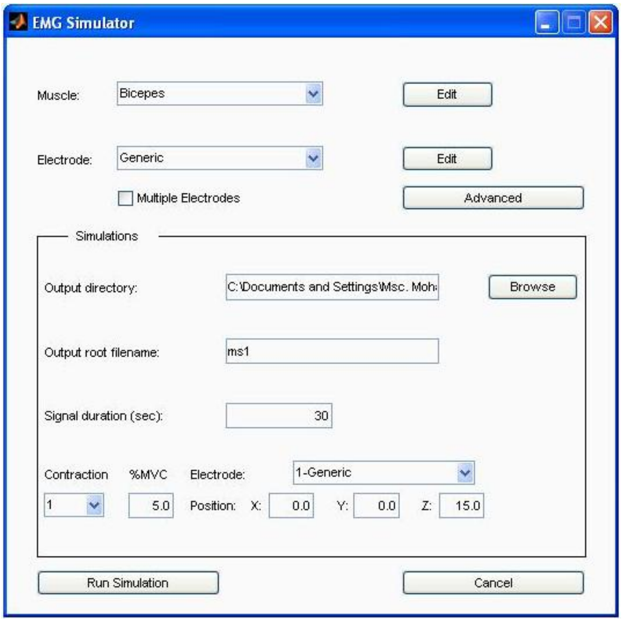
8. Simulation for EMG signal generation (EMG simulator)

The best modeling of clinical EMG signals was achieved in algorithm by A. Hamilton and D. W. Stashuk at 2005 (Hamilton & Stashuk ,2005). This algorithm is simulated by using MATLAB software and using the GUI approach to get full mathematical simulated model for generating real EMG signal of a specific human arm muscle as shown in Fig. 5 .

This simulator has many options used with rearrangement to generate EMG signal for human arm muscle. The option of the simulator can be summarized as:

Muscle: This popup shows the muscle being simulated. One can select from the list (which adding for human arm muscles), of already defined muscles or select "Custom." to define a new one. Clicking the edit button allows one to modify the muscle parameters. Helping for specification of human arm muscles is added to this window to help the user to generate EMG data more nearest to clinical data, Fig. 6 shows the window for this option.

Electrode: This popup shows the electrode being simulated. The user can select from the list of already defined electrodes or select "Custom..." to define a new one. Clicking the edit button allows the user to modify the electrode parameters. The "Electrode" panel allows the user to save new or modified sets of electrode parameters. Selecting "Multiple Electrodes" allows the user to simulate simultaneous recordings from more than one electrode. Select "Add" from the number popup to add a new electrode. Each electrode can be of a different type. If the user specifies multiple electrodes, then the program creates separate data files for each electrode. The signal from the first electrode is still named filename.dat, and the signals from the other electrodes are named filenameI.dat, where "I" is the electrode number. Fig .7 shows the window for this option.


EMG Simulator

Muscle: Bicepes
Electrode: Generic
Multiple Electrodes

Simulations

Output directory: C:\Documents and Settings\Msc. Moh
Output root filename: ms1
Signal duration (sec): 30
Contraction: 1
%MVC: 5.0
Electrode: 1-Generic
Position: X: 0.0 Y: 0.0 Z: 15.0

Figure 5. The EMG Signal Simulator.

Advanced: This allows the user to specify some advanced simulation preferences. Include all units: This causes the firing patterns of all the active motor units to be included in the annotation file, not just the ones closest to the electrode.

Output directory: Specifies the directory in which to write the data files. The default directory is the data subdirectory in the simulator directory.

Output root filename: Specifies the root filename for the output files.

Signal duration: Specifies the length of each signal, in seconds.

Contraction: Selects a contraction. To add additional contractions select "Add" from the popup.

%MVC: Specifies the contractile level for the selected contraction.

Position: Specifies the x, y, and z electrode locations (in mm) for the selected contraction. The z coordinate is the distance from the muscle endplate along the muscle axis. Use the "electrode" popup to specify locations for multiple electrodes. Note that you can specify different locations for each electrode in each contraction.

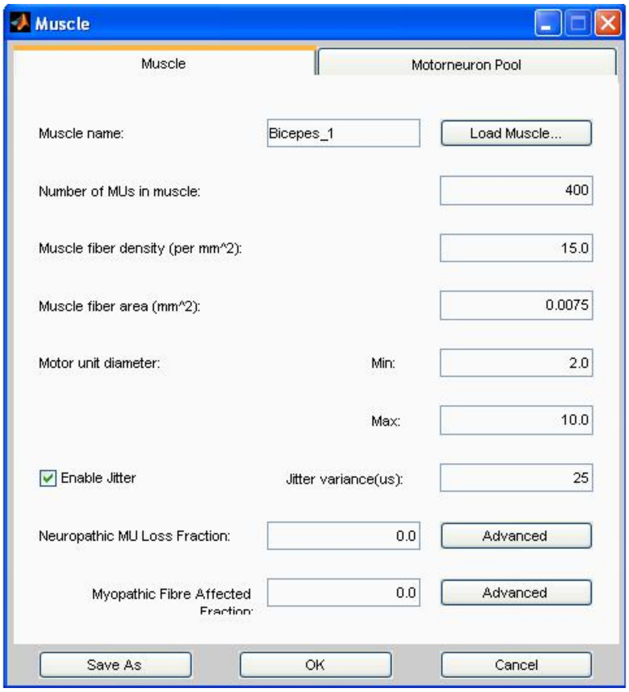
Delete: Deletes the current contraction.

Run Simulation. Runs the specified set of simulations (one simulation per electrode per contraction). The simulations can take a fair amount of time, depending on the specified signal duration. The trace statements from the simulation routines are displayed in the command window.

Cancel. Quits the simulator and returns to MATLAB.

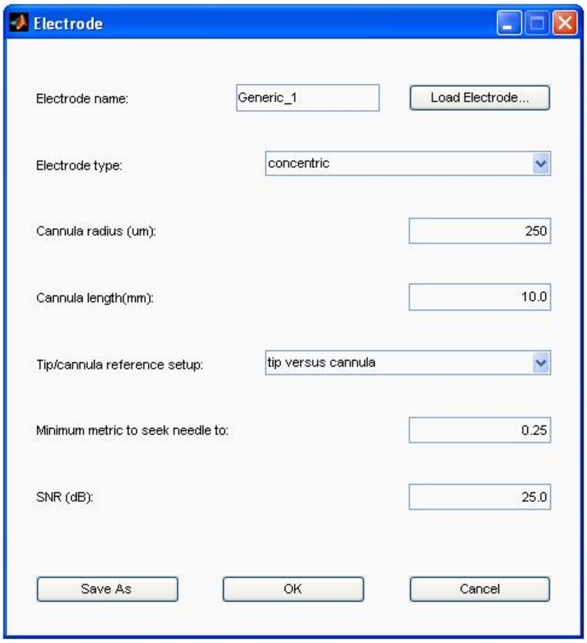
After press the " Run Simulation " button , each simulation may take a minute or more. The program creates the following files for each contraction:

Filename.dat →→ EMG signal, Filename.hea →→ header file (allows signals to be read by software).,Filename.eaf. →→ annotation file, These files can be used to analysis the EMG signal. The real value of the muscle parameters can be obtained by studying the anatomy of muscles in detail. Depending on the practical data obtained from medical table and consultation with the specialist, the muscle parameters adopted are as shown in table 2. Fig.5 represents the generation of EMG signal by EMG simulator. There are two choices to get the EMG data. The first one is by EMG simulator with specified values for the parameters of the muscle as shown in figure 6. The second choice is used to produce data of EMG signal by selecting a specific muscle of the human arm as shown in figure 7. The data are calculated by selecting the specification of normal human body as given in table 2



Muscle name: Bicepes_1
Number of MUs in muscle: 400
Muscle fiber density (per mm^2): 15.0
Muscle fiber area (mm^2): 0.0075
Motor unit diameter: Min: 2.0
Max: 10.0
☑ Enable Jitter Jitter variance(us): 25
Neuropathic MU Loss Fraction: 0.0
Myopathic Fibre Affected Fraction: 0.0

Figure 6. Window for the Selection or Design of Human arm Muscle



Electrode name: Generic_1

Electrode type: concentric

Cannula radius (um): 250

Cannula length(mm): 10.0

Tip/cannula reference setup: tip versus cannula

Minimum metric to seek needle to: 0.25

SNR (dB): 25.0

Figure 7. Window for Selecting the Type of Electrode

Muscle name	No. of MU	Fiber Density (per mm^2)	Fiber Area (mm^2)	Range MU diameter (mm)
biceps	400	15	0.0075	2 - 10
triceps	350	10	0.0055	2 – 9
deltoid	450	20	0.0085	3 - 11
extensor carpi radial	250	8	0.0035	1.5 – 8
flexor carpi ulna	100	5	0.0015	2 – 6

Table 2. Standard Parameters of Muscle Human Arm

9. Simulation for reading EMG signals generated by EMG simulator (EMG Lab)

The program, which explains the graphic of the files obtained from the EMG simulator, is called EMGlab (EMGLAB software,2008). This program is built in MATLAB software and the description of run EMGlab in MATLAB command window as follows. The program runs in a single MATLAB window, which is divided into five panels: At the top is the signal panel, which displays a segment of the EMG signal. Below that is the template panel, which displays the MUAP templates. Below that on the left is the firing panel, which displays the firing patterns of the identified MUs. To the right is the close-up panel, which displays a section of the EMG signal at an expanded scale. At the very bottom is the navigation panel which displays a thumbnail of the EMG signal. The buttons on the edges of the panels are used to change the display characteristics .

Fig. 8 shows the output of EMGlab with data of EMG signal. It receives the data that are generated by EMG simulator and the practical measured data and explain the decomposition of the signal and register the data of this signal. The inserted symbols in the window are:

+, - zoom in or out vertically

<, > scroll left or right, and | |, | | zoom in or out horizontally

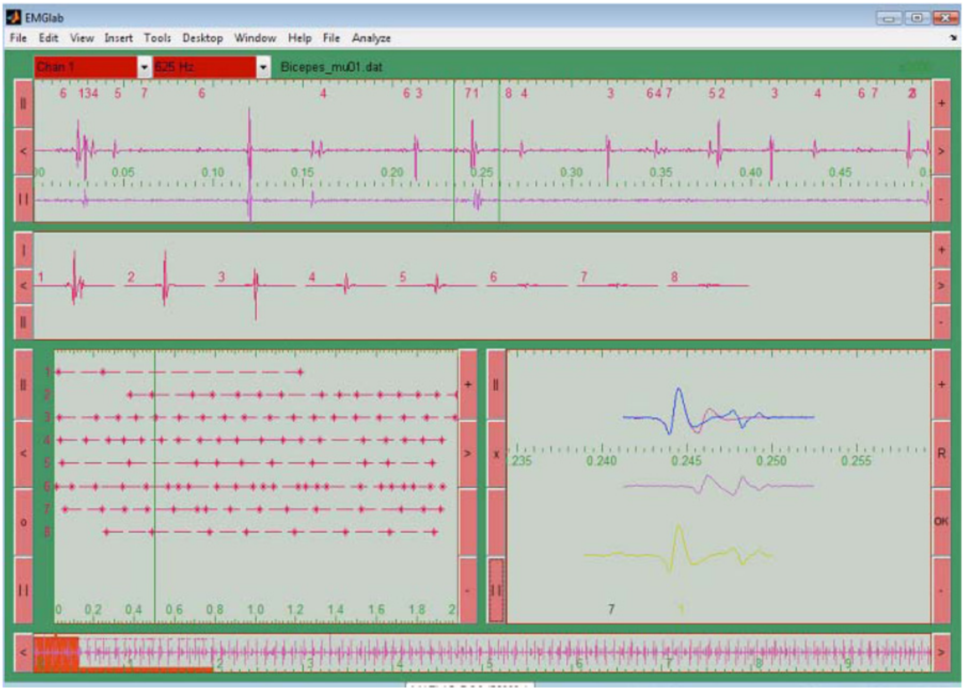

EMGlab
File Edit View Insert Tools Desktop Window Help File Analyze
Chan 1
625 Hz
Biceps_mu01.dat
R
OK

Figure 8. The Output of EMGlab for EMG Signal

10. Human arm movements recognition based on k-nearest neighbor algorithm

The discrimination of the EMG signal into the correct class of movement is a fundamental element of the system. The precision of a classifier lies on its capability to give the correct answers in spite of some inaccuracies that may occur during the process of detection of the EMG signal.

To improve precision of a classifier with decrease of the training time, a recognition system based K-NN algorithm is used. The success of any pattern classification system depends almost entirely on the choice of features used to represent the raw signals. In the proposed system multiple feature parameters for EMG pattern classification are used since it is very difficult to extract a feature parameter which reflects the unique feature of the measured signals to a motion command perfectly.

Five kinds of arm motion are recognized: Abduction of the arm, flexion the elbow, extension the elbow, extension and abduction of the wrist and flexes and adducts the wrist .These motions are produced by contraction of five muscles. Therefore ,if the EMG signal of muscle is recognized then the specified motion of this muscle is recognized. The proposed method is outlined in Fig. 9 and the stages of proposed system are discussed below:

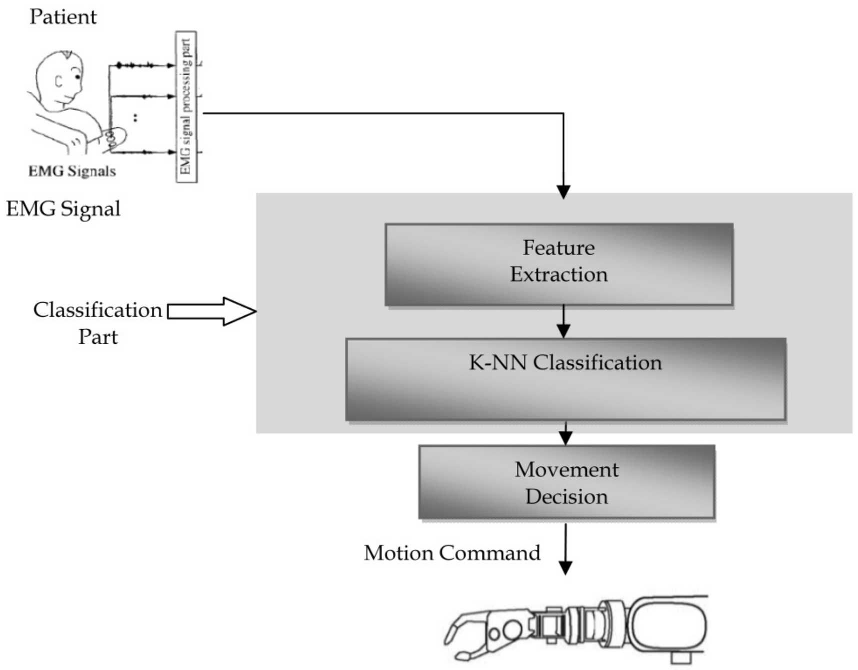


Figure 9. Structure of the Recognition system based on K-NN classifier

10.1. Data bases building with multiple feature parameters

Training of the system involves the partitioning of the feature space to represent different classes of separable motions. In this state, data base is constrained with five muscles (biceps, triceps, deltoid, extensor carpi radialis and flexor carpi ulnaris). This stage has the following steps:

Step 1. Take six frames from each muscle as shown in Fig. 10 to produce thirty frames.

Step 2. Six features (which were introduced in section (4)) are extracted from each frame. In this step thirty vectors are billed as a basic Data Base, each vector has six elements. See Fig.11.

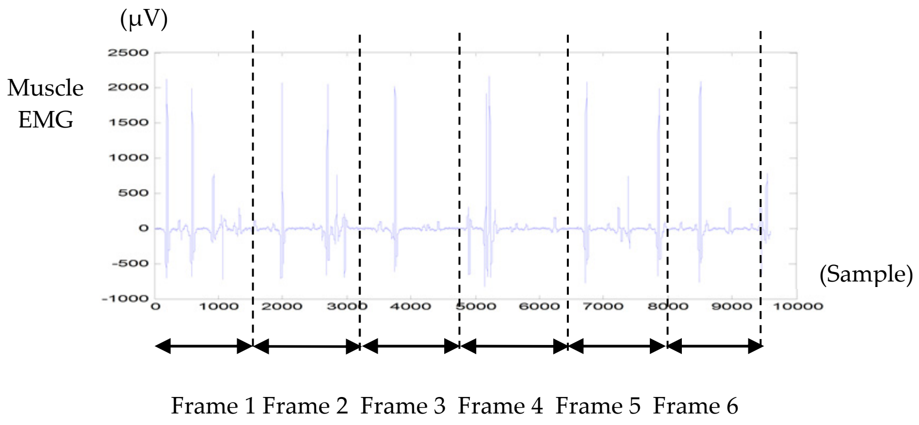


Figure 10. EMG Signal frames.

$$DB_{ij}=\{IEMG_{ij}\ MAV_{ij}\ MMAV_{ij}\ VAR_{ij}\ WAMP_{ij}\ WL_{ij}\ \} \quad (5)$$

$i=1,\ldots,30\ \ ,j=1,\ldots,6$

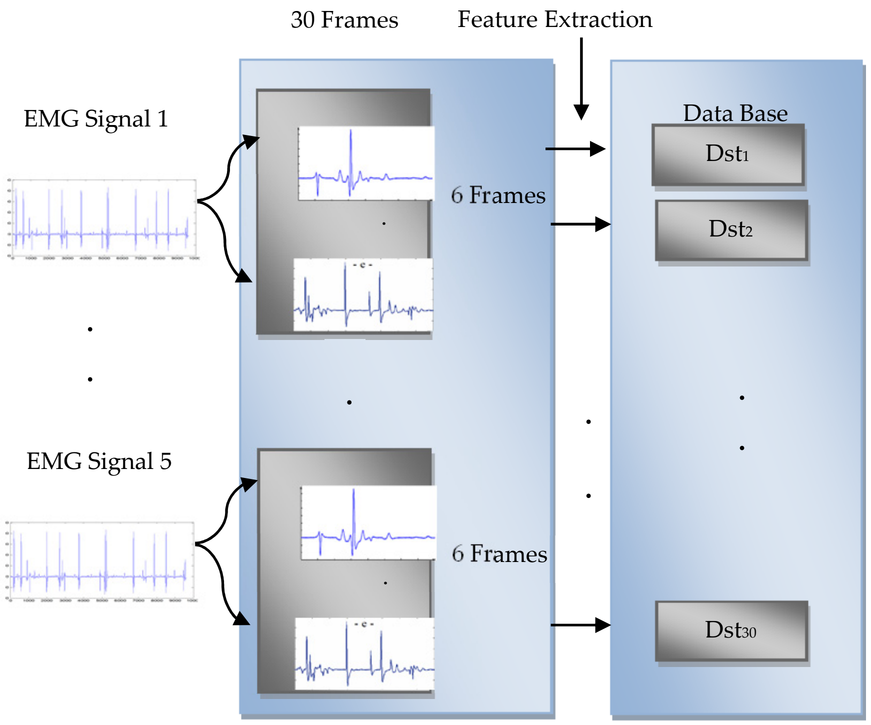


Figure 11. Building Data Base from EMG signals

10.2. Recognition of EMG signals based on k-nearest neighbor algorithm

The data for EMG are generated by EMG simulator then processing will be applied to this signal such as feature extraction.

The recognition system of EMG patterns consists of the following steps:

Step 3. In this step take the feature extraction of input signal to produce Input Feature (IF) which has six elements. Now, there is one vector and it is required to be classified to which type of EMG signal included with thirty frame found in data base, step 2-6 represent K-NN algorithm as shown in Fig. 12.

Step 4. Take Euclidean distance between IF and DB to produce set of distance elements (Dst_1, Dst_2,…, Dst_{30})

$$Dst_i =\{ Dist(DBi,IF) \;\; i=1,2,\ldots,30\} \tag{6}$$

Step 5. Take the nearest neighbors to IF by sorting the distance elements ascendant to produce sorted elements (S_1,S_2,…,S_{30}), where S_1 is nearest element to IF.

Step 6. Take the first K^th elements from sorted elements (S_1,S_2,…,S_K).

Step 7. Assign sorted elements to its original frames of EMG signals.

Step 8. Apply Majority Rule (Which EMG signals has largest number of neighbors from input signal in the K-Nearest Neighbors frames?). The input signal is assigned the most common class of EMG signal among its k nearest neighbors.

As illustrated above, it is necessary to find the distances between the query and all training samples and the closeness is defined in terms of Euclidean distance. Each muscle have specified motion therefore after recognize the EMG signal the command is sent to prosthetic arm to perform the motion of this muscle. The performance index for the Recognition Accuracy (RA) is given by (Momen et al.,2008):

$$RA = \frac{co}{tn} x100 \tag{7}$$

Where: co is the number of correctly classified EMG signals.

tn is the total number of EMG signals.

Case study:

The simulated data are generated from an EMG signal simulator. Several motions are recognized based on classification of five input EMG signals. In the present study, the accuracy for each participant is simply calculated by averaging the performance indices over all movements. To simulate real noise, different noise is considered, i.e. random noise. The noise is added to EMG signals which produce new EMG signal with lower SNR. Now some noised EMG signals are classified using Artificial Neural Network (ANN) and k-Nearest Neighbor algorithm.

Table 3 gives the comparison results of the K-NN with different K values with the back propagation neural network (BP-NN). The structure of BP-NN is (6-20-5). Where the input nodes equal to the number of the features and output nods equal to the number of classified EMG signals.

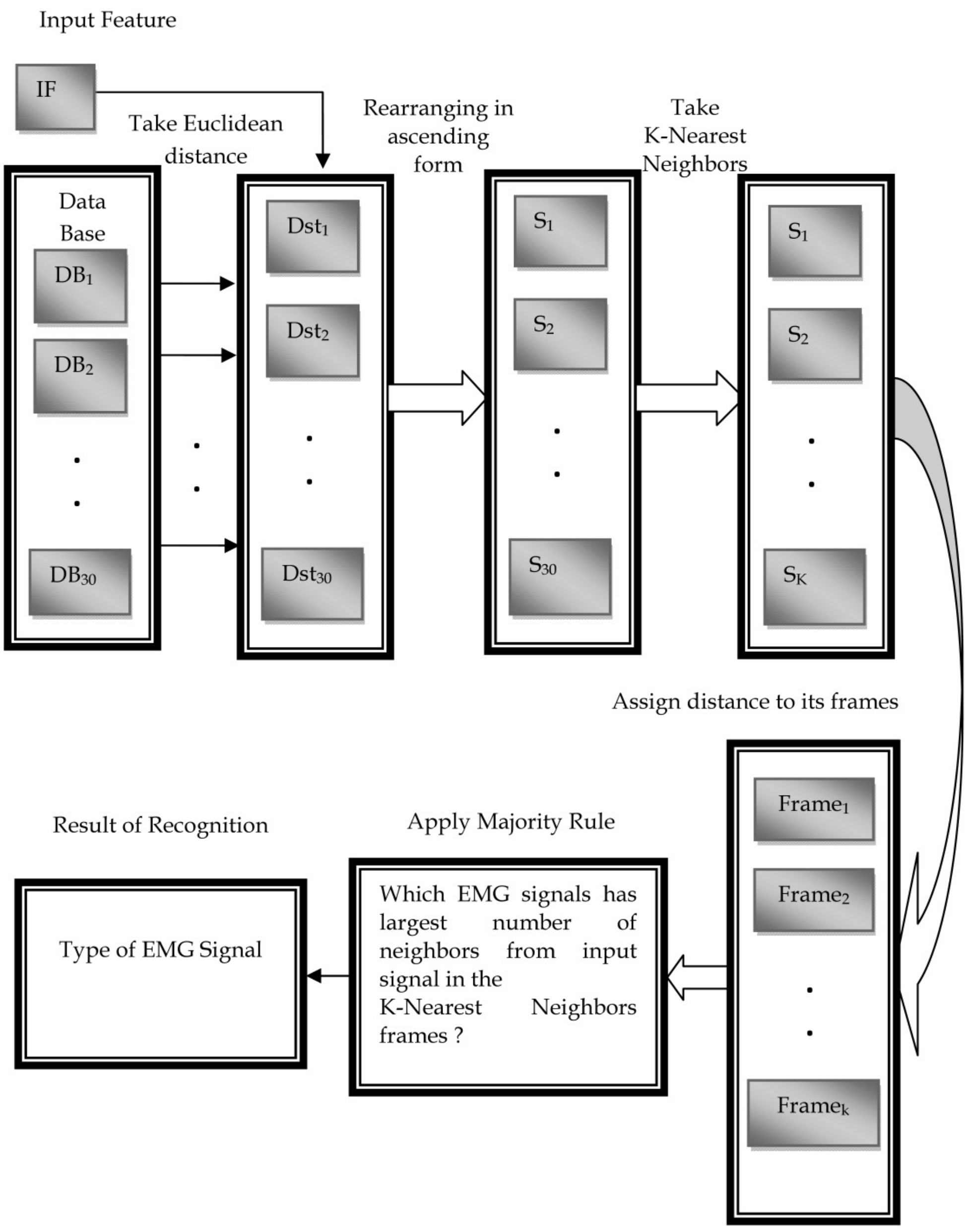


Figure 12. K-NN Classification Process

SNR (dB)	ANN	K-NN K=7	K-NN K=9	K-NN K=13	K-NN K=15	K-NN K=17
25	83%	100%	100%	100%	100%	100%
11	66%	100%	100%	100%	100%	100%
9	66%	100%	100%	100%	100%	100%
7	66%	80%	83%	88%	90%	83%
5	50%	70%	73%	80%	83%	80%

Table 3. Recognition Accuracy of Neural Network and K-NN Method with Noisy Signal.

The MATLAB program is given below:

```
%%%%%%% main program of EMG recognition
clear
clc
load mscl2
%---------------------
%  test signal
%--------------------
 interval=1600;
ths=50;v=120;
DB=build_data_base;
%%%%%%%%%% chose second EMG signal as test signal with added noise
x=mscl2(1,:);
xold=x;
x = x'+v*randn(size((x')));
r = SNR(x, xold');
 %---------------------
%  featur extraction
%--------------------
 mx=mean(abs(x));
vx=var(abs(x));
mavx=mean(abs(x))/interval;
w=.5*ones(1,interval);w(1,.25*interval:.75*interval)=1;
mmavt1=mean((w.*abs(x'))')/interval;
wwamp=WAMPsingle(x',ths);
wl=Wlsingle(x');
IF=[mx vx mavx mmavt1 wwamp wl];
```

```
 %---------------------
% apply k-nn
%--------------------
Dst=dist(IF,DB);
nn=6;
[a ,b]=sort(Dst);
st=floor(b/nn)+1;
kk=15;
Frame=st(1:kk)';
x=ones(length(Frame),1);
a1= Frame ==1;
b1=sum(x(a1));
a2= Frame ==2;
b2=sum(x(a2));
a3= Frame ==3;
b3=sum(x(a3));
a4= Frame ==4;
b4=sum(x(a4));
a5= Frame ==5;
b5=sum(x(a5));
bb=[b1 b2 b3 b4 b5];
[a ,b]=sort(bb);
af=fliplr(a);bf=fliplr(b);
result=bf(1)
```

11. Proposed model and vr simulation for artificial human arm

The simulator was built using MATLAB with Virtual Reality Toolbox. MATLAB provides powerful engineering tool including frequently used mathematical functions. It is easy to implement control algorithm including visualization of data used in the algorithm. In addition, by using Virtual Reality Toolbox, it is convenient to treat 3D objects defined with Virtual Reality Modeling Language (VRML). Thus, it is possible to build a simulator within a relatively short period. virtual reality is a system which allows one or more users to move and react in a computer generated environment. The basic VR systems allow the user to visual information using computer screens. The simulation contain two part ,first ,building model for human arm in VRML, second, call and run the model of human arm using virtual reality toolbox in the MATLAB.

To realize the VRML model for Human arm save the file as HumanArm.wrl file, which is the file format for Virtual Reality software, the VRML model of the human arm is designed in V-Realm Builder 2.0 . Fig.13 presents the VRML model of the human arm.

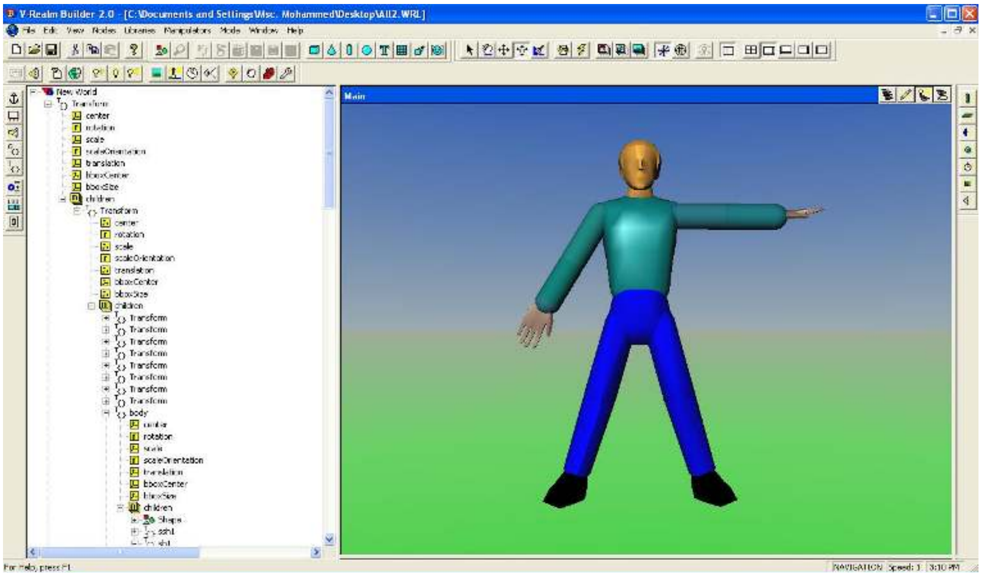

V-Realm Builder 2.0 - [C:\Documents and Settings\Msc. Mohammed\Desktop\All2.WRL]
File Edit View Nodes Libraries Manipulators Mode Window Help
New World
Transform
center
rotation
scale
scaleOrientation
translation
bboxCenter
bboxSize
children
body
Main
For Help, press F1
NAVIGATION Speed: 1

Figure 13. The VRML model of the Human arm

The MATLAB program is given as:

```
%% control of human arm as MATLAB code
world=vrworld('HumanArm.wrl', 'new');
open(world);
fig=vrfigure(world);
elbow=vrnode(world, 'elbow');
k=0;
for theta=pi/2:-pi/100:0
  k=k+1;
  x(k)=theta;
  pause(0.05)
  elbow.rotation=[0 0 theta theta];
  vrdrawnow;
end
```

12. EMG simulator based GUI

The method of recognition based K-NN algorithm and technique of virtual reality are introduced, now the simulation which connect between the EMG recognition system and virtual reality is presented.

From MATLAB, the user can set the positions and properties of VRML objects, create callbacks from Graphical User Interfaces (GUI), and map data to virtual objects. The user can also view the world with a VRML viewer, determine its structure, and assign new values to all available nodes and their fields. The Virtual Reality Toolbox includes functions for retrieving and changing the virtual world properties and for saving the VRML files corresponding to the actual structure of a virtual world. MATLAB provides communication for control and manipulation of virtual reality objects using MATLAB objects.

This part concerns with the simulation of human arm movement using EMG Signal as shown in the fig 14 .

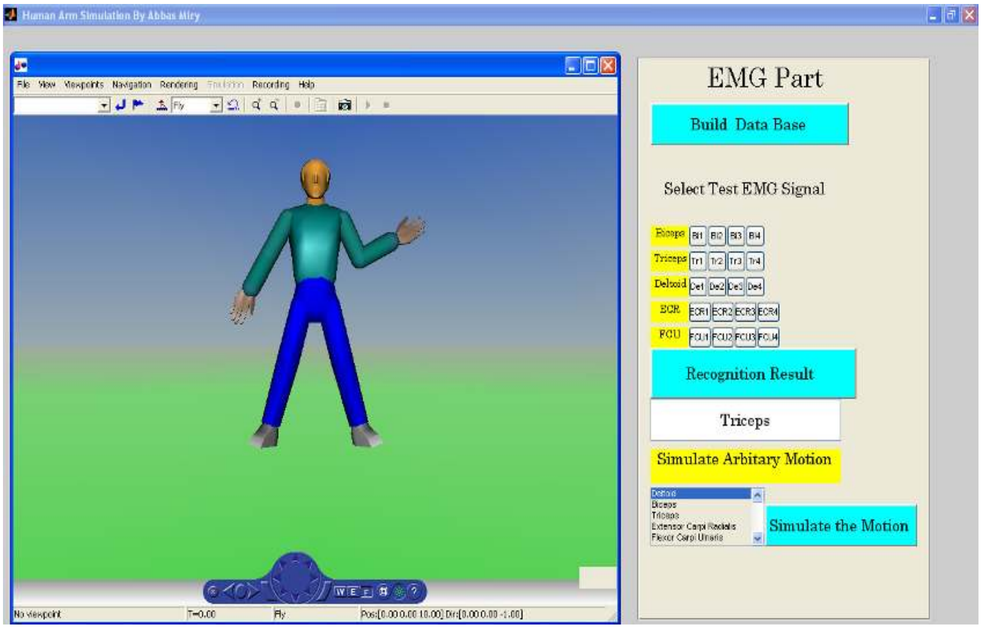


Figure 14. Recognition Result Windows

It has the following stages:

6. **Build Data Base**: This stage represents the training state in which data base is constrained with five muscles (biceps, triceps, deltoid, extensor Carpi radialis and flexor carpi ulnaris). More details found in subsection (10.1).
7. **Select Test EMG Signal**: Select the type of EMG signal to recognize with added noise to test the equality of proposed method with different noise level. Five muscle are tested (biceps,triceps, deltoid, extensor carpi radialis and flexor carpi ulnaris)are tested with four level of noise to produce four noised EMG signals for each muscle.

8. **Recognition Result:** In this stat the K-NN algorithm is applied to recognize the EMG signal in order to produce the motion provided by this EMG signal in virtual reality. The details of this stage can be found in subsection (10.2).
9. **Simulate Arbitrary Motion:** In this part of simulation any motion can be chosen by selecting the muscles which produce this motion.
10. **Simulate the Motion:** After pressing on this button the human arm is moves in the virtual reality.

The recognition algorithm consists of following steps. Firstly, a nearest neighbor algorithm is applied to compute the distance between the feature extraction of input motion to be recognized and each of the 30 feature extraction representing the recognizable motions, which were collected in a Data Base. The algorithm then sorts the motion indexes starting from the nearest candidate in descending order. The distance between two patterns is computed in the feature extraction as the Euclidean distance between the two vectors of feature extraction.

The basic motions types are executed to test the model's performance: a Abduction of the arm, flexion the elbow, extension the elbow, extension and abduction of the wrist and flexes and adducts the wrist. For all motions, the system will receive an EMG signal of human arm.The MATLAB program, which executes this algorithm, is given as:

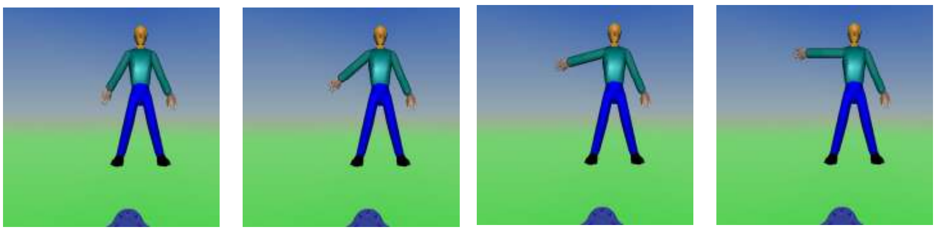
```
function EMGGUI(hObject,eventdata)
 fh2 = figure('MenuBar','none','Name',' Artificial Human Arm Driven by EMG Signal
','NumberTitle','off','Position',[0 0 1300 800]);
 v1=10;
v2=20;
v3=30;
v4=200;
ph = uipanel('Parent',fh2,'Position',[.65 .05 .3 .9]);
e1 = uicontrol(fh2,'style','text','string','EMG Part','Position',[875 625 250
75],'FontName','Century Schoolbook','FontSize',24);
database = uicontrol(fh2,'style','pushbutton','string','Build Data Base
','BackgroundColor','c','Position', [850 600 260 50],'callback',@Data_Base,'FontName','Century
Schoolbook','FontSize',14);
 e2 = uicontrol(fh2,'style','text','string','Select Test EMG Signal ','Position',[850 500 250
60],'FontName','Century Schoolbook','FontSize',14);
 XX7=480;
e3 = uicontrol(fh2,'style','text','string','Biceps','Position',  [850 XX7 50
25],'BackgroundColor','Y','FontName','Century Schoolbook','FontSize',10);
e4 = uicontrol(fh2,'style','pushbutton','string','Bi1','Position', [900 XX7 25
25],'callback',@Biceps1);
e5 = uicontrol(fh2,'style','pushbutton','string','Bi2','Position', [925 XX7 25
25],'callback',@Biceps2);
```

```
e6 = uicontrol(fh2,'style','pushbutton','string','Bi3','Position', [950 XX7 25
25],'callback',@Biceps3);
e7 = uicontrol(fh2,'style','pushbutton','string','Bi4','Position', [975 XX7 25
25],'callback',@Biceps4);
 XX1=XX7-30;
e8 = uicontrol(fh2,'style','text','string','Triceps','Position',  [850 XX1 50
25],'BackgroundColor','Y','FontName','Century Schoolbook','FontSize',10);
e9 = uicontrol(fh2,'style','pushbutton','string','Tr1','Position', [900 XX1 25
25],'callback',@Triceps1);
e10 = uicontrol(fh2,'style','pushbutton','string','Tr2','Position', [925 XX1 25
25],'callback',@Triceps2);
e11 = uicontrol(fh2,'style','pushbutton','string','Tr3','Position', [950 XX1 25
25],'callback',@Triceps3);
e12 = uicontrol(fh2,'style','pushbutton','string','Tr4','Position', [975 XX1 25
25],'callback',@Triceps4);
 XX2=XX1-30;
 e13 = uicontrol(fh2,'style','text','string','Deltoid','Position', [850 XX2 50
25],'BackgroundColor','Y','FontName','Century Schoolbook','FontSize',10);
e14 = uicontrol(fh2,'style','pushbutton','string','De1','Position', [900 XX2 25
25],'callback',@Deltoid1);
e15 = uicontrol(fh2,'style','pushbutton','string','De2','Position', [925 XX2 25
25],'callback',@Deltoid2);
e16 = uicontrol(fh2,'style','pushbutton','string','De3','Position', [950 XX2 25
25],'callback',@Deltoid3);
e17 = uicontrol(fh2,'style','pushbutton','string','De4','Position', [975 XX2 25
25],'callback',@Deltoid4);
 XX3=XX2-30;
 e18 = uicontrol(fh2,'style','text','string','ECR','Position',    [850 XX3 50
25],'BackgroundColor','Y','FontName','Century Schoolbook','FontSize',10);
e19 = uicontrol(fh2,'style','pushbutton','string','ECR1','Position', [900 XX3 30
25],'callback',@ECR1);
e20 = uicontrol(fh2,'style','pushbutton','string','ECR2','Position', [930 XX3 30
25],'callback',@ECR2);
e21 = uicontrol(fh2,'style','pushbutton','string','ECR3','Position', [960 XX3 30
25],'callback',@ECR3);
e22 = uicontrol(fh2,'style','pushbutton','string','ECR4','Position', [990 XX3 30
25],'callback',@ECR4);
 XX4=XX3-30;
 e23 = uicontrol(fh2,'style','text','string','FCU','Position',    [850 XX4 50
25],'BackgroundColor','Y','FontName','Century Schoolbook','FontSize',10);
e24 = uicontrol(fh2,'style','pushbutton','string','FCU1','Position', [900 XX4 30
25],'callback',@FCU1);
```

```
e25 = uicontrol(fh2,'style','pushbutton','string','FCU2','Position', [930 XX4 30
25],'callback',@FCU2);
e26 = uicontrol(fh2,'style','pushbutton','string','FCU3','Position', [960 XX4 30
25],'callback',@FCU3);
e27 = uicontrol(fh2,'style','pushbutton','string','FCU4','Position', [990 XX4 30
25],'callback',@FCU4);
 result1 = uicontrol(fh2,'style','pushbutton','string','Recognition Result
','BackgroundColor','c','callback',@test,'Position',[850 300 270 60],'FontName','Century
Schoolbook','FontSize',14);
result2 = uicontrol(fh2,'style','edit','Position',[850 250 250 50],'FontName','Century
Schoolbook','FontSize',14,'BackgroundColor','W');
e28 = uicontrol(fh2,'style','text','string','Simulate Arbitary Motion
','BackgroundColor','y','Position',[850 200 250 40],'FontName','Century
Schoolbook','FontSize',14);
e29 = uicontrol(fh2,'Style','listbox','String',{'Deltoid','Biceps','Triceps','Extensor Carpi
Radialis','Flexor Carpi Ulnaris'},'Max',5,'Min',0,'Value',[1],'Position',[850 125 150 70]);
e30 = uicontrol(fh2,'style','pushbutton','string','Simulate the Motion
','callback',@simulat_motion,'BackgroundColor','c','Position',[1000 125 200
50],'FontName','Century Schoolbook','FontSize',14);
 end
```

After execution of this program, its results will be given as in Fig. 14 .

As an examples the progress of the Abduction of the arm can be observed in Fig. 15.



(0 sec) (0.1 sec) (0.3 sec) (0.5 sec)

Figure 15. Abduction of the arm movements

The execution of the elbow flexion can be observed in Fig. 16 along with several of the arm states. At the initial position, the human arm is in extension. When flexion begins, the human arm opposition angle is controlled to the right position for the elbow flexion. After receiving the elbow flexion signal, the structure of the arm causes its joints to flex in a natural motion

The block diagram of the package for the human arm simulator is shown in Fig. 17. It enables testing of different control algorithms. The simulator uses MATLAB as this language both provides a virtual reality toolbox and an extensive mathematical library.

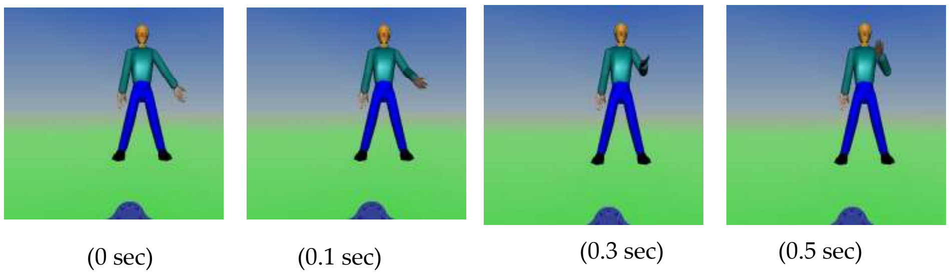


Figure 16. Flexion of the elbow movements.

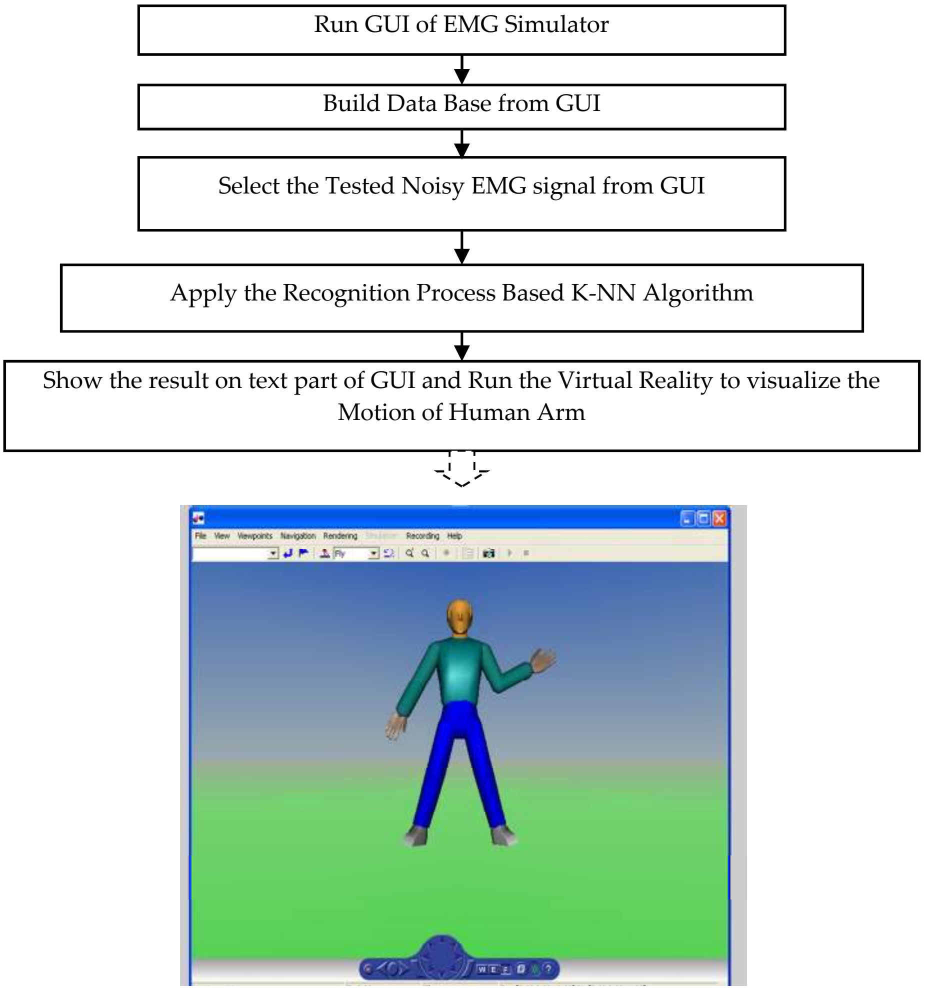

Run GUI of EMG Simulator
Build Data Base from GUI
Select the Tested Noisy EMG signal from GUI
Apply the Recognition Process Based K-NN Algorithm
Show the result on text part of GUI and Run the Virtual Reality to visualize the Motion of Human Arm
File View Viewpoints Navigation Rendering Simulation Recording Help
Fly

Figure 17. Block diagram for Human Arm Movement based EMG signal

13. Conclusions

In this chapter the motion classification simulations are carried out, in order to evaluate classification performance of the human arm movements recognition based on k-Nearest Neighbor algorithm. The simulated data were generated from an EMG signal simulator. Several motions are recognized based on classification of five input EMG signals. In the present study, the accuracy for each participant was simply calculated by averaging the performance indices over all movements.

The results illustrate that the recognition using K-NN presents better results than artificial neural network in term of recognition accuracy as shown Table 2. This table shows the result of recognition with noisy signal having lower SNR for the neural network and different values of k. It can be found that the K-NN method with the value of k=15 achieves better performance than neural network method and K-NN with the other values of k. The reason of successful of K-NN algorithm that, the input signal may be similar to other frames of EMG signal due to the effect of the noise, therefore ,it is necessary to check the input EMG signal with more frames of EMG signal to give good recognition result .

The possible reason for the poor results of ANN may be due to the simple decision function realized by this method. EMG signal has variation with time therefore necessary to check the input signal with more frame for each muscle as shown in K-NN which provides more accuracy in the recognition. The choice multiple features provide more information about the input signal, failer one of these features can be repaired by the other features.

This chapter also presents the simulation of human arm motion in virtual reality to test the algorithm of EMG recognition. It can be concluded that, The Virtual Reality is useful to test the viability of designs before the implementation phase on a virtual reality prototype. It found that, MATLAB a convenient platform for development of computational algorithms, and with the visualization functions of MATLAB Ver.R2009a a reasonable amount of visualization techniques are available.

Author details

Mohammed Z. Al-Faiz
Computer Engineering Department, Al-Nahrain University, Baghdad, Iraq

Abbas H. Miry
Electrical Engineering Department, AL-Mustansiriyah University, Baghdad, Iraq

14. References

Bawaneh, M.; Alkoffash, M. & Rabea, A.(2000). Arabic Text Classification using K-NN and Naive Bayes, *Journal of Computer Science*. ISSN 1549-3636, Vol.4 No.7, pp: 600-605.

Beau C. (2005), Real-Time Classification of Electromyographic Signals for Robotic Control, *Technical Report No. 2005-03-05, Department of Computer Science, University of Washington*

Bu, N.; Okamoto, M. & Tsuji, T. (2009).A Hybrid Motion Classification Approach for EMG-Based Human–Robot Interfaces Using Bayesian and Neural Networks, *IEEE Trans. on Robot*. Vol. 23, No. 3, pp. 502–511.

Elliott, R. (1998).Feature Extraction Techniques for Grasp Classification ", *Master Thesis, University of Canterbury*.

EMGLAB software Version 0.9 User's Guide, "The MathWorks, at www.mathworks.com, 2008.

Hamilton, A. & Stashuk, D. W. (2005).Physiologically based simulation of clinical EMG signals, *IEEE Trans. Biomed. Eng*., Vol. 52, No. 2, PP:171-183, 2005.

Huang, H.; Liu, Y. & Wong, C.(2003).Automatic EMG Feature Evaluation for Controlling a Prosthetic Hand Using a Supervised Feature Mining Method :An Intelligent Approach, *International Conference on Robotics & Automation, Taiwan, IEEE*, pp:220-225, 2003.

Kiguchi, K.; Watanabe, S.; Izumi, K. & Fukuda, T. (2001).An Exoskeletal Robot for Human Elbow Motion Support—Sensor Fusion, Adaptation, and Control, *IEEE Trans on System, Man and Cybernetics*, Vol. 31, No.3, pp:353-361.

Momen K.; Krishnan, S. & Chau, T. (2007). Real-Time Classification of Forearm Electromyographic Signals Corresponding to User-Selected Intentional Movements for Multifunction Prosthesis Control, *IEEE Trans on Neural System and Rehabilitation Engineering*, Vol. 15, No. 4, pp:535-542.

Moradian, M. & Baraani, A.(2009). KNNBA: K-Nearest-Neighbor-Based-Association Algorithm. *Journal of Theoretical and Applied Information Technology*, Vol.6. No. 1, pp 123 – 129.

Pal, N. & Ghosh, S. (2001).Some Classification Algorithms Integrating Dempster–Shafer Theory of Evidence with the Rank Nearest Neighbor Rules, *IEEE Trans on Systems, Man, and Cybernetics—Part A*, Vol. 31, No. 1, pp: 59-66.

Park, S. & Lee, S. (1998). EMG Pattern Recognition Based on Artificial Intelligence Techniques, *IEEE Trans on Rehabilitation Engineering*, Vol. 6, No. 4, pp:400-405.

Parvin, H; Alizadeh, H. & Bidgoli, B. (2008). MKNN: Modified K-Nearest Neighbor, *Proceedings of the World Congress on Engineering and Computer Science*, pp:831-834.

Perry, J.; Bekey, G. (1981).EMG-force relationships in skeletal muscle." *CRC Critical Rev. in Biomed. Eng.*, pp. 1-22.

Phinyomark, A.; Limsakul, C. & Phukpattaranont, P.(2009). A Novel Feature Extraction for Robust EMG Pattern Recognition, *Journal of Computing*, ISSN: 2151-9617, Vol.1, No. 1, pp:71-80.