State Detection from Electromyographic Signals towards the Control of Prosthetic Limbs

Pamela A. Hardaker, Benjamin N. Passow and David Elizondo
De Montfort University’s Interdisciplinary Group in Intelligent Transport Systems (DIGITS) & Centre for Computational Intelligence (CCI), De Montfort University
Leicester, LE1 9BH, United Kingdom
pamela.hardaker@ieee.org, benpassow@ieee.org, elizondo@dmu.ac.uk

Abstract — This paper presents experiments in the use of an Electromyographic sensor to determine whether a person is standing, walking or running. The output of the sensor was captured and processed in a variety of different ways to extract those features that were seen to be changing as the movement state of the person changed. Experiments were carried out by adjusting the parameters used for the collection of the features. These extracted features where then passed to a set of Artificial Neural Networks trained to recognise each state. This methodology exhibits an accuracy needed to control a prosthetic leg.

Index Terms — Artificial Neural Network, Electromyographic Sensor, Feature Extraction, Pattern Recognition, Prosthesis

I. INTRODUCTION

The development of effective and functional prostheses has become increasingly important. This is in part due to the rise in injuries caused by the number of conflicts in the world and also due to the improvements in medical science meaning that people are more likely to survive limb threatening injuries [1].

There are many problems experienced by those trying to produce prostheses to emulate a real leg which are slowly being resolved. However, one particular problem still persists, the way in which the unit reacts to changes in the movement state of the wearer. Generally this change in state requires a manual command from the wearer either through manipulation of the prosthetic or through a remote control [2], [3].

The purpose of this research was to examine ways in which this area could be improved and the use of an electromyographic (EMG) sensor on the surface of the prosthesis wearer’s skin was explored. Signals from the EMG sensor were recorded while a subject was standing still, walking and running. These recorded signals were then examined to extract a series of features including the average, the maximum signal height, the width of the last cluster of signals and the gap between the last two clusters. Various artificial neural networks (ANNs) were then trained to recognise each state.

This paper is divided into five sections. A literature review is presented in section two, the methodology is presented in section three and experimental results are given in section four. Final conclusions are drawn in section five.

II. LITERATURE REVIEW

There are two main ways in which computational intelligence can be used for prostheses control. The first is in interpreting signals from the wearer to determine what movement they are making or are about to make. These could be done by using pressure or movement sensors within the prostheses [2], [3] or EMG sensors on the surface of the skin. The second is in controlling the actions of the limb and there is overlap between the development of prostheses and of robots and exoskeletons. In both cases the artificial limbs need to move and balance like a human limb to be successful. This area of control can be split into two. The first being simple pattern recognition to determine the current state of the prosthesis from the incoming signals. This could be a simple command to open or close a hand or it could be a more complex output controlling individual fingers. The second is in the control of the entire walking gait of e.g. a bipedal robot.

A. Gathering EMG signals

Signals from an EMG sensor have previously been used to control a prosthetic arm as early as 1984 [4]. Research has been done into the fact that the residual limb can still give the signals required to control the missing part of the limb and the challenges of doing this are discussed in [5]. One of the main challenges is that each person and each type of step produces different signals at different times making pattern recognition difficult. Another challenge is that the signals that reach the surface mounted EMG sensor have to pass through other muscles, tissue and bone, all of which will have an effect upon the signal, and this is discussed in [6].

Another consideration is how much of the residual limb and muscle remains and also how far this muscle has atrophied. Often, amputees have to work hard to build these muscles back up so that the residual limb becomes useful again. In some cases this development has gone even further and [5] describes work that has been done with upper limb amputees to transfer the nerves previously used to control the amputated limb to other muscles so that they can be used to stimulate the prosthesis. Based on this, it is suggested that this technique could also be used for lower limb amputees.

In [7] multiple sensors were used over the main six muscles of the thigh. While this gave excellent results it is unlikely that mounting this number of sensors regularly would be practical.
However, Cheron et al process the data from these sensors to give relevant angles for each part of an entire limb rather than creating a usable prosthesis that will function in real-time.

As well as the control of prostheses, the possibility of using EMG signals for teleoperation has previously been considered [8], [9]. This allows a human to control a remote limb where the environment in which the limb needs to operate would be hostile to a human. It is suggested that using EMG sensors for control is more intuitive and less fatiguing than traditional manual control methods such as joysticks. Another form of remote operation is discussed in [10] where the authors have used EMG signals to directly control an exoskeleton. This has potential for disabled people with paralysis of the limbs.

B. Feature Extraction

The information that comes from the EMG sensor, shown in Figure 1, is in the form of a signal that an ANN would not easily be able to recognise. It is therefore beneficial to extract relevant features that distinguish one movement state from another. In [11] this is described as the “main kernel of classification systems and it is essential to the motion command identification”. Park and Lee highlight that it is difficult for one feature to reflect the overall state of the signal and so several different features are required.

This research looks at using a variety of features collected over a number of samples and the relevance of the size of this number is discussed. Two factors that need to be taken into account when choosing this number are the sampling frequency and the number of movements being made. Another technique used to extract features is filtering [6]. Most papers agree that the extraction of data from the signal in a timely fashion is one of the hardest aspects of the process.

C. Pattern Recognition

Once the features have been extracted from the EMG signal they need to be used to distinguish between the different states. This process involves looking at the patterns within the signal features and then training a computational intelligence system to recognise and separate these patterns. This can be done by using e.g. fuzzy mapping functions [11] or a variety of ANNs [6], [9] and [12].

Finding a relationship between the features and the different states or patterns can be achieved in a variety of ways. In [13], it is discussed that the amplitude, magnitude and intensity of the EMG signal has a relationship with the force and position of the limb. However, rather than using this data for real-time control of a limb, the authors use it to predict how the user is walking and to simulate the walking gate from the EMG signals through an ANN.

D. Controlling a limb or prosthesis

Computational Intelligence has been used to control prosthetic limbs in a variety of ways. In [7] the outputs from the EMG are passed through a dynamic recurrent neural network (DRNN) to control all three sections of a virtual limb on screen. The outputs from the DRNN are in the form of angular velocities for the hip, knee and ankle joints.

A true control system is discussed in [14] where the authors have studied the intricacies of walking. They use a feedforward neural network to overcome the limitations of rule based control systems which are unable to take into account changing demands and terrain. This produces a simulation of a prosthesis which is very effective but not real-time. The authors of [15] look at using a real-time neural network to control a biped walking robot. This uses a cerebellar model arithmetic computer (CMAC) neural network to control the walking gait of the robot using sensors built into the legs to give feedback.

The Plymouth Hand [6], [12] and [16] project has extensively researched the use of an EMG signal to control a prosthetic hand. The authors describe how a single EMG signal is gathered, passed through a series of filters and then fed into a neural network which has been trained to recognise a series of positions for the hand.

The Plymouth Hand [6], [12] and [16] project has extensively researched the use of an EMG signal to control a prosthetic hand. The authors describe how a single EMG signal is gathered, passed through a series of filters and then fed into a neural network which has been trained to recognise a series of positions for the hand.

III. METHODOLOGY

An EMG sensor can pick up the signals sent by the nerves through the body to control the muscles. The EMG sensor used for this research consists of:

![Figure 1 – A sample trace from the sensor showing the output during standing, walking and running](Image)
A Motion Lab Systems MA 317 A300 A3 Preamplifier
An amplifier circuit to further increase signal levels
A Data Translation DT9801 Multifunction USB Data Acquisition Module
Data Translation DT Chartrecorder software

The performance of EMG based pattern classification can deteriorate due to inevitable disturbances to the sensor interface. Therefore, it is critical to locate, within the upper part of the leg muscle of a trans-femoral amputee, the positions which provide the best EMG signals. This was one of the key aims of this research.

Positioning the sensor proved to be quite challenging. It was found that the sensor needs to be held on securely and in precisely the correct position. A trainee doctor was consulted and following reference to two texts, [17] and [18], he advised trialling the sites shown in Figure 2.

![Figure 2 – Sites trialled for the location of the sensor and the final successful and recommended location](image)

- Rectus femoris
- Vastus lateralis
- Vastus medialis
- Quadriceps tendon
- Successful Location

A. Signal Capture

The setup for the DT9801 Data Acquisition Module allows for many changes to be made to the parameters including the units, offset, range and pre-set triggers. However, for the purposes of this research only the sampling rate was altered between 200, 500 and 1000 Hz.

Figures 3, 4 and 5 show the Sensor output in different states. As can be seen in Figure 3, the signal oscillated between plus and minus 9v when there was no contact with the leg. Figure 4 shows the Sensor output when connected to the leg but without any movement. Finally, Figure 5 shows the Sensor output when movement was detected.

B. Extraction of Features

The raw signal from the sensor was recorded and then passed through a simple filter whereby the last three absolute signals were multiplied together and averaged. The resulting output was examined to determine what features were changing in each state. The extraction of these features was trialled with the original signal, the absolute signal and the spikes of the signal above a certain height. The extracted features are outlined below.

- **Maximum sample (MAX)** - The maximum sample over the last SS_Max samples was acquired where SS_Max is the buffer size used for this calculation.
Average of samples (AVG) - The average of the signal was calculated by summing all the samples over the last SS_Avg samples and placing them in a buffer called BF_Avg.

Cluster Width (CLUSW) - While observing the differences between the samples taken during walking and running it became clear that the width of the cluster of spikes above a certain height, SZ_Spike, was changing. This feature is found by using CT_ClusW to count the time between the first and last spike of a cluster. A Spike is deemed to be the last one in a cluster if a certain number of samples, BF_ClusW, have passed. This means that CLUSW has to be calculated retrospectively. CLUSW is limited to a value of MX_ClusW and normalised using NM_ClusW.

Cluster Gap (CLUSG) - This feature is determined by calculating the time between the last spike of the last cluster and the first spike of the next one within a certain number of samples. The Feature Extraction Parameters BF_ClusG, MX_ClusG and NM_ClusG were used in the same way as for CLUSW.

Table 1 - Typical feature extraction parameters used for different sampling frequencies

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Sampling Frequency 1000</th>
<th>500</th>
<th>200</th>
<th>100</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample_set</td>
<td>2000 1000 400 200</td>
<td></td>
<td></td>
<td></td>
<td>Maximum sample set</td>
</tr>
<tr>
<td>SS_Max</td>
<td>800 400 160 80</td>
<td></td>
<td></td>
<td></td>
<td>Sample set for MAX</td>
</tr>
<tr>
<td>SS_Avg</td>
<td>800 400 160 80</td>
<td></td>
<td></td>
<td></td>
<td>Sample set for AVG</td>
</tr>
<tr>
<td>BF_Avg</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td>Buffer to collect Average samples</td>
</tr>
<tr>
<td>NM_Avg</td>
<td>7 7 7 7</td>
<td></td>
<td></td>
<td></td>
<td>Normaliser for Average</td>
</tr>
<tr>
<td>MX_SpikG</td>
<td>40 20 8 4</td>
<td></td>
<td></td>
<td></td>
<td>The max value a spike gap can reach</td>
</tr>
<tr>
<td>MN_SpikG</td>
<td>10 5 2 1</td>
<td></td>
<td></td>
<td></td>
<td>The min value a spike gap can reach</td>
</tr>
<tr>
<td>SZ_spike</td>
<td>2.5 2.5 2.5 2.5</td>
<td></td>
<td></td>
<td></td>
<td>Height of the spike</td>
</tr>
<tr>
<td>NM_SpikG</td>
<td>9 9 9 9</td>
<td></td>
<td></td>
<td></td>
<td>Normaliser for SPIKG</td>
</tr>
<tr>
<td>BF_ClusW</td>
<td>300 150 60 30</td>
<td></td>
<td></td>
<td></td>
<td>Required Spike Gap for CLUSW</td>
</tr>
<tr>
<td>MX_ClusW</td>
<td>300 150 60 30</td>
<td></td>
<td></td>
<td></td>
<td>The Max value for a Cluster Width</td>
</tr>
<tr>
<td>NM_ClusW</td>
<td>9 9 9 9</td>
<td></td>
<td></td>
<td></td>
<td>Normaliser for CLUSW</td>
</tr>
<tr>
<td>BF_ClusG</td>
<td>500 250 100 50</td>
<td></td>
<td></td>
<td></td>
<td>Gap between spikes for CLUSG</td>
</tr>
<tr>
<td>MX_ClusG</td>
<td>1000 500 200 100</td>
<td></td>
<td></td>
<td></td>
<td>The Max value for a Cluster Gap</td>
</tr>
<tr>
<td>NM_ClusG</td>
<td>9 9 9 9</td>
<td></td>
<td></td>
<td></td>
<td>Normaliser for CLUSG</td>
</tr>
</tbody>
</table>

Figure 6 – This diagram shows the method used to calculate each feature and in particular shows how the buffers BF_ClusG and BF_ClusW are used to determine the start and end points of the features CLUSG and CLUSW. Abbreviations of these techniques are explained in more detail in Section II.B.

Figure 7 – Sample EMG absolute output data showing the positions of the extracted spikes and traces for the four successful features of Maximum, Average, Cluster Width and Cluster Gap.
A number of parameters were used for each extracted feature and these are shown in Table 1. This table also shows typical values for each parameter at each sampling frequency. The way in which the feature extraction parameters were used to calculate the features is shown in Figure 6 and the resulting features are shown, superimposed on the signal in Figure 7.

The captured EMG signals and the extracted features where combined into a single text file along with the correct “movement state” value. The “movement states” were 0 for standing, 1 for walking and 2 for running.

C. Artificial Neural Network for State Detection

In order to process these signals with an ANN it was necessary to add a “movement state” to each sample taken. The states used were 0 for standing still, 1 for walking and 2 for running. The movement state was added to the signal by simply recording the required state for a set period of time and then adding the state marking by eye afterwards.

This is a preliminary work which aims at having a sliding scale of outputs to cater for different levels of running/walking not just discrete states. Therefore an ANN was used to distinguish between the movement states based on the extracted features. Other techniques, including Bayes could be used in future work.

Two ANN implementation methods were used, the first used bespoke C++ code to create the ANN and process the data, the second used the MatLab artificial neural network toolkit for further investigation.

For this preliminary work, feedforward backpropagation ANNs with one or two hidden layers have been used. Hidden nodes used hyperbolic tangent sigmoid transfer functions, and the output nodes employed linear transfer functions. The Levenberg-Marquardt backpropagation learning [19] was used to train the ANNs. All feature vectors were tested individually and as a group and were tested on ANNs containing between one and five hidden nodes in one or two hidden layers thus giving 5 x 5 x 5 ANNs.

In addition, experiments were also carried out by passing the recorded signals through a series of ANNs in Matlab. A feedforward backpropagation network was used with the default transfer function of tansig. The training function was gradient descent with momentum and adaptive learning rate backpropagation (traindx), the weight/bias learning function was gradient descent with momentum weight and bias learning function (learngdm) and the performance function was the mean squared normalized error performance function (mse).

Experiments were carried out with different combinations of input and output neurons, extracted features and movement states. The input and output combinations trialled included 10,5, 2,3, 7,5 and 7,3.

The extracted features trialled consisted of various combinations of MAX, AVG, CLUSW, CLUSG and SPIKEG. These features were trialled in various combinations of 2’s, 3’s and 4’s.

Attempts were made to differentiate between the movement states of standing, walking and running in various combinations.

IV. RESULTS

The introduced methodologies have been tested on data gathered from a subject. In direct and lengthy comparison, the most successful site for the EMG sensor was the Quadriceps Tendon. It was felt that this was a reasonably realistic position as it could be within the socket of the prosthesis where the stump of the residual limb sits. Figure 1 shows a set of typical signals captured from the sensor at 1 kHz. Three different states have been recorded, as shown. In the standing state a regular signal of approximately ±2v can be seen when the muscle is at rest. Then, as each step is taken, each movement of the muscle causes a sharp increase in amplitude to a peak followed by a slower decrease in amplitude as can be seen in the walking and running states.

A. Feature Extraction

Figure 7 shows a composite of all the results with the absolute signal shown in continuous grey and the spike in continuous black. The features were extracted using both Excel and C++ code. The Excel method was good for rapid modelling and producing quick graphs. However, the C++ method made it much easier to handle values and buffers and provided an output to be directly applied.

In this work, the following four feature selection methods have been studied and compared in detail:

Maximum sample (MAX) - Different values of SS_Max produced different results and with SS_Max set to half the sampling rate the maximum sample generally increases as the state shifts from walking to running, as can be seen in the black dash and dot trace of Figure 7.

Average of samples (AVG) - Experiments were carried out using an average of all the positive signals and of the absolute signals and for some of these values a normalising value, NM_Avg had to be used. Experiments were carried out using different values of SS_Avg. Also the sampling rate was taken into account. Values between 0.5 and 1.5 for the sampling rate were found to be effective. The effect of averaging at 0.5 of the sampling rate is shown in the dotted black trace of Figure 7. As can be seen, the value gradually increases as the samples get closer together and are higher in amplitude.

Cluster Width (CLUSW) - As can be seen in the grey dash and dot trace of Figure 7, the results for this feature are not conclusive. The difference between standing and the other two states is obvious. However, the difference between walking and running is not quite so clear.

Cluster Gap (CLUSG) - The difference between the three states is quite clear for this feature, as shown in the grey dashed trace of Figure 7. As each cluster represents the muscle movement for one step, CLUSG is really a measurement of the gap between steps. A rough experiment on a treadmill showed that while the number of steps per second changes for walking at different speeds, it is almost exactly the same when running
at different speeds. This is to be expected as most runners extend their stride to speed up rather than increasing the number of steps [20].

B. Accuracy of artificial neural network in state detection

As can be seen in Figure 7, when the four most successful features are captured using optimal feature extraction parameters and placed together, they start to show common tendencies in each of the three states which suggest that they may be suitable for use with an ANN.

Once the different features to be extracted were determined experiments were conducted with different sampling rates. In general, a higher sampling rate tended to give better results and Figure 7 shows the AVG feature at 100Hz (the black trace) and 500 Hz (the grey trace). As can be seen the 100 Hz trace is far more erratic and changing the extraction parameters did not improve this much. However, when the same comparison has been done for CLUSG the time improvement of the higher sampling rate is not as obvious.

The Matlab implemented feedforward backpropagation neural network outlined in Section III.C was trialled first and produced one successful result. It used a two layer network with five input neurons and two output neurons and the four extraction features previously outlined. However, it only managed to distinguish between the walking and running movement states.

Figure 8 shows the results of the training data, which looks very promising. However, Figure 9 shows the results of using a new set of test data on the successful network, which is not as good. It is worth noting that the set of data used to train and the one used to test were taken at different times when the sensor had been removed and repositioned and this difference in signals is an issue that will need to be addressed.

The ranking plots in Figure 10 show the results of the C++ implemented ANNs with different learning and transfer functions. On the left are the results from the training data and on the right the test data. Each dot represents one trialled ANN with the cross representing the mean of the set of tests. From the training data test results it appears that using all four features produces the best mean results followed by AVG, CLUSW and then CLUSG. While the test data results also show all four features to provide the best mean results, here CLUSW performs better than AVG. The best all feature vector ANN, as measured on test data, was able to correctly identify the walking state 95.8% of the time. The best AVG and CLUSW based ANNs were able to achieve over 96% accuracy. Although it is difficult to directly compare, these results are relatively high compared to other research such as the work by Huang et al [21] where results are reported to range between 70% and 90%.

It is not surprising that the successful ANN consisted of roughly the same number of input and output neurons as there were inputs and outputs. The chosen features mean that the traces for walking and running were quite distinctly different and so extra neurons would not be necessary to choose between the two patterns. What was surprising is that the ANN was not able to distinguish between standing and walking. Further experiments may lead to success in this area but the trace for this movement state was so distinct that better results were expected.

It was also not possible to distinguish between running at two different speeds. This may, in part, be due to the fact that CLUSG does not significantly change at different running speeds.
V. CONCLUSIONS

In this paper we have presented the possibility of using the outputs from an EMG sensor on the surface of the skin of a leg to determine the wearer’s movement state. A suitable site was found on the leg for the EMG sensor and signals from the sensor were then recorded. A number of features were extracted from the recorded signals using a variety of extraction feature parameters. These features were then passed through ANNs to train them to distinguish between standing, walking and running states.

The features extracted have yielded successful neural networks which can distinguish between the different walking states. While the best AVG and CLUSW feature vectors based ANNs showed excellent accuracy of over 96%, the best mean performance over all test ANNs was achieved by using all feature vectors.

The success of these experiments shows that the ANN has the ability to learn how to distinguish between different movement states based on the extracted features. In a real life situation the ANN would be trained for each individual wearer and would therefore learn their own walking and running styles. It would then be possible to retrain the ANN as the wearer’s ability improved or changed. It is also hoped that it will be able to learn a variety of different other states and the transition between one state and another such as sitting to standing and stair climbing.

The speed with which the trained ANN can then make decisions about the current movement state make it ideal for use in controlling a micro-processor controlled limb in real time. This means that the output from the EMG sensor can provide the required sensory data to control a microprocessor controlled prosthetic limb. This would mean that a wearer of such a prosthetic limb would be able to move around more naturally and change the movement state of the limb without the need for a remote control.

The main aim of this research was to explore techniques to best detect the walking and running states of the user including finding the best locations for placing the EMG sensors and extracting useful features from the captured signals. Further work is planned with larger quantities of data collected on a treadmill and from more volunteers. It is also planned to develop a method of dynamically recording the movement state rather than manually determining it. This may further strengthen the accuracy and reliability of the proposed methods. Eventually these techniques will be tested directly on a completely self-sufficient prosthetic limb.

ACKNOWLEDGMENTS

The authors would like to acknowledge the assistance of Thomas Stubington, the trainee doctor who provided assistance in finding the relevant sites for the sensor.

REFERENCES


