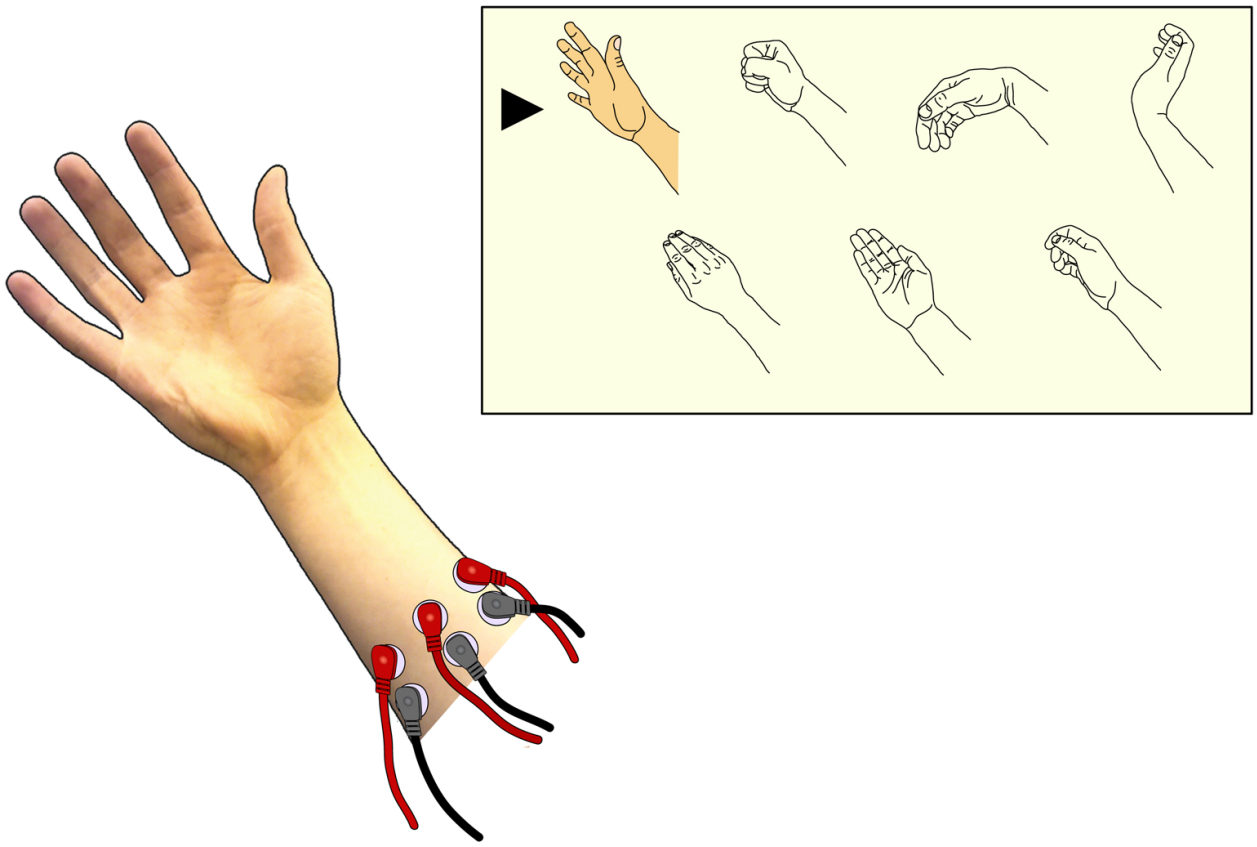
CHALMERS



Evaluation of Post-Processing Strategies for Simultaneous Pattern Recognition Based Myoelectric Prosthetic Control

Master of Science Thesis in Complex Adaptive Systems

JOEL FALK-DAHLIN

Department of Signals and Systems
Division of Biomedical Engineering and Processing
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2013
Report No. EX010/2013

REPORT NO. EX010/2013

Evaluation of Post-Processing Strategies for Simultaneous Pattern Recognition Based Myoelectric Prosthetic Control

JOEL FALK-DAHLIN

Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2013

Evaluation of Post-Processing Strategies for Simultaneous Pattern Recognition Based Myoelectric Prosthetic Control
JOEL FALK-DAHLIN

Department of Signals and Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
Myoelectric control of six movements involving the hand and wrist. Myoelectric control systems use electrical signals generated by muscle contraction to identify which of the available functions in a prosthesis that the user intends to activate. Illustrated by Joel Falk-Dahlin.

Department of Signals and Systems
Gothenburg, Sweden 2013

Evaluation of Post-Processing Strategies for Simultaneous Pattern Recognition Based Myoelectric Prosthetic Control

JOEL FALK-DAHLIN
Department of Signals and Systems
Chalmers University of Technology

Abstract

Myoelectric control is a field that has shown great potential when it comes to controlling advanced prosthetic limbs. However, due the reduced accuracy of movement classification during real time operation different post-processing algorithms have been proposed. Recent development extends the myoelectric control concept to allow multiple movements to be controlled simultaneously, but the post-processing algorithms has not yet followed the development.

This work examines the effects of using two post-processing algorithms, Majority Vote and Decision-Based Velocity Ramp, together with the simultaneous control approach. Both algorithms are well known in the field of myoelectric control and were both initially suggested for the control of individual movements. This work also studies the errors made by the classifier in a system for simultaneous movement control, and suggests modifications to the two post-processing algorithms based on the findings.

Experiments showed that a modified version of the Majority Vote algorithm could produce outputs of the same quality as the system without any post-processing applied, in spite of each prediction being based on half the amount of data. The result suggests that Majority Vote can be used to reduce the computational cost of the outputs, which might be necessary when implementing the control in a real prosthetic device.

The Decision-Based Velocity Ramp was found to improve the path efficiency for all types of movements compared to the system without post-processing applied. Movements involving all three available degrees of freedom were however performed slower when the Velocity Ramp was applied, suggesting that further modification of the algorithm is needed.

In this work it is concluded that the Majority Vote and the Decision-Based Velocity Ramp algorithms have to be modified in order to function optimally with the simultaneous control approach. The two algorithms has previously been considered as competitors, but this work suggests that they could be used as complements to each other.

Keywords: myoelectric, prosthesis, simultaneous, post-processing, velocity ramp, majority vote.

Acknowledgments

I would like to thank my supervisor Max Ortiz Catalan for the invaluable help he has provided during the work of this thesis. The work has been based on his research and the BioPatRec software which he has created. Without Max this thesis would not have been possible.

Further I would like to thank Bo Håkansson who has examined my work, and Integrum AB for offering a desk and a great working environment. Most of the work has been carried out at Integrum AB, which has been a very pleasant experience. I would like to give a special thanks to Nichlas Sander who has made major contributions to the BioPatRec software, developing the foundation of the target achievement test that has been used for this work.

Finally I would like to thank my loving family, especially my girlfriend, for supporting me throughout this work.

Contents

List of Figures

List of Tables

Abbreviations

AR	Auto regressive
Chalmers	Chalmers University of Technology
COO	Centre of Orthopaedic Osseointegration
DoF	Degrees of freedom
LDA	Linear discriminant analysis
MES	Myoelectric signals
MLP	Multilayer perceptron
RMSE	Root mean square error
SUH	Sahlgrenska University Hospital
SVM	Support vector machine
TAC-test	Target achievement test
TD	Time domain
TDAR	Time domain and auto regressive

Introduction

Multi functional prostheses are being developed with the main goal of improving the quality of every day life for amputees. These prostheses are, in contrast to cosmetic prostheses, intended to restore some or all of the functions that the amputee has lost. This is done by giving the amputee the possibility of manipulating the prosthesis, making tasks such as grasping an object possible. Functional prostheses can be powered either externally or by the body itself, both approaches having their advantages. Battery driven prostheses allows the user to perform tasks without a considerable amount of extra effort while body powered prostheses (e.g. through wire tension) gives reliable devices that do not require charging. Externally powered prostheses are the more sophisticated of the two and could ultimately restore all the lost functions to the amputee using modern electronics. Motors can move the prostheses in multiple degrees of freedom (DoF) and sensors can provide the user with different feedback. The idea behind externally powered prostheses is not a new one, one of the first documented is a pneumatic hand that was patented shortly after World War I (Childress, 1985). Society has since made some great technological advances, and the new technology has opened the door for even more sophisticated prostheses.

A multi-functional prosthesis requires that the different functions are easily controllable by the user. Myoelectric control is a field that aims to make the control of these prostheses as intuitive and natural as possible. The approach uses electrical signals that are produced when muscles are contracted as inputs to the prosthesis. The signals can be measured from remnant muscles in the stump, making it possible for the user to move the prosthesis by trying to move the amputated limb. Some of the modern myoelectric systems use pattern recognition algorithms to classify the signals. This approach increases both the functionality and the intuitiveness of the control as it allows the user to easily and directly switch between multiple functions. Functions can be mapped according to what patterns are actually observed in the signals when the user is trying to activate a known function, making the control as intuitive as possible. An artificial hand could for instance be taught to open when the system recognizes that the user is trying to open the amputated hand. Most of the research on pattern recognition based myoelectric control have considered the control of individual functions. To cope with the problem that the patterns are misclassified from time to time, different post-processing algorithms have been developed. These algorithms uses various techniques to minimize the impact of misclassified patterns during real-time operation of the prosthesis.

New myoelectric control systems have started to investigate the possibility of controlling multiple functions at the same time. These systems allow the user to perform movements such as the opening and rotation of a hand simultaneously, much like the movements of an intact limb. The success of these systems would be a huge step towards a natural control of artificial limbs. A new system, currently developed at Chalmers University of Technology (Chalmers) in close collaboration with the Centre of Orthopaedic Osseointegration (COO) at the Department of Orthopaedics at the Sahlgrenska University Hospital (SUH) and Integrum AB (Ortiz-Catalan et al., 2013*a*), has achieved simultaneous control of multiple movements by extending the pattern recognition approach to identify combinations of the individual movements as well (Ortiz-Catalan et al., 2013*c*). The problem with patterns being misclassified from time to time however still remains, and the problem may be even greater with the simultaneous approach as there exists more outputs that the system should recognize. No post-processing algorithm has yet been developed specifically to work with simultaneous control, and there is no guarantee that algorithms developed to work with individual control can be used, as the two approaches are very different in nature. There clearly exists a need to understand how the misclassifications are affecting the simultaneous control approach and how to design post-processing algorithms to improve the real-time prostheses controllability.

1.1 Purpose of Thesis

The purpose of this thesis is to improve the pattern recognition based myoelectric control systems that are designed to work with simultaneous movements. These systems apply pattern recognition algorithms to the electromyogram to recognize which movements that are desired by the user. Classification of the patterns are however not 100% correct and patterns are from time to time misclassified. Pattern-recognition based system were initially developed for the control of individual movements and those systems has developed different post-processing algorithms that reduces the effect of the misclassifications during real-time operation. However, there is currently no post-processing algorithm specifically designed to work with simultaneous control, and using one might increase the controllability significantly. Additionally, there is no guarantee that the algorithms developed for individual control will work as the two approaches are considerably different. In this thesis, the possibilities and the effects of bringing post-processing algorithms from systems controlling individual movement to the simultaneous movement control system developed at Chalmers, COO and Integrum AB (Ortiz-Catalan et al., 2013*a*) will be examined.

1.2 Limitations

The performance of a myoelectric control system depends on multiple factors. The electromyogram can be recorded from multiple sites, different filters can used to precondition the signals, different signal features can characterize the electromyogram, different pattern recognition algorithms can be used to determine the intention et cetera. This work aims to evaluate the use of post-processing algorithms, and for that reason only a specific setup of a pattern recognition based myoelectric control system were studied. The system that was used is developed at Chalmers, COO and Integrum AB and was set up based on previous work by Ortiz-Catalan et al. (Ortiz-Catalan et al., 2013*a*,*c*).

The problem studied is the control of an artificial hand that can move simultaneously in three DoFs. Movements involve both the hand and wrist and can be viewed in section 3.1. The electromyogram is collected from eight sites around the circumference of the forearm using an in-house designed amplifier. Signals are characterized using the time domain (TD) feature set which consists of four features; mean absolute value, waveform length, number of zero crossings and number of slope sign changes. The signals are windowed using 200 ms analysis windows applied every 50 ms. A single multilayer perceptron trained with back propagation makes the real-time classifications. More details on the system setup is given in section 3.2. All post-processing algorithm were taken from systems designed to work with individual control and were modified when necessary.

Background theory

This thesis will focus on the use of post-processing algorithms to improve the real time control of a powered arm prosthesis that is able to move in three DoFs, involving the hand and wrist. This is a specific version of the more general prosthesis control problem but the results can be applied to other versions, e.g. the control individual fingers or the control of an artificial leg. This chapter is the result of the literature review that initially was conducted during this thesis. It is aimed to give the necessary background theory to understand what a post-processing algorithm is, and how it is used for prosthetic control.

2.1 Pattern Recognition Based Myoelectric Prosthetic Control Systems

The combination of pattern recognition algorithms together with the myoelectrical signals (MES) has shown promising results as a solution to the prosthetic control problem (Scheme and Englehart, 2011). These systems use the myoelectric signals generated when muscles are contracted to understand what the user intends to do. The pattern recognition algorithms are applied to make the translation between the electrical signals and the users intentions. These algorithms do not normally operate directly on the recorded signal, but on a set of features that are calculated to characterize the signal. The selection of signal features plays an important part in the final performance of the control system. The features has to contain enough information to make it possible to distinguish between all desired user intention, be stable over time and be easy to compute to allow the system to operate fast enough.

A lot of different feature sets have been proposed and tested for pattern recognition based myoelectric control but no set is generally accepted as the best one. A good feature set should cluster features originating from different intentions to make it possible to distinguish between them. Feature clustering in feature space is illustrated in figure 2.1, where features from two different movements are easily distinguish from each other.

The pattern recognition based myoelectric control systems are described using a model of the prosthesis control problem that was introduced by Fougner et al. (2012). The model, presented in appendix A, divides the control problem into three major categories; preprocessing, intent interpretation and output. Preprocessing involves recording and conditioning, windowing and feature extraction of the input data. Intent interpretation is the problem of identifying what the user intents to do, given the signal features. For pattern recognition based systems, this problem is solved using pattern recognition algorithms. Output is the problem of controlling the actual device once the intention has been determined.

2.1.1 Preprocessing

Preprocessing is the first category of problems in the model introduced by Fougner et al. (2012). These problems deal with the collection, conditioning and feature extraction from the inputs. The model describes the general prosthesis control problem, to which the myoelectric approach is a solution. In these systems the preprocessing involves recording, conditioning, windowing and feature extraction of the MES.

Recording the MES

Recordings of the MES can be done either on the surface of the skin, extra- or intra muscular. Depending on how many movements the system should be able to recognize and the nature of those movements, different number of electrode channels and different electrode placements may be required. The impact of

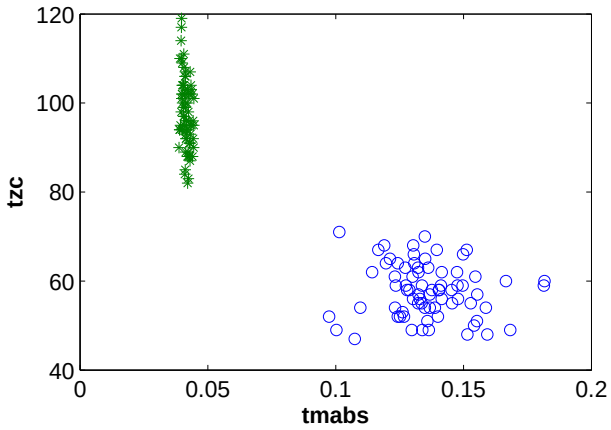


Figure 2.1: A scatter plot of two features calculated from the MES. The MES were recorded from a single electrode channel while the subject performed two movements, 'Open Hand' represented by circles and 'Close Hand' represented by stars. The calculated features are the mean absolute value (tmabs) and the number of zero crossings (tzc), both from the time-domain. Features originating from the different movements cluster in the feature space which makes them distinguishable. Pattern recognition algorithms automatically finds a boundary that separates the clusters, which they later use to predict the intentions when they are presented with new signal features.

choosing either surface or intra muscular measurement of the MES was studied by Hargrove et al. (2007). The results showed that the surface electrodes had much higher crosstalk between the channels than the intra muscular, but both of them could still be used to maintain a high offline classification accuracy ($> 95\%$) of ten different arm movements. However, many of the problems that myoelectric prosthetics currently are facing are due to the usage of surface electrodes (Ortiz-Catalan et al., 2012). Implantable electrodes might be necessary for the development of a robust long-term solution, even though the same offline accuracy can be achieved from surface recordings.

To investigate the optimal number of surface electrode channels, the same report (Hargrove et al., 2007) used data from a subset of the channels to calculate the classification accuracy. The channels were selected according to two different schemes, one that optimizes the performance, and one where the channels are naively kept equidistant around the forearm. For their particular version of the myoelectric control problem, involving ten individual movements of the hand and wrist, they saw that only three surface electrode channels were needed if they were placed optimally, and four if they were naively placed, to achieve an offline classification accuracy of 97% on average. However, it is known that offline accuracy does not directly translate into a high real-time performance (Ortiz-Catalan et al., 2013*a*).

Windowing

Pattern-recognition based myoelectric control builds on the idea that features from the MES can be grouped together into classes. Due to the stochastic nature of the MES, single samples do not provide enough information to distinguish between different movements, and more advanced signal characteristics than the instantaneous magnitude and sign have to be calculated. The calculation of such signal features, e.g. the waveform length or the mean absolute value, requires that the signal is divided into smaller portions, windows. The signal features calculated from a time window can then be used to predict which movement the signal is originating from. Usually several channels are used to record the MES and thus each channel has to be windowed. There exists two types of windowing schemes, overlapped or disjoint in time, meaning that the windows either share or do not share a portion of the MES. A windowing scheme can be represented by two numbers, the window length and the time increment, see figure 2.2. The window length is the amount of time-data that is analyzed when calculating the chosen signal features, and the time increment is the time that passes before applying a new analysis window.

It has been shown that if the analysis windows are chosen too short, the variance of the signal features will be too high for the classifier to be able to make good predictions. If they are chosen too long the user have to wait before the movements are outputted, making the prosthesis uncontrollable (Englehart and Hudgins, 2003). There exists a trade-off between having a good classification accuracy and a responsive

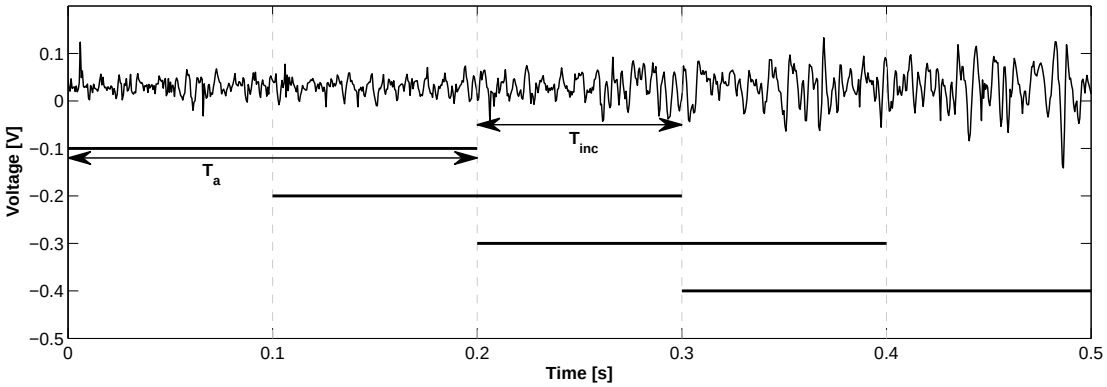


Figure 2.2: Windowing scheme of a time signal. The figure shows an overlapped windowing scheme of a time signal using an analysis window of length (T_a) 200 ms and a time increment (T_{inc}) of 100 ms. The thick lines represents different analysis windows, and the portion of the signal that lies above the line is said to lie in that window. The time increment is the time that passes before applying a new analysis window to the signal, here shown as the time between the end of two windows which is the same. If the time increment is equal to or larger than the analysis window length, no overlapping of the thick lines will occur and the windows are said to be disjoint in time. The portion of the signal that lies within an analysis window is the portion of the signal that is used to extract features from in pattern recognition based control.

controller that has to be considered when building these systems.

Smith et al. (2011) studied the compromise between classification accuracy and controller delay by letting 13 able-bodied subjects perform the TAC-test described in section 2.4.1 using different windowing schemes. The window lengths varied between 50 and 550 ms and the time increment was fixed to 25 ms during the experiment. The system were trained to recognize seven different movement of hand and wrist, and the results from the experiment showed that the optimal time window lie between $150 - 250$ ms.

Feature Set

The feature set is the collection of characteristic measurements, signal features, that are chosen to represent the MES. The feature set has to be chosen in a way that they separate points in feature space that originate from different movements, to make sure that they can be distinguished. If there is no separation between the points, the task of classification becomes impossible. Different feature sets can be used in combination with different pattern recognition algorithms to make the classification, and there exists no universal solution on how to combine the two. Definitions of the most commonly used features for myoelectric control are given by Micera et al. (2010), where features from the time domain, time-serial domain, frequency domain and time-scale/time-frequency domains have been collected. These features include everything from mean absolute value and auto regressive models to short-time Fourier and wavelet transformations. Other features such as the Hjorth Time Domain Parameters has also been suggested for myoelectric control (Mouzé-Amady and Horwat, 1996), which makes the choice of feature set even harder. The features are of varying complexity, some being computationally heavy and others very simple, which might influence the choice when designing a myoelectric control system.

A comparison between three different feature sets were made by Hargrove et al. (2007). They compared the TD, auto regressive (AR) and a concatenated version (TDAR) feature sets using two different pattern recognition algorithms. They found that the performance difference between the feature sets were very small but the concatenated version performed best. Yet, all this work has been done in with offline classifications. The effects different feature sets have on real-time and simultaneous predictions are still unknown.

2.1.2 Intent Interpretation

After input recordings, filtering and feature extraction is done, and the information is used to interpretation the users intention. In pattern recognition based myoelectric control the intention is interpreted using pattern recognition algorithms. There exist a lot of algorithms that can be used to make the

classification of the signal features. Common for most of the algorithms is that they require some training on a prerecorded data set. The algorithms are presented with both the inputs and the correct outputs to help it determine the decision boarders, lines that separate the different feature clusters. Once the decision boarders has been determined by the algorithm, it can use the boarders to classify the recorded MES.

Pattern Recognition Algorithms

There exists many algorithms that detect patterns in the input data, and uses that information to classify new data. Some of the most common algorithms are linear discriminant analysis (LDA) , multilayer perceptrons (MLP) and support vector machines (SVM) . These algorithms have different intrinsic properties, e.g. as the name of LDA implies it has a linear nature, trying to find planes in feature space that separates the clusters with maximum distance. Predictions are made with LDA by looking at what side of the plane a new data point lie. The MLP on the other hand is capable of making highly nonlinear decision boundaries, making it possible for the decision boundary to entirely enclose a cluster. All modern classifiers are likely to perform well with offline metrics if they are presented with a good feature set (Scheme and Englehart, 2011). They should however be chosen based on their ability to classify data during real-time operation (Ortiz-Catalan et al., 2013*a*).

Post-processing algorithms

The myoelectric patterns can sometime be misclassified in these systems, i.e. when the pattern recognition algorithm fails to identify the correct intention. Such misclassifications will affect the real time controllability of the prosthesis, as wrong prosthesis functions may be activated. To minimize the impact of these misclassifications during real time operation, these systems use post-processing algorithms. Post-processing algorithms are applied after the predictions have been made by the classifier, using the information to improve the control.

Three post-processing algorithms were found during the literature review, Majority Vote, Bayesian Fusion and Decision-Based Velocity Ramp. All of these algorithms were developed for control systems of individual movements, and none were found that had been developed for simultaneous control. The algorithms take two different approaches to the task. Majority Vote and Bayesian Fusion try to identify misclassified movements and prevent them from being outputted, while the Velocity Ramp makes sure new movements are outputted with low speed.

Post-processing algorithms are the focus of this thesis and they have received extra attention in their own section. Previous important works in the development of these algorithms are presented in chronological order in section 2.2, and a more detailed description of the algorithms are given in section 2.3.

2.1.3 Output

The last problem in the model presented by Fougner et al. (2012) treats the activation and control of the actual prosthetic device. This problem can be solved virtually when no real device exists, letting the outputs be sent to a virtual environment. Another important part of the output layers is the user feedback. In most cases the user can see how the prosthesis behaves, forming a visual feedback to the user. This will however require the user to focus for even the simplest tasks to be performed as there is no other way of knowing what is going on. More sophisticated systems can use other user feedback, such as vibrotactile or electrical muscle stimulation, to deliver information to user. That information could for example be the grasping force or tactile information from the artificial limb. The importance of additional user feedback should not be neglected as the lack of sufficient feedback was identified as one of the main reasons that myoelectric prosthesis are not being used by amputees (Peerdeman et al., 2011).

2.1.4 Known Problems With Pattern Recognition Based Myoelectric Control

Even though the pattern recognition based approach to myoelectric control has shown great potential to bring natural control to artificial limbs, several problems has been reported with these systems. These

problems has to be solved for the real time control of a prosthesis to function properly for everyday use.

Electrode Shift

When surface mounted electrodes are used, some problems arise due to shifting of the electrode placement during usage. The relative position between the electrodes on the skin and the underlying muscles can vary during operation as the muscles are moving. This will make the recorded electromyogram vary to some extent and the effect has been studied by Hargrove et al. (2006). They placed four surface electrodes around the circumference of the forearm and collected the electromyogram while performing 10 different movements. The electrodes were then shifted on the skin in various directions and electromyogram were collected again. The results showed that the classification accuracy dropped from 95% to 80% when the electrodes were shifted one cm distally and all the way down to 60% when rotated one cm. Systems with classification accuracies this low would be uncontrollable as the misclassifications would make the system produce a lot of undesired movements. The report also showed that the effect can be reduced by using data from several electrode positions when training the pattern recognition algorithm, this would however increase the already time consuming training phase and might not be very practical.

Variation in Force

Pattern recognition based systems has been reported to perform differently when trying to recognize the same movement performed at different force levels (Scheme and Englehart, 2011). This indicates that the features that are produced differs between the cases. The effect was briefly studied by Scheme and Englehart (2011) and it was concluded that the force level has a big impact on the classification accuracy. This makes the system less robust if movements are not performed at the correct force levels, since they can be classified as something else. This effect was also shown to be reducible by introducing data from different force levels when training the classifier.

Variation in Limb Position

Classification accuracy has been shown to drop when the prosthesis is operated at different limb positions. This is an issue, as many of the reported results on these systems have been produced in a laboratory environment where the position of the arm has been held fixed. In daily life the arm has to be operated in many different positions and if the accuracy is position dependent, operation might not be possible. The effect has been studied by Fougner et al. (2011) and their results showed that the classification accuracy dropped from 96% to 80% when the limb was operated in other positions than it were trained in. They were able to reduce the effect and achieve an over all classification accuracy of 95% by including data from all limb positions when training the classifier.

Transient Changes in EMG

When a prosthesis is used during extended times, the qualitative features of the EMG will change. This is an effect of different factors such as muscle fatigue, the user adapting to the control et cetera. When surface mounted electrodes are used, this problem further increases as the electrodes can shift their position or lift from the skin. The transient changes of the features will reduce the accuracy of the system over time and at some point the user must recalibrate the system. A recalibration would require the user to undergo a new training phase of the system, performing all the desired movements several times. A way to help the system not having to recalibrate so often is to use adaptive learning algorithms that retrains the classifier during online operation to incorporate the transients changes. Such an algorithm would have to decide by itself when the features of the signal has changed enough for a recalibration to be needed and update the system to recognize the new patterns. Different algorithms proposed as a solution to this problem has been compared by Sensinger et al. (2009). They used an entropy measurement to determine how confident the system was that the predictions were correct and tagged predictions based on their entropy. Different strategies that used the tagged predictions were then proposed and tested in an offline analysis. The results showed that the classification accuracy could be increased using adaptive learning, however the algorithms have not been tested (to the best of my

knowledge) during real time operation. Using adaptive learning algorithms does not guarantee that the system will be following feature trends and manual recalibration may still be needed.

2.2 Previous Work on the use of Post Processing Algorithms

This section describes the most important work in the field of applying post-processing algorithms to myoelectric control. The work is presented in a chronological order so the results easily can be followed.

The earliest work taken into consideration were made by Englehart and Hudgins (2003). They realized that previously suggested myoelectric control systems did not fully utilize their processing power. More predictions could be made by the system if the MES were windowed with an overlapping scheme, applying a new analysis window as soon as previous window was processed. The increased number of predictions made by the system could be subjected to post-processing techniques to further improve the classification accuracy. They suggested an algorithm they called the Majority Vote that took into consideration the current, previous m and next m predictions when making a decision. The movement that was predicted the most number of times within the $2m + 1$ predictions was taken to be the desired. As the algorithm had to wait for the coming m predictions before making a decision, they set up a condition on the number m to make sure that the system produced decisions acceptably fast.

$$m\tau \leq D \tag{2.1}$$

Here τ [s] is the time it takes for the system to process one analysis window and D [s] the acceptable delay before the algorithm makes a decision. Larger analysis windows requires more time to be processed, which makes less predictions available for the algorithm to make a decision within the time D. The results showed that the Majority Vote algorithm could increase the accuracy substantially when analysis windows smaller than 128 ms were used. The improvement when larger analysis windows were used were limited to two or three percent (Englehart and Hudgins, 2003).

There still existed an ambiguity on how long an acceptable delay is for prosthetic control. If the delay is too long the system would be sluggish and hard to operate. A lot of different acceptable delays, ranging from 50 to 400 ms, were used at this time (Farrell and Weir, 2007). To remove this ambiguity Farrell and Weir (2007) built an extremely responsive one DoF prosthesis, with which they could test the impact of the controller delay by adding it virtually. The prosthesis was used in a box and block test where subjects were asked to pick up blocks from one box and put them into another using the prosthesis. The test was conducted by twenty subjects, using two different prehensor speeds, with seven different delays. The results of the experiment showed that the performance was significantly reduced when adding control delays of 150 ms or more. An analysis of the results, using a linear mixed effects model, showed that the delay should not be greater than 100-125 ms for the controller to work optimally for 90 percent of the users. This delay should be used to create an accurate classification of the desired movements as possible.

Farrell and Weir (2008) later showed that the perceived controller delay is not the same as the delay between outputs of the system. The delay between outputs had previously been accepted as the number of predictions the system had to wait before making a decision, see section 2.1. The perceived delay is however a function of the time data that is used when making the decision, including not only the coming m predictions but also past $m + 1$ predictions. Using a lot of old time data to make the decision means that movements the user performed before affects the output now. Farrell and Weir (2008) suggested instead an equation for calculating the average perceived delay. The equation was also confirmed in a simple experiment where subjects switched between two movements.

$$T_D = \frac{1}{2}\left(T_a + nT_{inc}\right) + \tau \tag{2.2}$$

T_D [s] is the average delay perceived by the user before a movement is performed by the prosthesis. T_a is the length of the analysis window [s], T_{inc} the time increment before applying a new time window [s] and τ the processing time [s], i.e. the time it takes for the computer to make a classification given a time window. As the delay depends on all predictions used to make the decision, they used the number $n = 2m + 1$ as the total number of predictions considered when making a decision in their equation.

These new results showed that the length of the analysis window was important to keep the controller delay down. This pinpoints a compromise that has to be made when designing these systems, since the accuracy of the classification also depends of the length of the analysis windows. If the analysis window is chosen too short, the signal features will have high variance making them hard to classify, if chosen too long the perceived controller delay will increase (Englehart and Hudgins, 2003). Smith et al. (2011) performed an experiment to find the optimal analysis window length. They had 13 subjects perform a real-time performance test, using analysis windows ranging from 50 to 550 ms. The results showed an optimal balance between the classification error and controller delay when analysis windows were chosen between 150-250 ms. It should be noted that the results were based on the completion rate of the TAC-test using their particular setup. The delay is not only a function of the window size, but also of the time increment, processing time and the number of predictions considered when making the decisions. Using equation (2.2) with their setup, 150-250 ms T_a, 25 ms T_{inc} and $n = 1$, the perceived delay is found to be roughly between 90-140 ms, which is consistent with the results found on the optimal controller delay.

Later the same year, a new idea on how to reduce the impact of the misclassifications was presented by Simon et al. (2011*a*). They had come up with a new post-processing algorithm they called the Decision-Based Velocity ramp, that in contrast to Majority Vote operates on the output speeds of the movements instead of the predictions themselves. The idea was that by passing all predictions through to the output, the perceived delay could be held at a minimum. By attenuating the output speed of new movements, the impact of spurious misclassifications would still be limited. To see the benefit of using the velocity ramp Simon et al. (2011*a*) performed two experiments, one with a physical prosthesis and one within a virtual environment. The performance improvement was drastic, results from the experiment with the physical prosthesis showed that amputated subjects could stack 89% more 1-inch wooden cubes when using the suggested algorithm. The second experiment compared the Velocity-Ramp and the Majority Vote using the TAC-test (section 2.4.1). The results showed no significant differences between not using majority vote at all, and using majority vote with 3 and 5 votes respectively. When the Velocity Ramp was used, the completion rate increased and users were taking a more direct path to the target.

In an attempt to develop an improved post-processing algorithm that could be used instead of Majority Vote, Khushaba et al. (2012) developed an algorithm they called Bayesian Fusion. By using a disjoint windowing scheme, they assumed that data in different windows would be very weakly or not at all correlated to each other due to the stochastic nature of the MES. This allowed them to approximate the conditional probabilities of the different movements given time data from several analysis windows. They compared the new algorithm against Majority Vote in an experiment where they recorded movements from nine different finger movements using only two electrode channels. The results showed that the Bayesian Fusion outperformed the Majority Vote algorithm, and it performed best when given data from nine consecutive windows. However, this would generate an unacceptable controller delay. Assuming that the delay still can be estimated using (2.2), their setup would result in a delay of approximately 500 ms (100 ms T_a, 100 ms T_{inc}, $n = 9$). To cope with this, Khushaba et al. (2012) employed both a weight matrix, favoring new predictions over older, and a threshold to detect the on/off set of movements. When no movement was detected, the prediction buffer was reset to allow new movements to be outputted faster. This does however only work when movements are transitioned via the resting state, i.e. the user rests between the different movements. The effect of adding these extra features to the algorithm on the real time delay were never studied, making it impossible to say whether the approach actually was beneficial.

All the results presented here have been developed for systems controlling individual movements. The important results show that there exists a trade-off between classification accuracy and controller delay when selecting the analysis window. The perceived controller delay is important for the real-time performance of these systems and should not be mixed with the output delay. Experiments have shown that the perceived delay should not be larger than 125 ms to make sure that the performance is not reduced. As there is no benefit of having a smaller controller delay, the system should make use of these 125 ms to achieve as good classification accuracy as possible. This can be done using e.g. Majority Vote, which showed significant improvement when using analysis windows smaller than 128 ms. The results show also that Majority Vote can keep the performance at a constant level even when the analysis window length is reduced. This could for example help to improve systems with low processing power,

where small analysis windows are necessary to keep the processing time down. The Decision-Based Velocity Ramp has proven to be useful as amputees could stack 89% more cubes using the algorithm. The Bayesian Fusion can smooth the outputs more effectively than the Majority Vote given that more time data can be used. However, this requires that it somehow can be detected when the user wants to change movement to allow the system to operate within the optimal controller delay. These three post-processing algorithms will be presented in more detail in next section.

2.3 Description of Post Processing Algorithms

To deal with the inevitable fact that misclassification will occur in a pattern recognition based control system, a few post-processing control algorithms have been proposed as mentioned in the previous section. Below will follow a more thorough description of the algorithms.

2.3.1 Majority Vote

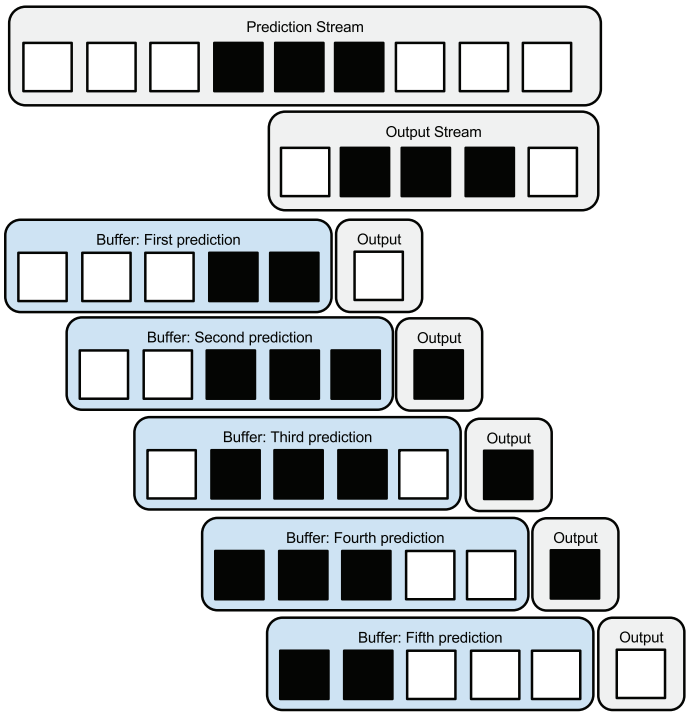


Figure 2.3: The figure shows the inherent delay of the Majority Vote algorithm. The prediction stream includes the predictions made by the classifier, and the output stream the outputs made by the system. Each output is based on several predictions that are stored in a buffer, in this case five predictions are used to produce an output. White squares mark a desired movement and black squares an undesired one. The buffer is illustrated in the figure below the prediction stream to show which predictions that are used to calculate the different outputs. Because the buffer size in this example is too small, all the misclassified predictions are outputted, but at a later time. The same mechanism introduces a delay when switching between different movements, think of black squares being another desired movement.

The Majority Vote algorithm was first proposed by Englehart and Hudgins (2003) and tries to remove spurious errors made by the classifier. This is done by not only looking at the most recent prediction made by the classifier, but at a group of predictions at the same time. The prediction that is occurring most times amongst that group is then considered to be the desired one. Majority Vote has been implemented in different ways, e.g. the original implementation of the algorithm looks at a group of the next m, previous m and current predictions to make the decision. Other implementations has reported to only use the previous m and the current predictions. Both of these approaches are however the same, if the last prediction is thought of as the current one, the same group of predictions as the original implementation can be constructed by using the previous $2m$ and the current predictions. The number

m will therefor not be used from here on, but instead we introduce the buffer size N, the total number of predictions considered when making a decision.

The algorithm is very simple, when a prediction is made by the classifier, put it in the voting group and remove the oldest prediction, keeping the voting group at a constant size N. Count the occurrence of each prediction in the voting group and take the winning prediction as the desired movement. It is easy to see that in order for the Majority Vote to successfully remove the classification errors, the voting group has to contain at least one more vote of the correct movement than the misclassified. This means that to be able to remove n consecutive misclassifications of the same movement, the group needs to be at least of size $N = 2n + 1$. If the group it too small, the misclassifications will not be removed but shifted in time. This is because of an intrinsic delay of the Majority Vote algorithm which is illustrated in figure 2.3. The delay was presented in section 2.2 as Majority Vote increases the perceived delay.

2.3.2 Bayesian Fusion

To further develop the idea behind the majority vote, Khushaba et al. (2012) formulated the Bayesian Fusion control algorithm. They thought the Majority Vote approach to be naive just using the movement with the highest probability from each prediction. The basic principles in this algorithm are however the same as in Majority Vote, data made by the classifier is stored in a buffer and is used to remove classification errors.

Many classifier algorithms give a measurement of how confident the classifier is of its prediction, e.g. the neurons in neural networks uses activation function, often a sigmoid function, mapping all values to lie between zero and one. The output is often taken to be true if this value is above a certain threshold, and false otherwise. The output can therefor be viewed as a measurement of how confident the network is that the prediction is true, the closer this value gets to one, the more probable it is that the output is true. By using these confidence measurements, the Bayesian Fusion attempts to make the predicted movements more accurate. The algorithm is based on Bayes rule, which they used to formulate the probability of movement M_i being desired given data from two consecutive time windows, w_1 and w_2.

$$p(M_i|w_1, w_2) = \frac{p(w_1|M_i, w_2)p(M_i|w_2)}{p(w_1|w_2)} \tag{2.3}$$

The Bayesian approach was motivated by the stochastic nature of the electromyogram, which makes the data in two consecutive disjoint time windows very weakly, if at all, correlated. Since the time windows are uncorrelated, the equation can be simplified using, $p(w_1|M_i, w_2) = p(w_1|M_i)$ and $p(w_1|w_2) = p(w_1)$. The result is an equation describing the conditional probability of movement M_i being desired given two time windows, as the product of the probabilities that M_i is desired in each time window separately.

$$p(M_i|w_1, w_2) = \frac{p(M_i|w_1)p(M_i|w_2)}{p(M_i)} \tag{2.4}$$

The results can then be generalized to contain any number of time windows, resulting in the equation

$$p(M_i|w_1, w_2, ..., w_N) = \Delta \prod_{n=1}^{N} p(M_i|w_n) \tag{2.5}$$

where Δ is a normalization constant making sure that the probability distribution is valid.

A big flaw with this equation is that if the probability of a movement M_i is predicted to be zero in any of the time windows, then $p(M_i|w_1, w_2, ..., w_N)$ also becomes zero. Assuming that switching between two movements happens perfectly, the probability of one movement switches from 1 to 0 and another from 0 to 1. Then no movement will be outputted until the buffer is emptied and all the 0s are replaced. To cope with this problem, a weight matrix is added to the probabilities to make sure that they never become zero. The weight matrix can also be designed to favor new predictions over older ones, to reduce the perceived controller delay. How this actually affects the delay has however not been analyzed.

The weight matrix was suggested to be created with the following formula,

$$k_n = 10 \times \frac{\exp(-0.5j/N)}{\sum_{l=1}^{N} \exp(-0.5l/N)} \tag{2.6}$$

where $n = 1, ..., N$ indexes the time windows, 1 being the most recent and N being the oldest, k_n the weight to be added to $p(M_i|w_n)$ and N the total number of windows used in the buffer.

Combining the all these equations, the Bayesian Fusion algorithm works as follows; the confidence values for each movement are stored in a buffer along with the $N - 1$ previously calculated values. The conditional probabilities are calculated for all movements M_i according to,

$$p(M_i|w_1, w_2, ..., w_N) = \Delta \prod_{n=1}^{N} (p(M_i|w_n) + k_n) \tag{2.7}$$

and the movement with the highest conditional probability is taken as the desired output.

2.3.3 Decision-Based Velocity Ramp

The Decision-Based Velocity Ramp, from hereon called only Ramp, was introduced by Simon et al. (2011*a*). It takes a different approach than Majority Vote and Bayesian Fusion by changing the speed of the outputted movement rather than the predictions themselves. The algorithm works by adding an acceleration to the outputted movement, forcing movements to start at low speeds and increase the speed as predictions of the same movement are made consecutively. The algorithm does not induce any controller delay since every prediction is outputted, the spurious misclassifications as well. The strength of the algorithm is that these misclassifications will be outputted at a low speed, reducing their impact on the controllability of the prosthesis. An additional feature of the algorithm makes sure that recently outputted movements can be resumed at a higher speed, as misclassified movements would have made the desired movement start from zero otherwise.

The implementation of the algorithm is straightforward, using counters to keep track of the speeds that each movement should be outputted with. Once a prediction has been made by the classifier, the counters associated with the predicted movements are increased by one, and the counter of the movements that were not predicted are decreased by a preset value called the *down count*. The counters are not allowed to increase above a maximum value, called the *ramp length*, or decrease below zero, giving all counters a value between zero and *ramp length*. Only the movement that is predicted is outputted, and the speed at which it is outputted is calculated by,

$$\textit{output speed} = \frac{\textit{counter value}}{\textit{ramp length}} \cdot \textit{desired output speed} \tag{2.8}$$

The parameter *ramp length* determines the number of consecutive predictions it takes for a movement to reach the desired speed. Parameter *down count* makes sure both that undesired movements are outputted with low speeds and that recent movements can be continued at higher speeds. Figure 2.4 shows the output speeds for two movements. The figure was constructed using a *ramp length* of ten and a *down count* of two.

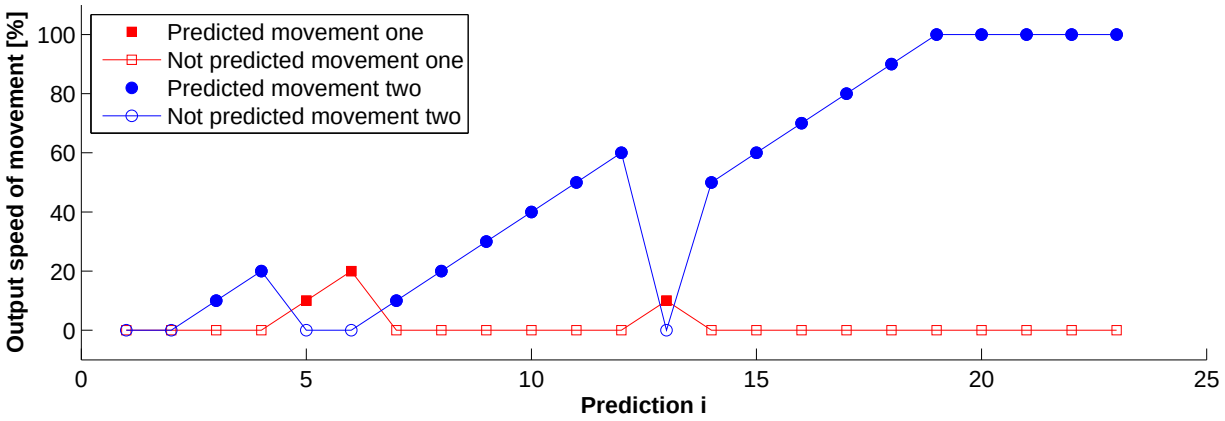


Figure 2.4: The output speed of two movements using the Ramp algorithm. Here movement two is the desired movement. At predictions 5, 6 and 13 the classifier makes a misclassification and predicts movement one. The speed is ramped up towards the desired speed as more predictions are made consecutively. Every prediction of the movement increases its counter by one, and any prediction not including the movement decreases the counter by *down count*. This figure were generated using a *ramp length* of ten and a *down count* two.

2.4 Control System Performance Evaluation

When a control system is designed it is important to be able to evaluate its performance to see that the design actually is beneficial. Many of the articles that treat the myoelectric control problem are focused on the pattern recognition part, testing different training and classification algorithms together with different feature sets to see which combinations performs the best. The performance is often reported by the classification accuracy of the pattern recognition system, i.e. the system is tested on a prerecorded data set and the accuracy is given as the percentage of correct classifications on that data set. However, Lock (2005) and Lock et al. (2005) showed that the relationship between classification accuracy and real time prosthesis controllability are very weak. Even if a system only makes a few classification errors, these errors can have a big impact on the overall control. If all classification errors occur within the same prosthesis function, that function maybe uncontrollable by the user and thus become useless. In contrast, Smith et al. (2011) report that they do find a statistically significant relationship between classification accuracy and real time controllability. This would make sense since a system that more or less randomly moves the prosthesis will be totally uncontrollable, while a system that outputs the desired movement all the time will be perfectly controllable. If nothing else, Lock (2005) showed that important information is lost when only the classification accuracy is presented. It does not translate well to the real time controllability of the prosthesis. It is obvious that real time performance tests are of great importance and two of them are presented below. These are the Target Achievement Test (TAC-test) and the Motion Test, both has been used in this thesis.

2.4.1 Target Achievement Test

The TAC-test is a test that was proposed by Simon et al. (2011*b*) to compare the real-time controllability between different prosthetic control systems. The test is performed in a virtual reality environment where subjects control a virtual limb. During the test, subjects are asked to position the virtual limb in different postures, called targets, and measurements are performed on how fast and efficiently they can be reached. Targets are said to be completed if the relative distance between the virtual limb and the target posture can be held within a certain limit for a set period of time.

This thesis has used implementation of the TAC-test in BioPatRec. BioPatRec is an open source software implemented in MATLAB that has been designed to promote research on myoelectric control systems. The virtual limb along with a target posture as they are implemented in BioPatRec can be viewed in figure 2.5.

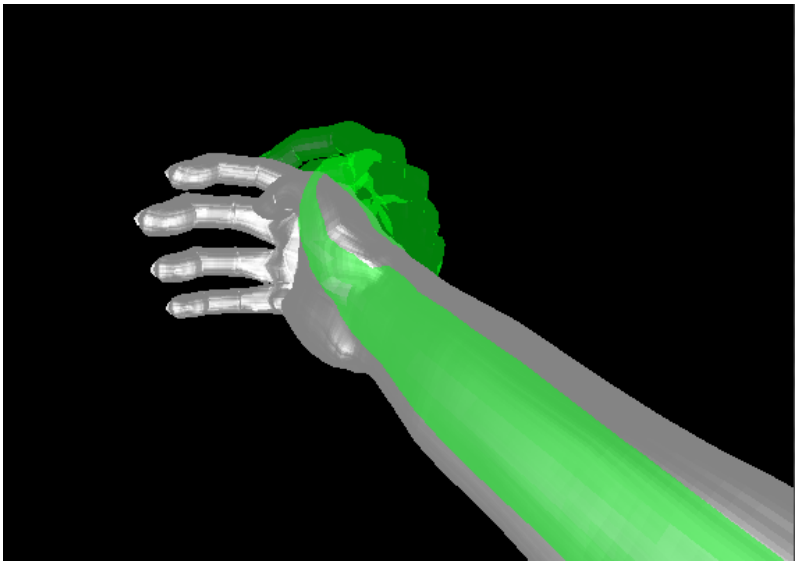


Figure 2.5: Target achievement test as it is implemented in BioPatRec. The silver colored arm is controlled by the subject and the green arm shows the desired posture. During the TAC-test the user tries to position the silver hand in the same posture as the green. If the subject is close enough, the silver arm disappears and the subject tries to hold the posture for set period of time.

2.4.2 Motion Test

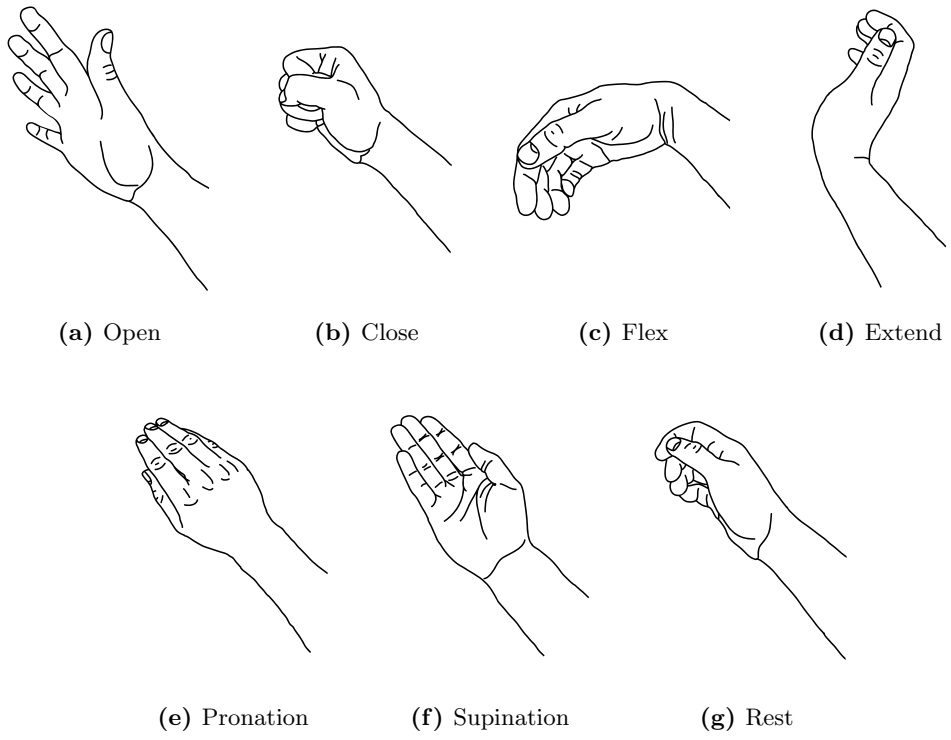
The motion test is simpler than the TAC-test and was first introduced by Kuiken et al. (2009). The user does not control a virtual prosthesis and the performance is instead measured by the systems capability to recognize motions in real time. The test is conducted by asking the user to perform one of the trained motions and hold that motion for a set period of time. If the motion is predicted by the system enough times it is said to be completed. Each test consist of several tasks where the user is asked to perform a motion, making sure that the user performs all the trained motions equal amount of times but in a random order. Different performance metrics can be extracted from the motion test to compare the results between two systems. Some metrics that can be used are the percent of completed motions, the average time before motions are completed and the average time before the correct motions are predicted. This is a real-time prediction test, while the TAC is a real-time controllability test.

Specification of Given Control Problem and Control System

In this work the effects of using post-processing algorithms have been studied with a specific setup of the prosthesis control problem. The control problem involves the simultaneous control of six individual movement, allowing an artificial hand to be moved in three degrees of freedom. All movements that has been used are described in section 3.1. To allow users to control these movements, a new myoelectric control system has been developed at Chalmers, COO and Integrum AB. The post-processing algorithms have been evaluated using their system. The same system has been used throughout the work of the thesis. Setup and operation of the system are described in section 3.2.

3.1 Prosthetic Control Problem

The control problem analyzed in this thesis consists of the simultaneous control of six movements involving three degrees of freedom of the hand and wrist. The movements can be combined in 26 different combinations and with the additional 'No Movement' class these movements make a 27-class problem. The six individual movements are illustrated in figure 3.1 and all the 26 combinations are listed in table 3.1.



(a) Open **(b)** Close **(c)** Flex **(d)** Extend

(e) Pronation **(f)** Supination **(g)** Rest

Figure 3.1: The six individual movements considered in this thesis. The six active movements can be combined in 26 different combinations. With the addition of a Rest-class it makes a 27-class problem for the classifier.

Index	Movement	No active dofs
1	Open Hand	1
2	Close Hand	1
3	Flex Wrist	1
4	Extend Wrist	1
5	Pronation	1
6	Supination	1
7	Open Hand + Flex Wrist	2
8	Close Hand + Flex Wrist	2
9	Open Hand + Extend Wrist	2
10	Close Hand + Extend Wrist	2
11	Open Hand + Pronation	2
12	Close Hand + Pronation	2
13	Open Hand + Supination	2
14	Close Hand + Supination	2
15	Flex Wrist + Pronation	2
16	Extend Wrist + Pronation	2
17	Flex Wrist + Supination	2
18	Extend Wrist + Supination	2
19	Open Hand + Flex Wrist + Pronation	3
20	Close Hand + Flex Wrist + Pronation	3
21	Open Hand + Flex Wrist + Supination	3
22	Close Hand + Flex Wrist + Supination	3
23	Open Hand + Extend Wrist + Pronation	3
24	Close Hand + Extend Wrist + Pronation	3
25	Open Hand + Extend Wrist + Supination	3
26	Close Hand + Extend Wrist + Supination	3
27	Rest	0

Table 3.1: Movement used to define the prosthesis control problem for this thesis. The six individual movements presented in figure 3.1 can be combined in 26 different ways. The table presents all 26 combination together with the resting movement. The movement indexes in the table are the same that is used by the myoelectric control system and can be used to translate results found in the thesis back to the movements.

3.2 Myoelectric Control System

A pattern recognition based myoelectric control system, using BioPatRec, that can recognize combined movements as well as individual has been used for this work. A conceptual image of the system is presented in figure 3.2 and the operation of the system is described below.

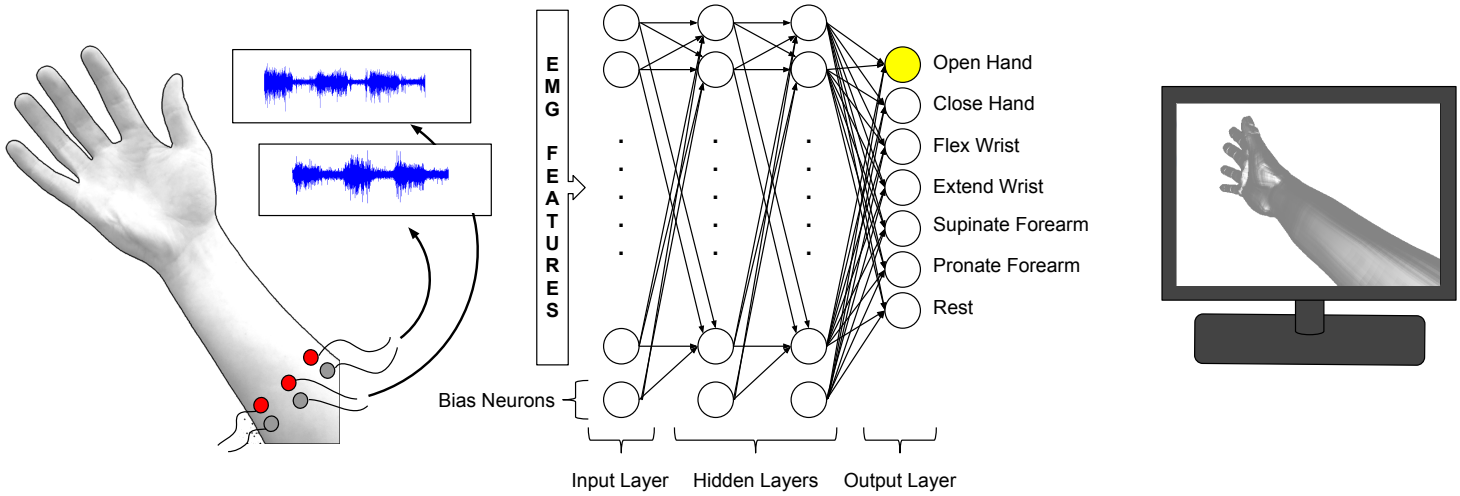


Figure 3.2: Conceptual description of the myoelectric control system used in the thesis. The electromyogram is recorded from the skin of the forearm using bipolar Ag/AgCl electrodes. Each electrode channel produces its own time series of the MES which is represented by a set of characteristic values, signal features. The signal features from the different channels are fed to a MLP that predicts which movements the user is trying to perform. The myoelectric system uses one output neuron per individual movement, seven including the no movement neuron, and allows several of them to activate at the same time. The activated output neurons are interpreted as the desired movement and a virtual hand is moved in the corresponding manner.

The system uses surface MES recorded from eight untargeted sites around the most proximal third of the forearm to control a virtual limb. Recordings are made using commercially available Ag/AgCl electrodes. The signals are amplified 66 dB and filtered with a 4th order high-pass filter at 20 Hz, a 2nd order low-pass filter at 400 Hz and a Notch filter at 50 Hz using an in-house designed amplifier. The signals are then digitized at 2 kHz with 16-bit resolution using a data acquisition card from National Instruments.

The system uses four time-domain features in the classification; mean absolute value, waveform length, number of zero crossings and the number of slope sign changes. A single MLP is used to make the classifications. The MLP has two hidden layers, both with the same number of neurons as the input layer, making the architecture 32-32-32-7 as there are 32 inputs and seven outputs. The user undergoes a guided recording session to collect training data for the classifier. During the training session, the software ask the user to perform all 26 movements in table 3.1. Each movement is contracted three times for three seconds with three seconds of rest between, collecting nine seconds of data from each movement. Fifteen percent of the data is removed from the beginning and end of each contraction. This to make sure that any inactive portions of the signal are removed before presenting it to the classifier. The remaining data is windowed with an overlapped scheme using a 200 ms analysis window and 50 ms time increment. The four singal features are then extracted from the time windows. The resting period between contractions are used to construct a 'No Movement'-class, calculating an equal amount of features as for the other movements. The resulting features are split into three data sets, one for training (40%), one for validation (20%) and one for testing (40%). The features are randomized amongst the three sets, making sure that data from all contractions could end up in either of the three data sets. Normalization constants are calculated from features in the training set. The constants are used to normalize both the other two data sets and features that are calculated during real-time operation. A movement onset threshold is calculated from the 'No Movement'-class features in the training set, by averaging the mean absolute value across all eight channels. This threshold must be overcome during real-time operation for the classifier to make a prediction, this to remove activation due to noise in the

signals.

The MLP classifiers are trained using stochastic backpropagation with fixed momentum $\alpha = 0.1$ and learning rate $\eta = 0.1$. For each iteration in the training, 70% percent of the feature vectors in the training set are randomly chosen and used to update the weights. The feature vectors in the validation sets are then used to evaluate the performance of the MLP, and the best version of the MLP is saved based on the root mean square error (RMSE) and the number of correctly fired neurons. The convergence criterion of the training is based on the average RMSE. The training is stopped if the RMSE is lower than 0.1, and no progress is observed amongst the most recent five iterations compared to the most recent fifteen. The training is otherwise allowed to continue for a maximum of 200 iterations.

During real time operation of the system, the recorded signals are windowed using the same scheme that is applied to create the training data, i.e. an analysis window of 200 ms and a time increment of 50 ms. The four time-domain features are extracted from the analysis windows and normalized using the constants that were stored during the training. The average mean absolute value across the eight channels are compared with the onset threshold. If it is larger than the threshold, the system makes a prediction, otherwise the system automatically predicts the 'Rest'-class. All neurons in the MLP are using sigmoidal activation functions and a value of 0.5 of more is considered as firing. Each output neuron in the MLP is representing one of the possible individual movements and the firing neurons are used to construct the predicted movement. The predicted movement is send to a virtual environment were it is used to move a virtual limb in a corresponding manor. The limb moves a preset of degrees for each prediction, making it move at a constant speed.

Analysis of Available Motion Test Data

The aim of the first part of the thesis was to understand the real-time misclassifications that are produced by the myoelectric control system. For this purpose, data from 12 previously conducted motion tests was analyzed. All time-data recorded during the real-time tests had been stored, forming a good data set. The tests had been conducted by my supervisor, Max Ortiz Catalan and six different subjects participated, conducting two motion tests each.

4.1 Description of Data Used in Analysis

The data used in the analysis had been collected from six different subjects performing the 26 simultaneous movements described in section 3.1. The real time classification during these tests had been made using MLP classifiers configured in two different topologies, a single perceptron (single) and one perceptron per individual movement (one-vs-all) (Ortiz-Catalan et al., 2013*b*). All motion tests had been conducted with the system described in 3.2, with the exception that only half of the training and validation data were used to train the classifiers.

Each motion test consisted of two trials which in turn included three repetitions of all 26 motions. The resulting data had been collected during 72 repetitions of each movement. During the experiment, a motion was said to be completed if it had been predicted 20 times. Subjects had a total of 10 seconds to realize which motion was desired, execute it and hold it. The movements had been presented in random order to the user to remove any bias as subjects adapted to the system. The two different topologies had been tested in random order on the subjects for the same reason.

4.2 Analysis Method

The motion test data was analyzed by calculating a few characteristics from each repetition of every movement. The characteristics that were calculated were; the number of misclassifications during that repetition, the number of active movements during the misclassifications and the number of consecutive misclassifications activating or not activating a specific movement.

Misclassifications were categorized by two properties during the analysis, those that activate an undesired movement and those that do not activate a desired movement. Example given, if the movement 'Open Hand' is desired while 'Open Hand + Flex Wrist' is predicted, then an undesired movement is activated making the misclassification have property one. If 'Open Hand + Flex Wrist' is desired but only 'Open Hand' predicted, a desired movement is not activated making the misclassification have property two. A misclassification could have both these properties, e.g. if 'Open Hand' is desired but 'Flex Wrist' is predicted, then is an undesired movement activated at the same time as a desired one is not.

Misclassification An output that does not correspond to 100% with the desired output.

Undesired Activation An output that activates one or more movements that are not desired by the user, one or more false positives in a prediction.

Interruption An output that does not activate one or more movements that are desired by the user, one or more false negatives in a prediction.

Some of the initial time data was removed from each repetition before it was analyzed. This was done to remove the initial period before subjects had started performing the movements. To make sure that the

user had started performing the correct movement, data before five consecutive predictions containing the correct motion had been made was removed. The number of consecutive correct predictions were counted for each of the desired movement separately and irrespectively of the predictions of other movements. In the case of combined movements, all the involved movements had to be predicted at least five times each, but not necessarily together.

The number of consecutive misclassifications within a specific movement were extracted from each repetition since all the proposed post-processing algorithms in section 2.3 utilizes consecutive predictions in one way or another. It is of great importance that not too many consecutive misclassifications occur within the same movement, as it makes it impossible to remove the effects of the misclassifications. The cases between the two different types of misclassifications were split, consecutive misclassifications that did not output one of the desired movements were called interruptions and those that activates one or more undesired movements were called streaks. The concepts of interruptions and streaks can be viewed in figure 4.1. Streaks and interruptions that crossed either the starting or the ending of the analyzed data were not counted because the data was cut, either by the repetition stopping or the choice of starting index (they were still counted as misclassified predictions).

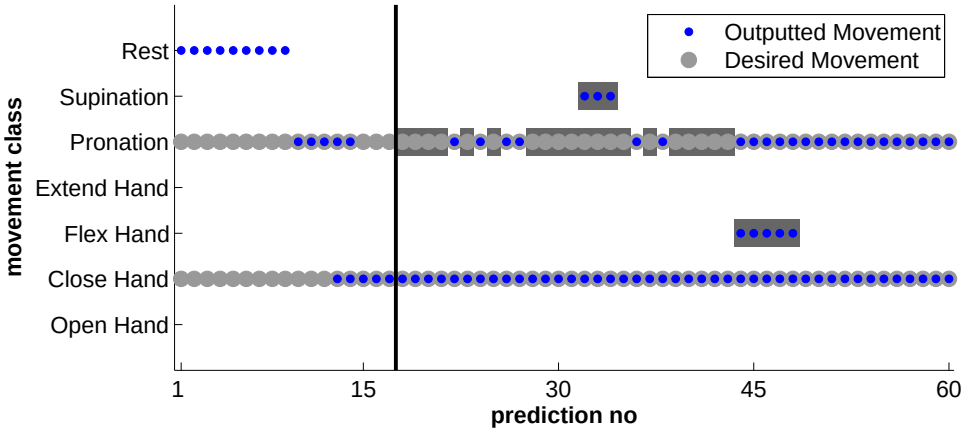


Figure 4.1: Real time predictions made during a motion test. The desired movement in this trial is the combined movement of 'Close Hand' and 'Pronation'. The subject is initially at rest when presented with the motion and does not start to move until some time has passed. The analysis only used data after every desired movement had been predicted at least five times each, here represented by the data to the right of the black line. The gray boxes indicates what were called interruptions (number of consecutive predictions not predicting a desired movement) and streaks (number of consecutive predictions predicting undesired movements). The first interruption occurs at the beginning of the analyzed data and is not considered in the analysis because it is chopped in length. This particular example would count as one streak of length three, one streak of length five, three interruptions of length one, one interruption of length five and one interruption of length eight.

The percent of predictions that were misclassified were calculated from each repetition of the motion test to see how well the system had performed during real time operation. The results were divided into three groups, movements involving one, two and three DoFs respectively to allow any differences amongst the three cases to be noted. The percentage of misclassifications that had properties one and two respectively were also calculated, to see if either of them caused bigger issue in the system. The percentages were calculated per repetition, and the results were averaged across all 72 repetitions.

The six motion tests that had been performed with the single topology were further used to extract the confidence values of the MLP classifier. The confidence values were extracted for all activated movements and were divided into two groups, correctly activated and wrongly activated movements. The resting class were excluded from the analysis as it is the only movement that can be outputted without exceeding the MLP threshold. Rest can be outputted either if the mean absolute value across all channels do not exceed the onset threshold, if the MLP is not confident enough to output any movement or if the resting class exceeds the MLP threshold. Including the resting class in the calculations would mean that confidence values that had nothing to do with the predictions would be included in the analysis.

4.3 Results

Depending on the complexity of the movement, between 20-30% of the predictions made during the motion tests were found to not be the same as the desired. These results are presented in figure 4.3, which presents an average across all 72 repetitions made during the 12 motion tests that were analyzed. On average the system misclassified more predictions when the subjects were asked to perform individual movements than when they were asked to perform combined movements. However, predictions were found to be misclassified for all 26 movement combinations, as can be seen in figure 4.2.

The percentages of misclassified predictions that had properties one or two are presented in figure 4.4. The results show that the system activated a lot of undesired movements but rarely deactivated the correct one when subject performed individual movements. The system had the reverse problem when subjects were asked to perform movements involving all three DoFs. The correct movements were then deactivated, but the system rarely activated any undesired movements. Movements involving two DoFs showed a mixed result between the two problems.

Figure 4.5 shows the number of activated movements during the misclassified predictions. The percentages were extracted from each of the 72 repetitions and the averages are shown in the plot. The rest movement was counted as zero active movements as it does not produce any output. The figure shows that misclassified predictions during both individual and three DoF movements most commonly activated two movements. During two DoF movements the most common misclassification activated three movements. Sometimes even four movements were found to have been activated, which is possible because the system allows opposing movements in the same DoF to be activated simultaneously.

The distributions of streak and interruption lengths are plotted in figure 4.7 were they are given as percentages of the total number of streaks and interruptions found during all repetitions. Most often the system only misclassified one consecutive prediction, meaning that the misclassification should be identifiable as long as it is not occurring in an on/off manor. If the same error is made too frequently it is impossible to tell whether it is an undesired activation or an interruption. Noticeable is that there is a change in the trend around four consecutive predictions, 200 ms, where longer streaks and interruptions are becoming more probable again.

The MLP confidence values were extracted from the single MLP classifiers for every activated motion. Activations were categorized into two groups, correctly and incorrectly made activations. The distributions of these values are presented in figure 4.6. It was found that the majority of the desired activations were made with a confidence of 0.9 or higher. Movements activated with confidence of 0.8 or less were on average equally probable to be desired as undesired.

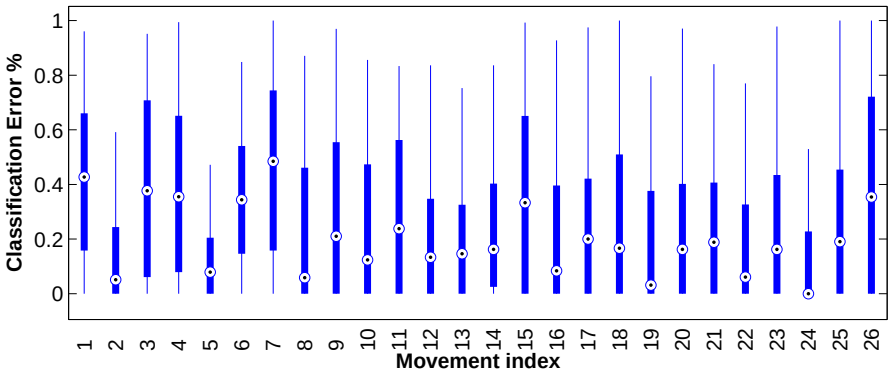


Figure 4.2: Percent of misclassified predictions per movement. The percentages of misclassified predictions were calculated from each repetition of the motion test and are presented in the figure as a boxplot. The boxes include the 25th and 75th percentiles and the whiskers reach to the upper and lower adjacent values. The adjacent values are the most extreme values that are not considered to be outliers, and outliers are calculated with equation (B.1) that can be found in the appendix. The circle in the middle illustrates the median value amongst the 72 repetitions of each movements. The figure shows that misclassifications are happening with all movements, some movements having more misclassifications than others. Movement one "Open Hand" seems to be harder to predict correctly than the opposing movement, movement two "Close Hand".

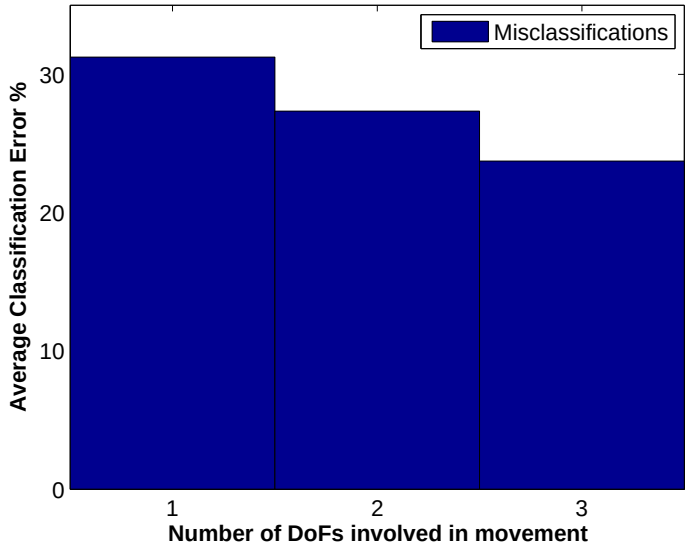


Figure 4.3: The average percent of predictions that were misclassified across the 72 different repetitions of motion tests that were considered in the analysis. The figure shows that the extracted classification error on average was unacceptably high. Movements involving one DoF were on average misclassified more than movements involving more DoFs.

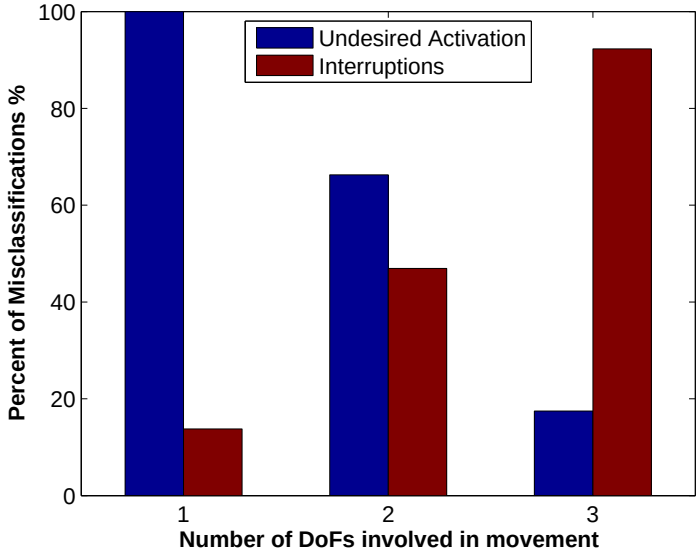


Figure 4.4: Percentage of misclassifications that involved undesired activation and interruptions respectively. Blue bars represents misclassifications that activated one or more undesired movements and red bars misclassifications that did not activate one or more desired movements. Movements involving one DoF are almost always misclassified by activating undesired movements but rarely by not activating the desired one. The combination suggests that movements are misclassified by activating the desired movement plus one or more undesired movements as well. Movements involving three DoFs has the opposite relationship between the two misclassification types. They are misclassified by one or more of the desired movements not being outputted. Movements involving two DoFs has a better balance between the two, suggesting that those movements are misclassified in both ways.

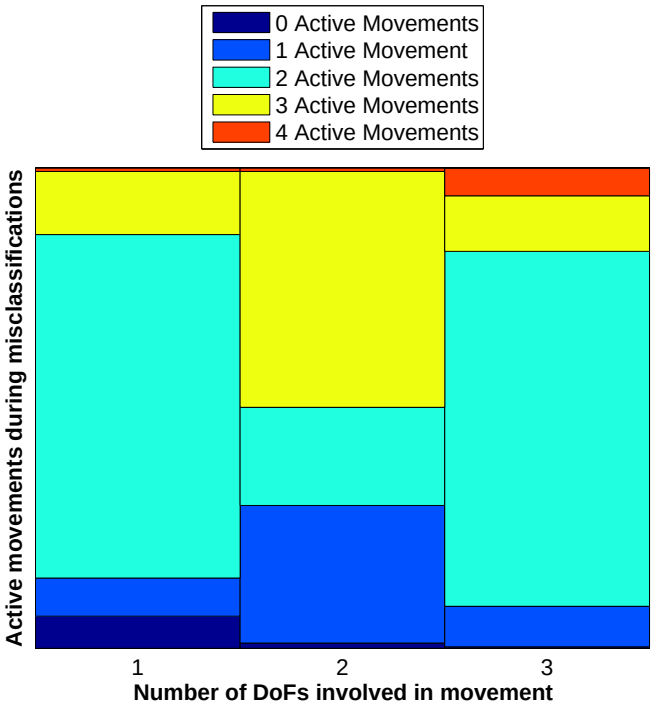


Figure 4.5: The number of active movements during misclassifications. The figure shows that two movements were active during most of the misclassifications both for one and three DoF movements. Misclassifications of two DoF movements mostly activated three movements. It can be noted that sometimes even four movements were activated during a misclassification which should not be possible with the three DoF system. This result is however possible due to the implementation of simultaneous control in the system that allows two opposing movements to be activated simultaneously.

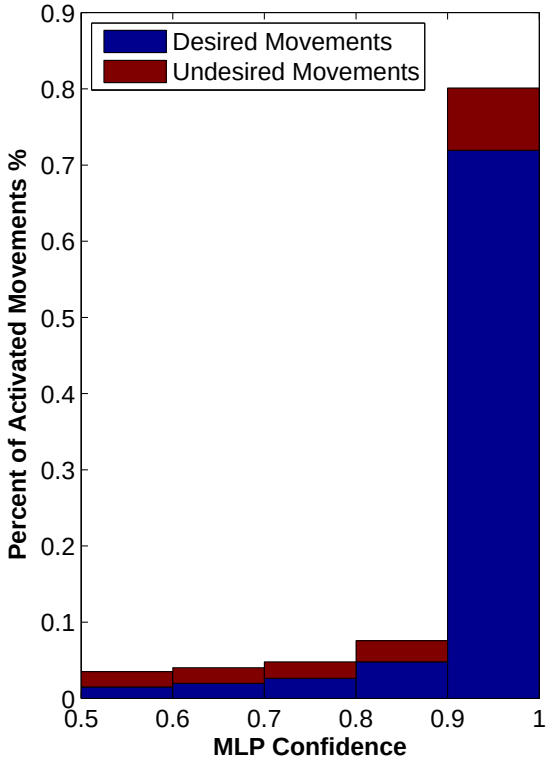
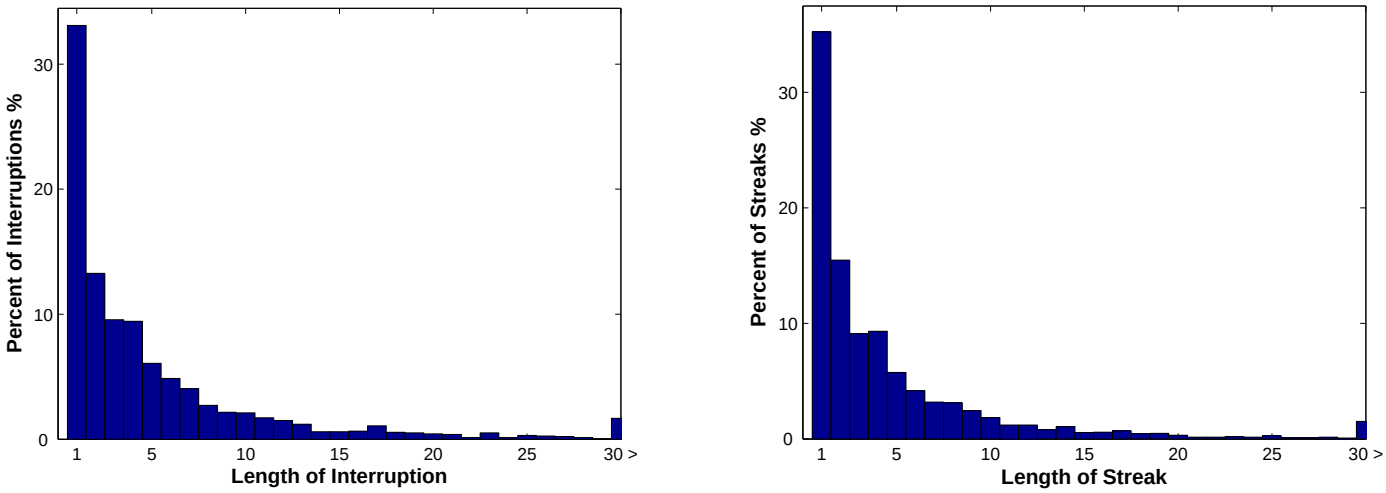


Figure 4.6: MLP Confidence values extracted from the MLP classifiers with single topology. The values have been divided in two groups, desired and undesired activations. The figure shows that most of the activations were made with the MLP classifiers being more than 0.9 confident. Confidence values of 0.8 or less generated approximately equal amount of undesired activations as desired. It would be desired to increase the output threshold of the MLP classifier to reduce the percent of misclassified movements.



(a) **(b)**

Figure 4.7: Distribution of streak and interruption lengths made during real-time predictions. The distribution of interruption lengths is presented in (a), and (b) shows the distributions of streak lengths. Most of the streaks and interruptions are of short lengths which would allow a post-processing algorithm to possibly remove them. It can be noted that something happens in the distributions at the fourth prediction, i.e. after $200ms$, where lengths are becoming more probable again. This trend could possibly indicate the reaction time of the subjects as they had visual feedback of the real-time predictions. After $200ms$ the subject had time to react to any misclassification and adjust the contractions to achieve the desired result. This would mean that there exists two types of misclassifications, spurious ones made by the system, and longer ones due to the user performing the contractions in a way that the system can not recognize.

4.4 Discussion

Misclassifications are a problem with the current system as, on average, 20-30% of the predictions were misclassified. The classification accuracy were calculated the same way as it is in BioPatRec, i.e. by comparing the predicted movements with the desired ones, counting the prediction as misclassified if the two are not the same. However, the measurement gives a skewed picture of the performance of the system since a prediction can include multiple movements. Any desired movements in a misclassified prediction are unaccounted for, even though the system correctly activated some movements. A better performance metric should therefor be used to tell whether or not the misclassifications cause the system to be uncontrollable or not.

Further did the analysis only considered data after five predictions containing the correct movement had been made by the classifier. The condition was chosen to make sure that the user actually was performing the desired movement. The constraint was applied to all the desired individual movements irrespectively of each other, meaning that they did not have to be predicted together to count. By applying the constraint to movements individually, initial misclassifications that activated undesired movements could be present in the data. However, if the initial misclassifications were due to one of the movements not being predicted, the data would be lost. This means that the results in figure 4.3 could be shifted to favor three DoF movements, where almost every misclassified prediction was missing one or more of the desired movements.

It was noted that individual movements were misclassified with zero active movements more than the combined movements. The increased frequency of rest being predicted could be the result of two different causes, either the subjects did not overcome the onset threshold or they rested after they started the movement. The second case could be a way for subjects to "restart" the movement in an attempt to perform it in a way that the classifier could recognize. Both cases show the difficulty of performing individual movements during real time operation.

A shift in the trend of the distribution of streak and interruption lengths were noticed at 200 ms, where longer streaks and interruptions suddenly became more probable again. This could possibly be explained by the reaction time of the subjects before they had visual feedback of the predictions. If a movement was performed in a way such as the system could not recognize it, the reaction time would limit how fast subjects could compensate for the mistake. This would mean that two different types of streaks and interruptions are present in the system, those that are spurious and those where the user performs the movement in such a way that the classifier consistently predicts the wrong movement. The second type of mistakes are reduced as the user adapts to the system, learning how to perform movements in a way so the system can recognize them. Streaks and interruptions were most of the time observed only for few consecutive predictions. As long as they do not occur in an on/off fashion, they should be recognizable by a post-processing algorithm. Movements that are predicted in an on/off fashion could have trouble with either undesired activation or interruption, and it is not possible to decide if the movement is desired or not. It should be noted that there exist more possibilities to have shorter streaks and interruptions in the analyzed data. An on/off behavior of a misclassification could generate streaks or interruptions of lengths one every other prediction, while interruptions or streaks of lengths two can only occur every third prediction.

Most of the results were calculated as average percentages across the 72 repetitions. This was done because the repetitions all contained different amount of time data. The repetitions had ended once the user had performed the desired movement for one second. Repetitions where the subject easily could perform the movement only contained a few predictions, while repetitions where the user struggled contained a lot of predictions. The number of misclassified predictions could therefor not be compared with the total number of predictions made, as a lot of the good predictions would be left out. Ideally a separate experiment would have been performed, where it could be certain that the subjects were performing the desired movements, and where all repetitions were performed for equal amount of time. The lack of such uniformity is a limitation of the current method and data sets.

Movements that are activated by the MLP with less than 0.8 confidence proved almost equally probable to be undesired as desired. By increasing the MLP threshold from the current value of 0.5 to 0.9, roughly 50% of the undesired activations would never have happened. The increased threshold would only have missed roughly 13% of the desired movements, meaning that there is a lot of performance to gained by doing this. To further optimize the classification, the thresholds should be set differently

for different movements depending on how easily they are activated. This is however a problem with the classifier and is not investigated further in this work as it has been studied by Ortiz-Catalan et al. (2013*c*).

4.5 Conclusions

Analyzing the available motion test data has shown some interesting properties of the misclassifications and their occurance in simultaneous myoelectric control. The system described in section 3.2 were found to misclassify 20-30% of the predictions made during real-time operation. However, the classification accuracy as it is calculated by BioPatRec does not reflect the performance of the system. Correctly activated movements during the misclassified predictions are not accounted for, making the measurement worse than it actually is. Individual movements are on average misclassified more often than combined movements, and the biggest issue with these movements is due to the activation of additional movements. The same is true for two DoF movements, but the situation is another with three DoF movements, where the biggest issue is due to the desired movements not being activated properly.

The current MLP thresholds were found not to be optimal for the classification. Increasing the thresholds would reduce the number of wrongly activated movements and could increase the performance of the system with one and two DoF movements. This would decrease the performance with three DoF movements, as movements would be harder to activate. However, the improvements gained were found to be greater than the losses, making it desirable to increase the thresholds.

Two different kinds of misclassifications has be shown to be present in the system, those that are intrinsic to the system, and those where the user performs the movements in such a way that they are unrecognizable to the system. Most of the time only a few misclassifications are made consecutively. As long as they do not occur in an on/off fashion, a post-processing algorithm should be able to detect them while still operating within an acceptable controller delay.

Choosing and Modifying Algorithms for Simultaneous Control

The goal of this thesis is to bring post-processing algorithms from myoelectric control systems of individual movements to those of simultaneous movements. This step is not trivial and depending on how the simultaneous control is implemented in the control system, these algorithms may have to be modified in order to work at all. The myoelectric control system that has been used for this work treats the combined movements as combinations of the individual movements, rather than as separate classes themselves. This approach to the simultaneous control problem inevitably leads to some problems as the buffer based post-processing algorithms work by outputting the most likely movement class and several movement classes can be desired simultaneously.

The Majority Vote algorithm in its traditional form only allows the output of one movement class. This would be a problem in the simultaneous case as the classifier can predict several movement classes at the same time. The first approach to solve this problem was to allow the algorithm to output all movements with the same amount of votes in the buffer, but two more versions of the algorithm are also suggested in this chapter.

The Bayesian Fusion algorithm was not considered for this thesis as it could not easily be modified to work with the simultaneous control approach taken.

The Decision-Based Velocity Ramp could be adopted in its traditional form as it outputs every predicted movement, regardless if multiple movements are predicted. A modified version of the algorithm is also presented based on the results found in chapter 4.

5.1 Majority Vote

Three different versions of the Majority Vote algorithm are proposed to allow the algorithm to work with the simultaneous control system. The proposed algorithms takes three different approaches and are described below as well as presented in figure 5.1.

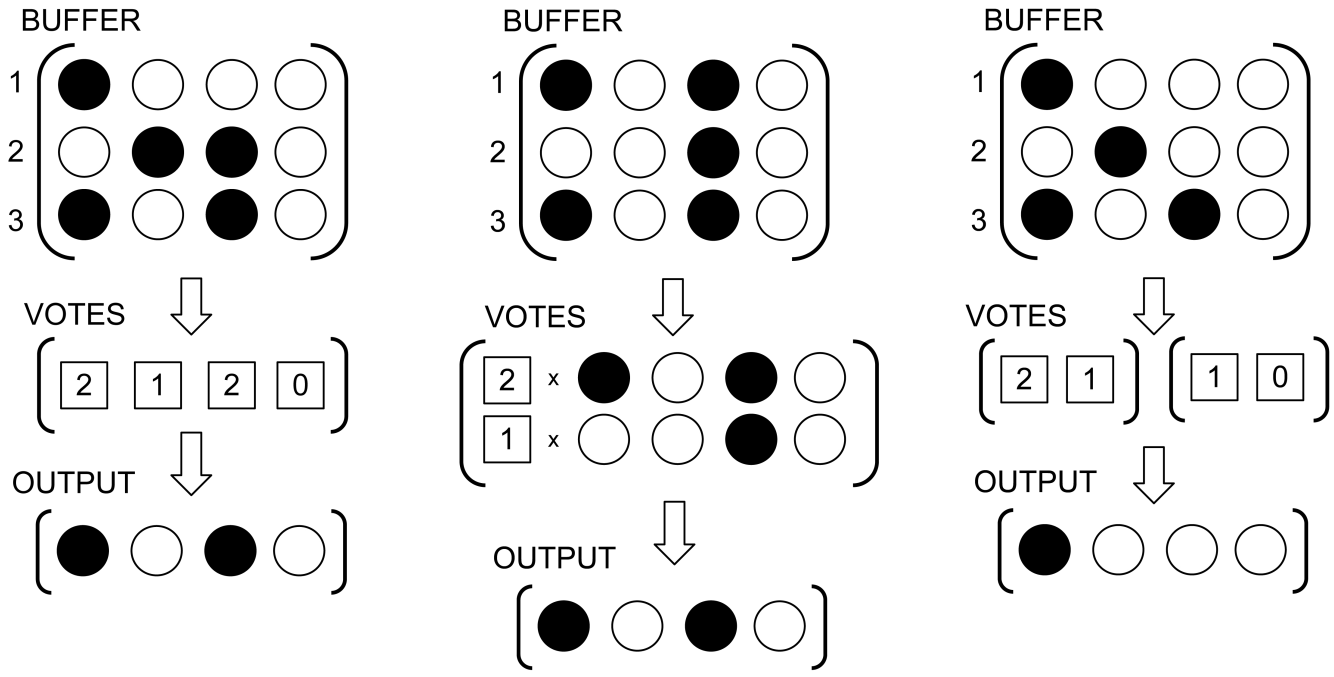
Majority Vote One

The first version of the Majority Vote algorithm works the same way as the traditional version, it holds a voting process amongst the individual movement classes and outputs the one with the highest number of votes. The difference to the traditional majority vote is that if several movement classes get the same number of votes, they are all outputted.

This approach to the Majority Vote algorithm would be suitable to prevent undesired activations as movements have to have the same amount of votes to be outputted. If a movement is spuriously activated by misclassifications it will not get the same number of votes and thus not be outputted. However, this approach would penalize interruption of desired movements extra hard as an interruption would cause one of the desired movements to have one less vote than the other, hindering it from being outputted.

Majority Vote Two

The second version of the Majority Vote algorithm treats the movement combination of each prediction as a movement class itself. A voting process is held between the movement combinations instead of the individual movement classes, and the combination with the highest number of votes is outputted. If



(a) MV1 **(b)** MV2 **(c)** MV3

Figure 5.1: Three versions of the Majority Vote algorithm are suggested which would allow the algorithm to work with simultaneous control. The buffer contains the three most recent predictions made by the classifier and the predicted movements are illustrated by black circles. The algorithms are illustrated using four possible movements, one in each column of the buffer, and they are assumed to be pairwise ago-antagonist movements. The first algorithm, MV1, performs one voting process giving each movement a vote and outputs all the winning movements. MV2 treats the combination of movements in a prediction and performs the voting process amongst all unique patterns found. The last algorithm, MV3, performs one voting process per ago-antagonist pair making sure that only one movement in each DoF can win. The winning movements are compared against a no-movement condition to see whether or not they should be outputted at all. The condition requires a movement to have more than $N/2$ votes, where $N = 3$ is the number of predictions in the buffer, to be outputted.

several movement combinations have the same amount of votes the most recently predicted combination is chosen to be outputted.

This approach to the majority vote problem would be able to recognize both undesired activation and interruptions. The algorithm does not recognize individual movements and could possibly fail if the predicted combinations vary a lot, even if they all contain the same desired movement.

Majority Vote Three

The third version of the Majority Vote algorithm holds one voting process per DoF. In this version e.g. the "Open Hand"-class competes against the "Close Hand"-class. A "No Output"-condition is formed for each DoF by forcing the winning movement to have more than $N/2$ votes (N is the buffer size) if it is to be outputted. The Rest-class has no competitor and only competes against itself. This is done in the same way as the other movements competes against themselves, it has to have more than $N/2$ votes. If the Rest-class wins over itself, all other winning movements are disregarded and no movement is outputted.

This version of the algorithm should be able to identify both types of misclassifications and could therefor potentially increase the overall performance of the system. The algorithm imposes a constraint not to allow competing movements in the same DoF to be outputted simultaneously. It was noted in section 4.3 that the control system did allow this, and the addition should therefor be beneficial for the performance of the system. The constraint could however just as well have been added in the classifier itself.

5.2 Bayesian Fusion

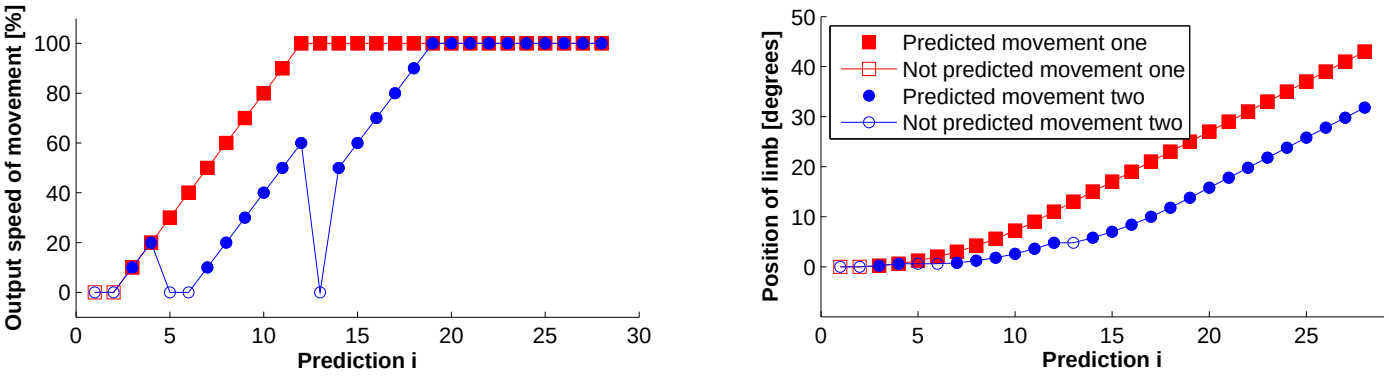
The Bayesian Fusion algorithm was designed for individual control and showed good results compared to the Majority Vote, especially when bigger buffer sizes were used. To cope with the fact that bigger buffer sizes induce longer controller delays, Khushaba et al. (2012) proposed that the buffer should be reset once the resting class was predicted and that voting should occur between the votes present in the buffer. This approach only works if all the movement transitions are made via the resting class, which might work well with the sequential nature of individual control. However, one of the major benefits of having a simultaneous control is the possibility of on- and offsetting of movements during the execution of others. The algorithm also requires a disjoint windowing scheme which would increase the controller delay even more. The algorithm was not chosen to work with the simultaneous control system for these reasons.

5.3 Decision-Based Velocity Ramp

The Decision-Based Velocity Ramp can be applied in its traditional form to the simultaneous control system as it outputs every predicted movement. A problem can however be noticed due to the fact that several motion classes can be outputted at the same time. If one of the movements has an interruption, it will in the traditional implementation be outputted with attenuated speed once it is predicted again. This behavior will cause interrupted movements to 'lag' behind movements that the system recognizes correctly. The problem is illustrated in figure 5.2 where it clearly can be seen that one of the movement has trouble keeping up with the other.

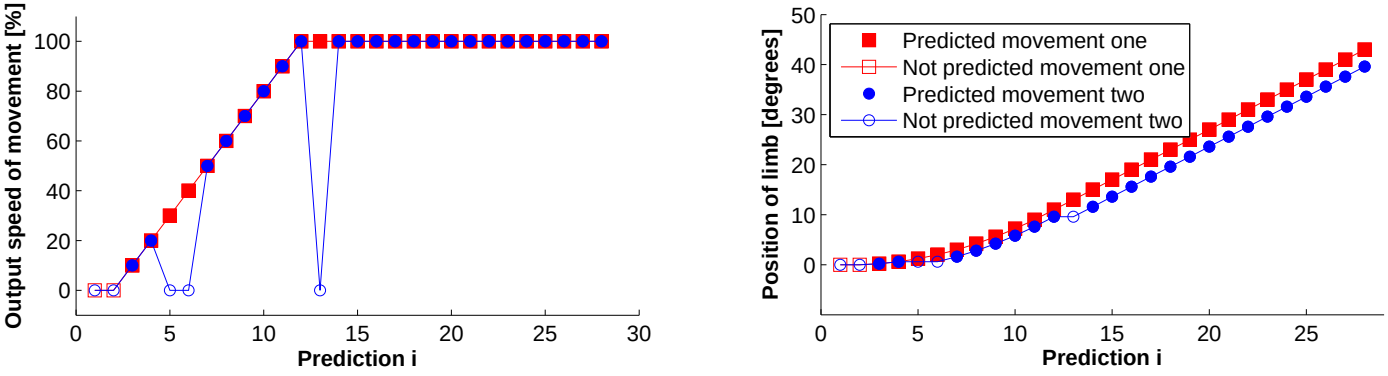
A modified version of the Decision-Based Velocity Ramp is therefor proposed. The modified version does not penalize the output speed during the first interruptions. Since the algorithm works in the same way as the traditional implementation in all other aspects, it is possible that the modified version will perform better. The modified version allows movements that have problems being interrupted to resume with higher speeds and thus reduce the position difference due the interruption. Because all the predicted movements are outputted, the additional feature should not affect misclassifications that activate undesired movements worse than the traditional version, as long as these activations do not occur in an on/off fashion. The modified version works exactly as the traditional ramp described in 2.3.3, except that the counters are increased for recent predictions even though they are not predicted. To compensate for the additional increase in the count, counters are decreased extra much once the movement has not been predicted amongst the most recent n predictions.

The difference is illustrated in figures 5.2 and 5.3, where the same prediction stream has been applied to both algorithms.



(a) Speed **(b)** Position

Figure 5.2: Speed (a) and position (b) of simultaneously outputted movements using the traditional decision based velocity ramp algorithm. Both movements are desired outputs and movement two struggles with interrupting misclassifications. The position is calculated using a prediction rate of 20 predictions per second together with a maximum speed of 40 degrees per second which is a reasonable scenario with the current control system. It can be seen that the position of the limb in the DoF corresponding to movement two is lagging behind the desired position due to the interruptions. This effect might frustrate the user as the limb does not reach the desired position. The user will have to compensate for the interruptions separately once the first movement has reached the desired position.



(a) Speed **(b)** Position

Figure 5.3: The same scenario as in figure 5.2 is depicted, but with the modified version of the decision based velocity ramp algorithm applied. The modified version allows for some spurious interruptions without penalizing the output speed of movements. The additional feature allows movement two to keep up with movement one better and the user will hopefully not have to compensate further once the desired position is reached.

Comparing Majority Vote Based Algorithm

Three new Majority Vote algorithms that would work with the simultaneous control system were presented in chapter 5, and their performances are compared in this chapter. The algorithms are compared by reclassifying the same motion test time data that was used for the analysis in chapter 4. By training new classifiers with smaller analysis windows, the results could be simulated to replicate the case where the system still operates within the optimal control delay and uses Majority Vote post-processing.

6.1 Method

The three Majority Vote algorithms were compared using the same 12 motion tests that had been used for the analysis in chapter 4. The F_1-score, a well known performance metric in the field of machine learning, was used to measure the performances of the algorithms. The score weights the *precision* (the percent of outputs that were desired) and the *recall* (the percent of desired outputs that were found) equally. The two measurements *precision* and *recall* are defined according to the equations (6.1) and (6.2), and the F_1-score itself is defined according to equation (6.3).

$$precision = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} \tag{6.1}$$

$$recall = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} \tag{6.2}$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{6.3}$$

As mentioned, precision measures the percentage of outputs that were desired by the user, and recall measures the percentage of desired outputs that were recognized by the classifier. Both of these properties are essential for the system to perform well, since the system should recognize as many desired movements as possible while producing as few undesired movements as possible. The F_1-score is the harmonic mean between the two measurements, and will give a high score to systems that have both good precision and recall.

To calculate the precision and recall, all outputted movements except the rest-class were considered as positives, and all movements that were not outputted were considered as negatives. The true and false statements mean that the system made a correct or an incorrect assumption. A true positive is when a desired movement is outputted and a false negative when a desired movement is not outputted. The rest-class was left out from the calculations as it corresponds to no movements being outputted, it could therefor not be calculated as a positive.

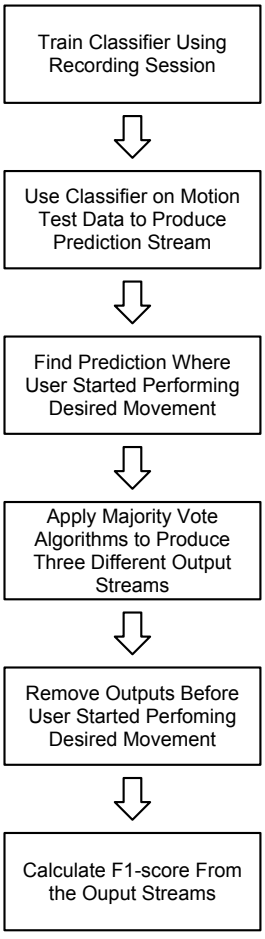


Figure 6.1: Flowchart of the method used when comparing the Majority Vote based algorithms.

The F_1-scores were first calculated using the predictions stored in the motion test data. One score was calculated from each of the 72 repetitions per movement that had been conducted. The scores were divided into three different groups depending on the number of DoFs involved in the movements. The initial predictions from each repetition, before the user had started performing the movement, were removed in the same way as in chapter 4. The same calculations were then done using a new set of classifiers, trained on the same recording sessions as those that had been used during the real time experiments. Ten new classifiers were trained per motion test, and used to make new predictions on the recorded time data. The F_1-scores were calculated using the new predictions, resulting in ten times as many scores. The results were compared with those calculated using the original predictions to assure that the results were similar.

The offline classification showed similar results as the online, which motivated that the Majority Vote algorithms could be compared using the offline calculations. Since this approach was deemed possible, ten new classifiers were trained per motion test. The classifiers were modified to work with the Majority Vote algorithms by using a smaller analysis window. This was to make sure that the system still could be operated within the limits of the controller delay, even when using the Majority Vote. The controller delay was chosen to match that of the current system, $125 + \tau$ ms. This controlled delay can be achieved in a great variety of ways using various windowing schemes, see equation (2.2). To keep the parameters to a minimum, the time increment was held fixed at 50 ms throughout the calculations. This resulted in one windowing scheme that fully used the controller delay limits, i.e. a 100 ms analysis window with $n = 3$ predictions in the buffer.

All classifiers that were trained for the offline calculations were single MLP classifiers. The MLPs were trained in the same way as the classifiers had been during the real-time tests, using the standard routines in BioPatRec described in 3.2. The classifiers were used to make new predictions on the motion test data. The three Majority Vote algorithms were then applied to the predictions to produce three different output streams. The predictions made by the classifier were used to identify when the user had started performing the movements, rather than the outputs, to make sure that each algorithm produced the same amount of outputs for the F_1-score calculations. The scores were compared to those calculated without Majority Vote to see the performance differences.

6.2 Results

All F_1-scores are presented with boxplots. Boxes in these plots ranges from the 25th to the 75th percentile of all the calculated scores. Whiskers reaches to the upper and lower adjacent values, and outliers are plotted with gray plus signs. The outliers are calculated with equation (B.1) that can be found in the appendix. Figure 6.2 shows the F_1-score, precision and recall calculated using the predictions that had been recorded during the real time tests. The F_1-scores are on average higher the more DoFs that are involved in the movements. The system performed better on combined movements than individual ones during the real time test.

The F_1-scores from the motion tests were compared to the simulated scores to make sure that the simulated results reflect those that are produced by the system during real time operation. Simulations were made using the same settings that were applied during the real time tests. The results can be seen in figure 6.3. The real time scores are on average slightly better than those produced offline but the results shows good agreement.

The system was further simulated using the three different Majority Vote based post-processing algorithms that were suggested in 5.1. The windowing scheme applied made sure that the average control delay would be kept the same as in the current system. The F_1-scores are compared with the simulated system without Majority Vote and the results can be seen in figure 6.4. The results show that the performance can be kept at the same level even though the analysis window has been reduced from 200 to 100 ms. The MV1 version of the algorithm could improve the performance of the system with individual movements, but the algorithm performed worse on combined movements. The two other versions, MV2 and MV3, performed similar to each other and could both be used to reduce the computational cost of the classifier.

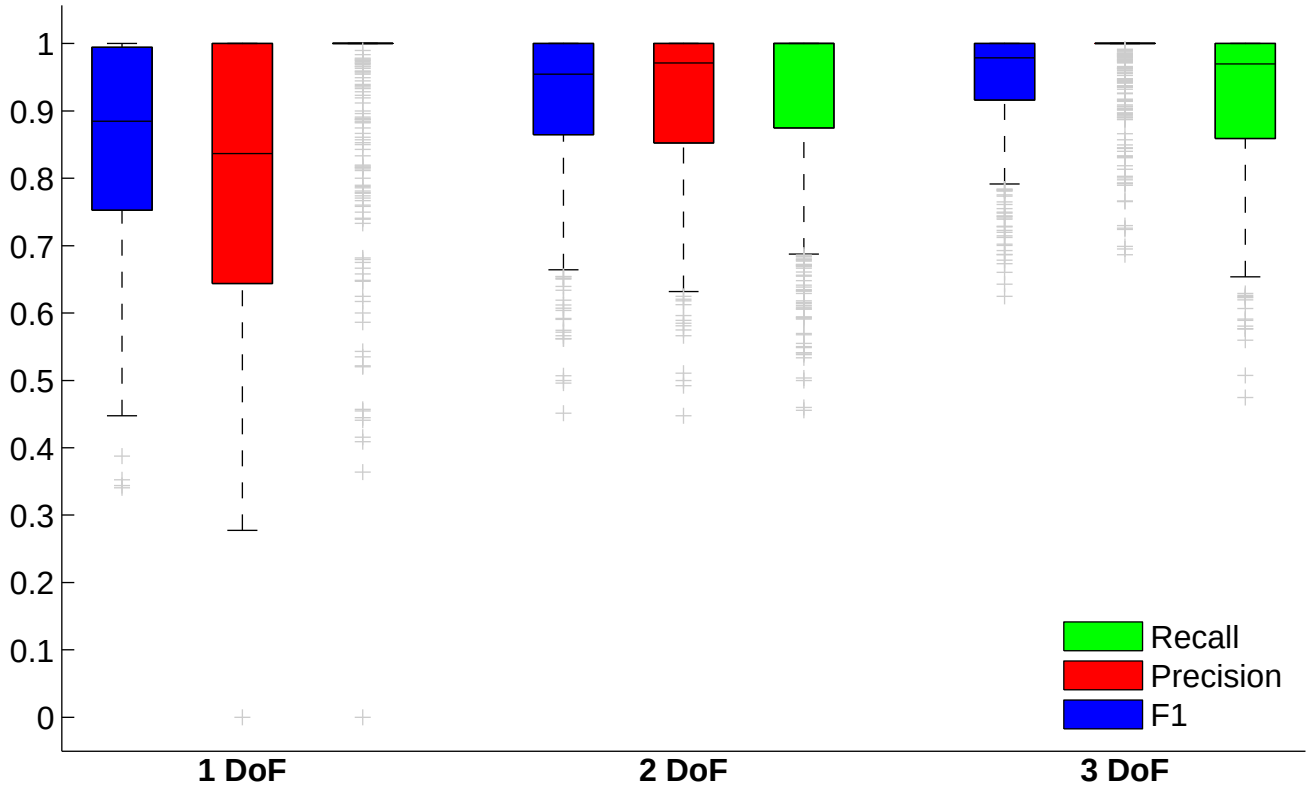


Figure 6.2: F_1 - scores, precision and recall calculated using the original motion test predictions. The F_1-scores are on average higher the more DoFs that are involved in the movements. The biggest problem in the system is with one DoF movements and the figure shows that the scores are brought down entirely by bad precision. The opposite is true for three DoF movements, were the score is limited by the bad recall.

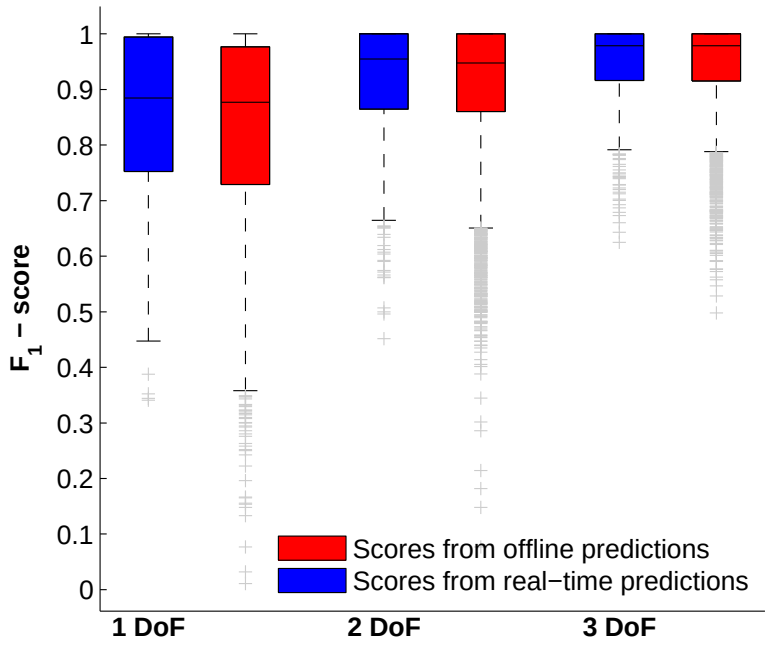


Figure 6.3: F_1-scores calculated from the motion tests compared to those that calculated offline using new classifiers. The results are very similar which suggests that the offline calculations can be used to compare performances between the different post-processing algorithms. The F_1-scores calculated from the motion tests are on average slightly higher which can be explained by the visual feedback the user gets during the real time tests. A contraction that is not performed the same way as during the recording session can be held by user because it still produces the desired results.

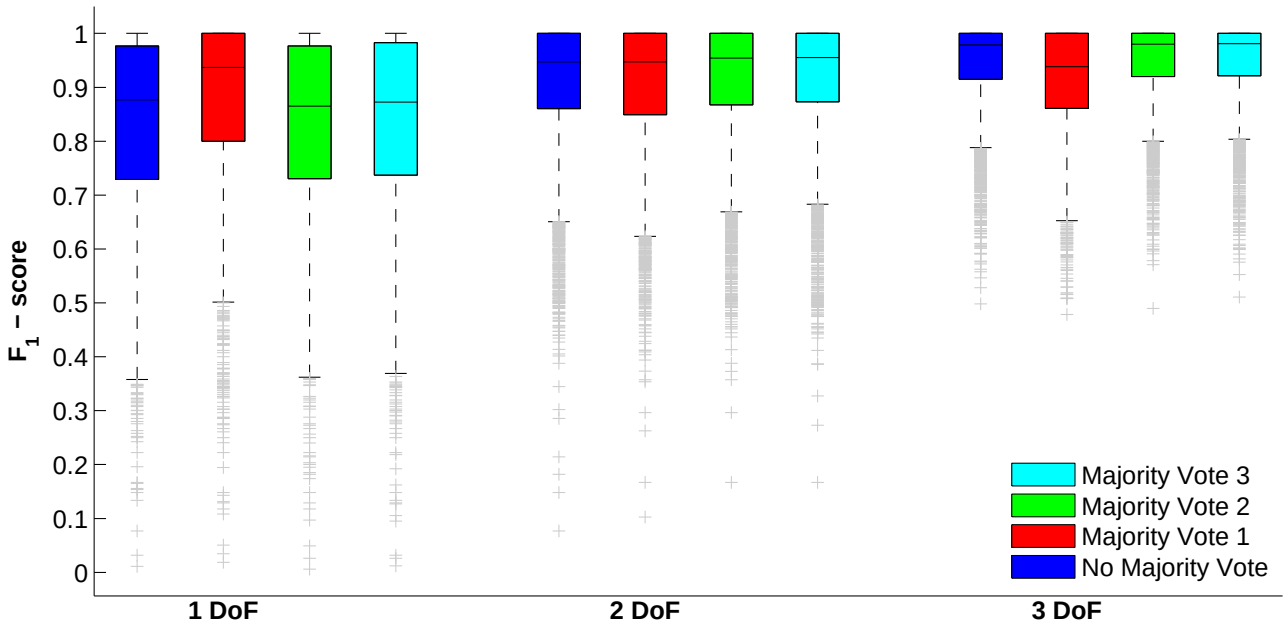


Figure 6.4: F_1-scores calculated for the different Majority Vote algorithms during the simulation compared to those calculated without post-processing. The values were calculated by training 10 new MLP classifiers for each trial of the motion test. The Majority Vote cases were calculated using an analysis windows of 100 ms and the no post-processing case was calculated using a 200 ms analysis windows, keeping the controller delay at 125 ms for both. The figure shows that there is not big improvement of the system when using any of Majority Vote strategies. The biggest difference that can be noticed is that MV1 improves the performance on one DoF movements, but reduces the performance on two and three DoF movements.

6.3 Discussion

The algorithms were compared using the F_1-score which is the harmonic mean of the precision and recall of the algorithm. This measurement was chosen because the results from chapter 4 showed that the system struggles with both false positives and false negatives (extra activations and interruptions). Calculations using the predictions streams from the real time experiments (figure 6.2) showed that the system performs better with combined than individual movements. The main problem with one and two DoF movements is the low precision, which in turn means that the prosthesis activates undesired movement. These results agree with those in chapter 4, where the main problem was found to be the extra activations. The biggest problem with three DoF movements are the recall of the algorithm, not all wanted movements are predicted. This is also consistent with previous results that showed that the biggest problems are due to the interruptions of desired movements.

The precision is calculated as the percentage of activated movements that are correct. This is not the same as the percent of correct predictions calculated in figure 4.3, predictions there were allowed to contain several movements. If a prediction includes two movements, one desired and one undesired, then 50% percent of activations are correct while 0% of the predictions. The F_1-score gives a better measurement of the performance of the system, and it can be seen that the score is on average higher than the classification accuracy.

The F_1-scores were calculated from each repetition of each movement that had been recorded during the motion tests. This was done to not favor bad repetitions as the repetitions had ended once the user had been able to perform the correct movement. Repetitions where the user easily could perform the movement lasted only for a few predictions while repetitions where the user had trouble performing the movement could last as long as 10 seconds. Since both the precision and the recall are percentage based measurements, calculations done on the individual repetitions will reflect the overall performance.

The simulated scores showed good agreement with those calculated from the real time predictions but the real time predictions generated slightly higher scores. The difference could be an effect of the user-system interaction that existed during the real-time experiments. Subjects had a visual feedback of the real-time predictions that were made, making it possible for subjects to adapt to the online classifiers.

Movements during the real time tests were not necessarily performed the same way as they were during the recording sessions, making it harder for the offline classifiers to produce as accurate predictions.

As the offline calculations produced similar results as the online, it was found unnecessary to redo the real-time experiments using the Majority Vote algorithms. The offline calculations were designed to have a controller delay similar to the current setup of the system, which operates roughly at the optimum of 125 ms. The time increment was held fixed at 50 ms during the offline calculations, making the perceived delay to depend only on number of votes and the size of the analysis window. The time increment was held fixed to match an existing limitation in the system, due to a slow communication between the data acquisition card and the computer, that currently does not allow decisions to made faster than every 50th ms. Using the same limit in the offline calculations made it possible to apply the Majority Vote algorithms directly to the system if they were found to improve the performance. The size of the analysis window was halved, from 200 to 100 ms, to keep the controller delay the same even though the algorithms used three predictions to produce an output. The F_1-scores were compared to those calculated without Majority Vote, and the results show very similar performances for all algorithms except the MV1 version. The MV1 version is basically the original algorithm, which was developed for individual control. It did, as expected, improve the performance on individual movements but performed worse on combined movements. The algorithm can be used to suppress extra activations, but same effect could be achieved by increasing the MLP thresholds. The other two algorithms performed very similar, either of them could be used for simultaneous control.

The fact that same performance, with the same controller delay, could be achieved using smaller analysis windows is the main result. Smaller analysis windows means that less processing power is needed for feature extraction (Englehart and Hudgins, 2003). The results might be important when implementing the system in a real prosthetic device that might be limited in its processing power. The performance might be increased further if the time increment would have been optimized to match the processing time τ of the analysis windows. This will however have to be investigated for each system specifically as the time increment is highly platform dependent. The algorithms will not have any effect on the current system as the same performance was achieved without the Majority Vote. The current system can easily extract the desired features from 200 ms analysis windows and make a prediction within the 50 ms time increment.

6.4 Conclusions

Calculations using offline predictions were found to produce similar results as calculations using the online predictions. The offline calculations were found appropriate to compare the Majority Vote algorithms, and the results show that the MV2 and MV3 algorithms can be used to achieve the same performance with smaller analysis windows. Depending on the complexity of the signal features that are extracted from the analysis windows, the reduced window size can be used to reduce the computational cost of the classification which might be important when implementing the system in a real prosthetic device. The MV1 version showed an increased performance on individual movements but a decreased performance on combined movements. The algorithm could be used to suppress extra activations made by the system, making it easier to control individual movement and harder to control combined. Same effect can however be achieved by increasing the MLP thresholds.

Comparing Decision-Based Velocity Ramp Algorithms

Two versions of the Decision-Based Velocity Ramp were proposed in 5 to be used with the simultaneous control system described in section 3.2. The performance of these algorithms can not be compared using the offline calculations that were used in chapter 6, because both of these algorithms output every prediction made by the classifier and no difference would be noticed. The two algorithms were instead tested in a series of target achievement tests (TAC-tests), described in section 2.4.1, to compare their real time performance. Two different experiments were conducted, one comparing the traditional implementation of the ramp against not using ramp at all, and one comparing it against the modified version.

7.1 Method

Two target achievement experiments were conducted to test the performance of the decision-based velocity ramp algorithms. The first experiment compared the performance of the system with and without the ramp. Since the system had no other way than ramp to alternate the output speed of movements, this experiment was carried out using a slow movement speed of the virtual limb. The slow speed had to be used to make sure that targets could be completed without the ramp as well. The second experiment compared the performances between the two suggested versions of the ramp algorithm, and was conducted using a faster movement speed of the virtual limb.

Both experiments were conducted using BioPatRec, which includes a version of the TAC-test. Each experiment consisted of three sub experiments where targets were constructed using 1, 2 and 3 DoF movements respectively. All targets were constructed by moving the limb 40 degrees from the resting position, using each of the individual movements. This allowed the user to reach all targets using only one of the 26 movement combinations per repetition (table 3.1). If the system produced any errors, or the user overshot the target, extra compensation had to be made to tweak the position of the virtual limb to match the target. The myoelectric control system described in section 3.2 was used in both experiments, and all subjects were experienced users of said system.

During the experiments, eight pairs of electrodes were placed around the most proximal third of the dominant forearm. Subjects went through a recording session of the 26 combined movements, guided by the software (BioPatRec). The classifier was tested in making real time predictions, and if it performed badly the subject underwent another recording session. The TAC tests were not started until the classifier was able to predict the different movements satisfactory.

Each TAC test consisted of six repetitions of each target, six targets for the first TAC-test (one DoF), twelve for the second (two DoF) and eight for the last (three DoF). Targets were presented in random order during the tests and all the different targets were performed before moving on to the next repetition. The TAC-tests were performed in rising order in the number of DoFs used to create the targets. Subjects first performed the test with one DoF targets, then with two DoF targets and lastly with three DoF targets. The first experiment was divided into two sessions where subjects first conducted all three TAC-test without the Ramp and then with the Ramp. Subjects were allowed to rest between the two sessions, and if the classifier worked badly when they returned they underwent another recording session.

The second experiment, where the two versions of the decision-based velocity ramp algorithm were compared, was conducted in one sitting. Each TAC-test was performed two times in a row, applying

the two algorithms in random order. The subjects did not know which one of the algorithms that was applied during which test to remove any biasing. The same algorithm was kept throughout an entire TAC-test so subjects could adapt to any differences between the two algorithms. Once the subject had completed the two one DoF TAC-tests, the experiment moved on to the two two DoF TAC-tests and so on. This approach made sure that the two algorithms were tested with the same conditions.

Targets were set to have an allowance of ±5 degrees during both experiments. A target was considered completed once the subject had stayed within the target for 1 second. The time before a repetition was considered uncompleted differed between three TAC-tests, depending on the number of DoFs involved in the targets. Subjects had 20, 25 and 30 seconds to complete the one, two and three DoF targets respectively.

All predictions during the experiments were made using single MLP classifiers. The classifiers were trained using the standard methods in BioPatRec, described in section 3.2. The BioPatRec "top four" feature set was used to make the classification, a feature set that consists of four time-domain features; mean absolute value, waveform length, number of zero crossings and the number of slope sign changes. The electromyogram was windowed using an overlapped windowing scheme with 200 ms analysis window and the time increment differed between the two experiments. The first experiment was conducted using the standard implementation of the TAC-test in BioPatRec, which requires a time increment of at least 100 ms. This is due to the fact that the distance between the target posture and virtual limb is measured inside the virtual environment, which slows down the execution time of the program. The TAC-test was modified for the second experiment to allow the time increment to be decreased to 50 ms, which is the limitation imposed by the communication between the data acquisition card and the computer.

The virtual limb was set to move at a maximum speed of 20 and 40 degrees per seconds respectively for the two experiments. The slower speed was used when comparing the traditional ramp with the no ramp system, and the faster speed was used when comparing the two versions of the velocity ramp. The algorithms were set to have a *ramp length* of 10 predictions, which translates to a 1 and 0.5 second acceleration for the two experiments. The difference is due to the fact that different time increments were used for the two experiments, the system made predictions faster in the second one. The *down count* was set to 2 throughout the experiments and the modified version was set to not penalize the output speed of the first two interruptions.

Two different performance metrics were extracted from the TAC-tests,

Completion Time
: The time it takes for the movement to be considered completed, time spent in the target posture is not counted. Timer is started from first non-rest prediction made by the classifier.

Path Efficiency
: Fraction between the length of a perfect path to the target and the traveled path. Any movements inside the target is not accounted for. $Path\ Efficiency = \frac{perfect\ path}{traveled\ path}$, a perfect path has an efficiency of 1 and all other paths have lower efficiencies.

The results were tested for statistical significance using a two-sided Wilcoxon Signed-Rank test, a non-parametric test that has been proven to be appropriate when comparing classifiers on common data sets (Demšar, 2006). The mean completion times and path efficiencies of the six repetitions of each movement (per subject) were compared between the two conditions existing in the experiments, i.e. ramp against no ramp in the first experiment and ramp against modified ramp in the second.

The first experiment was performed by four different subjects, three of them also conducted the second one. The experiments were performed at different times and the results from the first should not affect the second.

7.2 Results

Experiment 1

The first experiment compared the performance of the system with and without the velocity ramp algorithm. The traditional implementation of the algorithm was used for this purpose and the results

show that the system can be improved using the algorithm. A one second acceleration was added to the virtual limb and the maximum speed was fixed at 20 degrees per second during this experiment. The system without the ramp moved constantly at maximum speed and could therefor reach targets faster. The results are presented using box plots created from the completed repetitions during the experiment. Boxes in these plots ranges from the 25th to the 75th percentile and whiskers reaches to the upper and lower adjacent values. Outliers are plotted as gray plus signs and are calculated with equation (B.1) in the appendix, median values are shown by horizontal lines in the boxes. The completion times from the experiment are presented in figure 7.1 and the path efficiencies in figure 7.2.

Four subjects performed the experiment, conducting a total of 624 repetitions of the different movements. Seven of these repetitions were failed when the ramp was used and ten without, meaning that subjects could complete most of the targets using both systems. As most of the repetitions were completed in both cases, the box plot figures are representative of the over all performances of the systems. The results show that one DoF targets were completed faster when the ramp was applied while three DoF targets were completed faster without it. The efficiencies of the paths traveled to the target postures were increased for all targets when the ramp was used, suggesting that the limb were easier to control with the ramp.

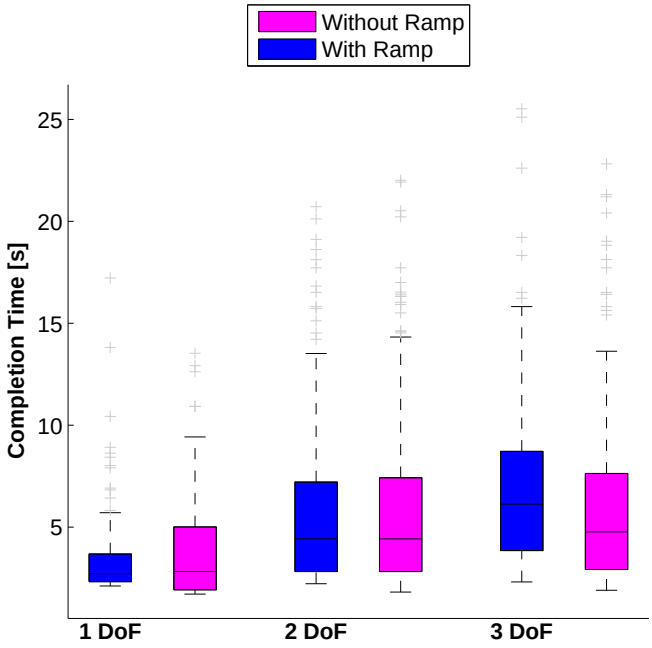


Figure 7.1: Completion times observed during experiment one. The plot was constructed using only the completed repetitions. The figure shows that one DoF targets could be completed faster with the ramp, while 2 DoF targets were completed equally fast without. Three DoF targets were completed faster without the ramp. This can be explained by the fact that the ramp algorithm penalizes the output speeds of movements as soon as they are missing in the predictions. As the system has proven to struggle with interruptions during three DoF movements, the ramp algorithm would punish the output speeds of desired movements in this case.

The cumulative distribution of the completion times can be seen in figure 7.3. The distribution describes the percent of repetitions that could be completed within a certain time and includes both completed and failed repetitions. It is clear from this picture that the acceleration that is added by the ramp increases the limit on how fast the subjects could complete the targets. The results are consistent with those presented in the box plots and show that the two systems performed similar on two DoF targets. The system with the ramp performed better on one DoF targets and the system without performed better on three DoF targets.

The complementary cumulative distribution of path efficiencies are presented in figure 7.4. The complementary cumulative distribution shows the percentage of repetitions that had larger path efficiencies than the corresponding x-value. The complementary cumulative distribution is used as the path efficiencies are desired to be as large as possible in contrast to the completion times that are desired to be as low as possible. The results show that repetitions performed with the ramp had larger path

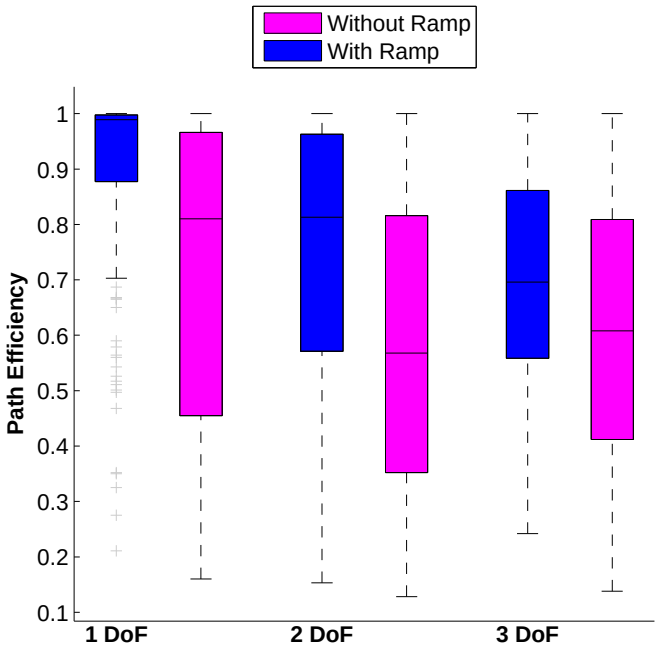


Figure 7.2: Boxplot of the path efficiencies observed during experiment one constructed using the completed repetitions. All types of targets were reached with higher path efficiencies when the ramp was used. The virtual limb followed the users intentions better with the ramp and did not do as many unexpected movements as it did without. All targets could be reached by using only one of the 26 movement combinations, making the optimal paths easy to figure out by the subjects. Paths were less efficient the more DoFs that were included in the targets which could be explained by the fact that these targets were further away from the resting position. A path efficiency of 1 means that the user reached the target with the optimal path and is the best that can be achieved.

DoF	1	2	3
Completion Time	0.130	0.448	0.130
Path Efficiency	0.001	0.000	0.004

Table 7.1: P-values calculated in a paired two-sided Wilcoxon Signed Rank Test. The pairs for the calculation were created by averaging the performance metrics per subject and movement, comparing the performances between the two systems that were used in the experiment.

efficiencies than those performed without. The algorithm helped subjects to take a more direct path to the targets which explains why the two systems could complete targets equally fast while one of them was moving faster.

The results were tested for statistical significance using a two-sided Wilcoxon signed rank test by pairing the average performance metrics per subject and movement using the two systems. The corresponding p-values are presented in table 7.1 and shows a statistical significance in the path efficiencies for all three types of targets ($p < 0.05$). No statistical significance could be found between the completion times.

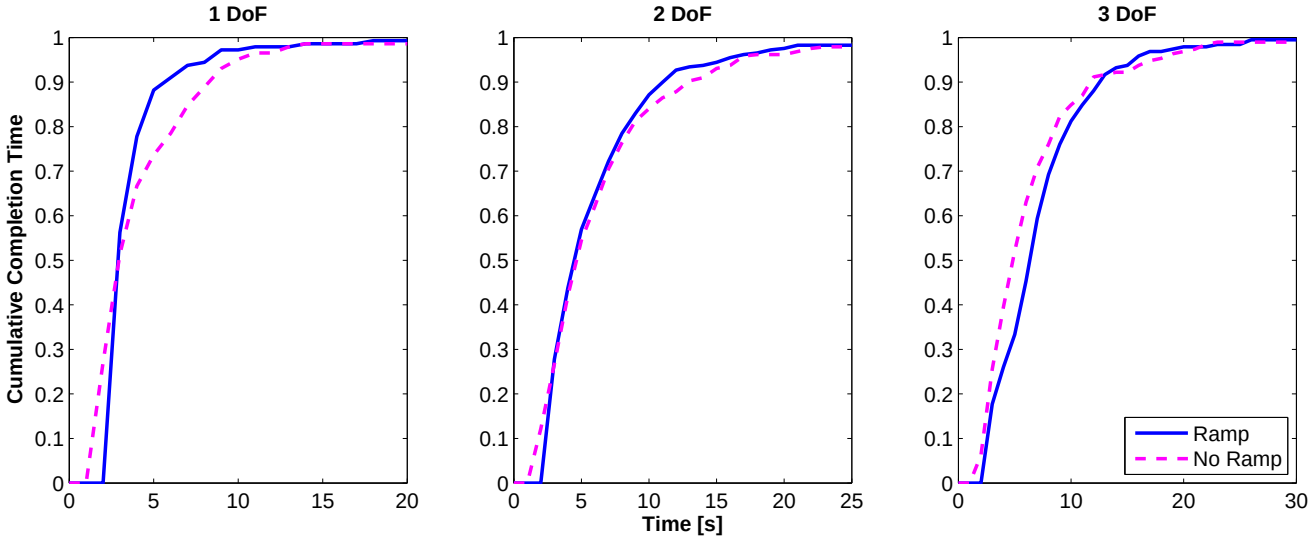


Figure 7.3: The cumulative completion times from the first experiment, comparing the traditional ramp against no ramp. The results are separated for the three different TAC-test that tested one, two and three DoF targets. The plots were constructed using all repetitions from all four subjects. The figure shows that the ramp helped to complete more one DoF targets faster.

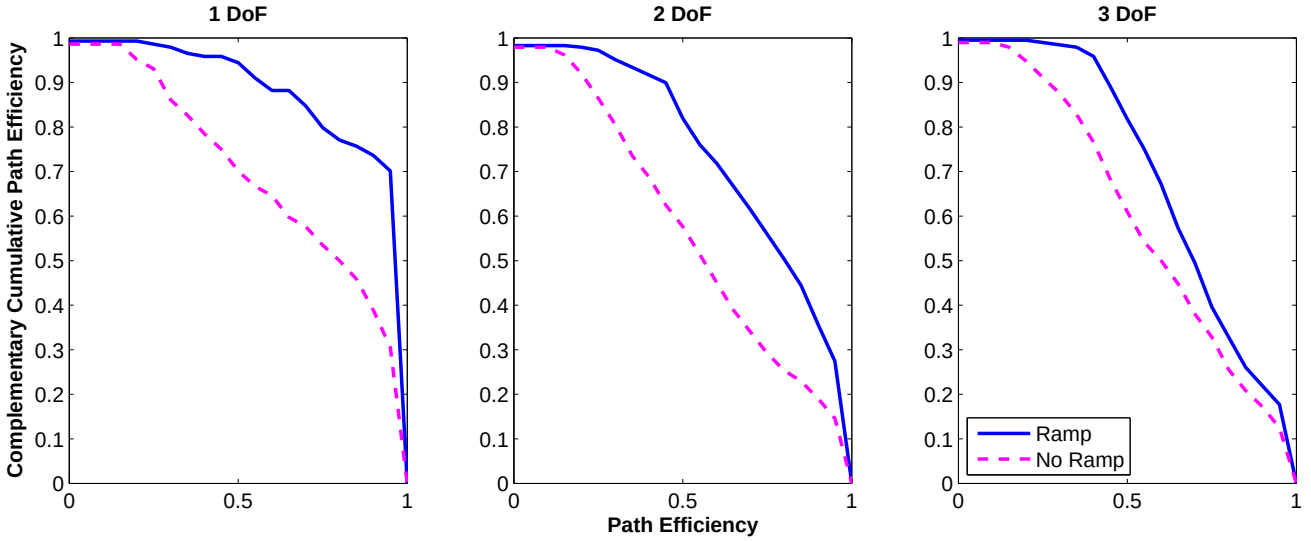


Figure 7.4: The complementary cumulative path efficiency from experiment one. The traditional ramp algorithm was compared against a no algorithm control case using the target achievement test. The x-axis shows the path efficiency that ranges from 0 to 1 and the y-axis shows the percent of repetitions that were performed with greater path efficiency than the corresponding x-value. The plot was constructed using all repetitions from all subjects and is not an average over subjects. The figure shows that the system using the ramp could complete targets with higher path efficiencies than the system without. Particularly can it bee seen that more than 70% of the repetitions using ramp had an efficiency close to 1 with one DoF targets. Correspondingly did only 30% of the repetitions without the ramp reach the same efficiency.

Experiment 2

The second experiment compared the traditional implementation of the ramp with the modified version that was presented in section 5.3. The results show that the traditional implementation of the Decision-Based Velocity Ramp performed better than the modified version when targets were constructed from one and two DoF movements. Subjects reached more targets when using the traditional ramp, and one DoF targets were reached more efficiently ($p < 0.05$). The experiment was carried out by three of the subjects from experiment one, conducting a total of 468 repetitions of the different targets.

The TAC-test in BioPatRec was modified for this experiment, to allow the system to operate with a time increment of 50 ms. The modification allowed the controller delay to reduced from 150 to 125 ms, excluding the processing time. The *ramp length* were kept at 10 predictions for both algorithm, meaning that the acceleration time was halved from 1 second that were used in experiment one to 0.5 seconds. The maximum speed of the virtual limb was increased to 40 degrees per second for this experiment to better mimic a real prosthetic device. The subjects managed to complete all but seven repetitions using the traditional ramp, and failed 25 using the modified version, suggesting that the traditional ramp is to prefer.

The completion times from the completed repetitions are presented with a box plot in figure 7.5. The results show that one and two DoF targets on average were completed faster with the traditional algorithm, while three DoF targets were completed faster with the modified version. The path efficiencies during these repetitions are presented in figure 7.6. The results show that one DoF targets were reached more effectively with the traditional ramp and three DoF targets with the modified version. Two DoF targets were completed with roughly the same efficiencies using both algorithms.

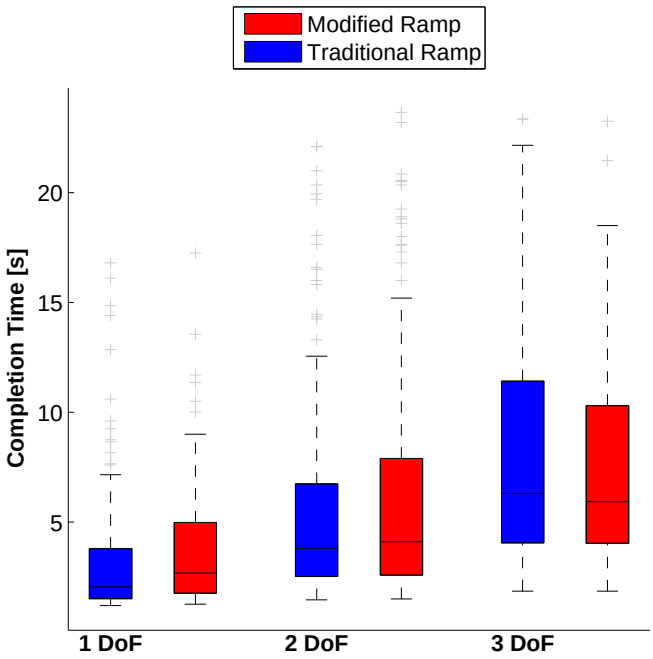


Figure 7.5: Completion times observed during experiment two. Only completed repetitions were used to create the figure which means that fewer data points were included for the modified algorithm. The figure shows that the traditional ramp was able to complete one and two DoF targets faster than the modified version. Three DoF targets could be completed faster using the modified version. The modified algorithm does not punish the output speeds the first interruptions which allows it to reach the three DoF targets faster.

Figure 7.7 shows the cumulative distribution of the completion times from experiment two. This plot includes all repetitions, both completed and failed. The figure shows that fewer repetitions were completed using the modified algorithm than the traditional. The fact that three DoF targets on average were completed faster using the modified algorithm cannot compensate for the increased number of failed repetitions. The cumulative distributions for the two systems follow each other well in the three DoF case, meaning that the overall performance of the two algorithms were similar to each other. The traditional algorithm is currently the one that should be used as it allowed the subjects to complete more movements.

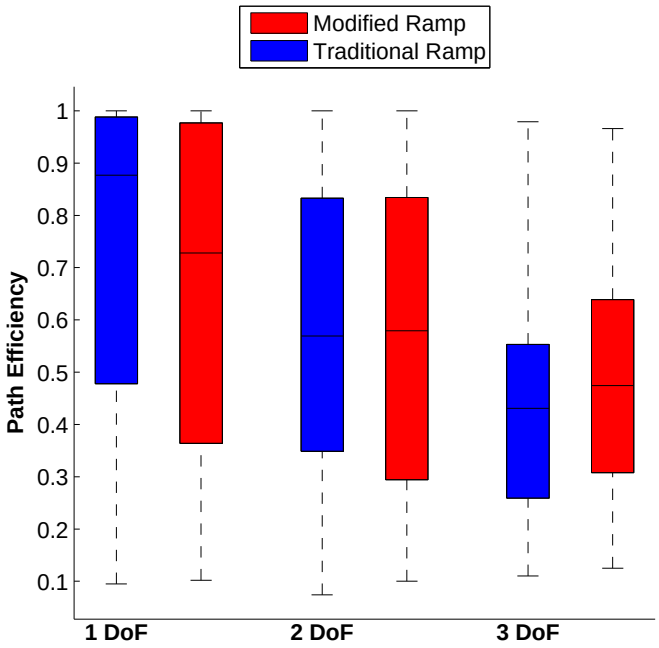


Figure 7.6: Path efficiencies observed during experiment two. Only completed repetitions were used when creating the figure meaning that fewer data points were used for the modified algorithm. The figure shows that the traditional ramp performed better than the modified on one DoF targets, and the other way around with three DoF targets. It is expected that the modified version should perform better with three DoF targets as it was designed to allow some interruptions of the desired movements before it punished the output speeds.

The complementary cumulative distribution of path efficiencies from the second experiment are presented in figure 7.8. The results are consistent with previous findings, one DoF targets were completed more efficiently using the traditional ramp. The other two types of targets were completed with similar performances. The fact that the path efficiencies were improved for three DoF targets with the modified algorithm cannot compensate the fact that fewer repetitions were completed.

The average distances between the virtual limb and the targets during the second experiment are presented in figure 7.9. The distances were calculated from both completed and failed repetitions. The results show that subjects were able to quickly close the distance to the target postures. The virtual limb could on average be positioned roughly ten degrees from the desired postures within the first couple of seconds. Subjects then had to tweak the virtual limb using small movements to actually reach the targets, which proved to be hardest part.

The traditional algorithm reached one and two DoF targets on average faster than the modified algorithm which is consistent with the results in figures 7.7 and 7.5. Both small and large distances were closed faster using the traditional ramp with one and two DoF targets. The modified algorithm could close large distances faster with three DoF targets, but the improved performance was compensated by a worse performance on smaller distances.

The median distance to the target during the failed repetitions using the modified ramp are presented in figure 7.11. There were eight, nine and eight failed repetitions in the one, two and three DoF cases respectively. To avoid repetitions considerably different than others affecting the plot, the median distances were used instead of the average. The results shows that subjects failed the repetitions because the postures could not be tweaked to match the target. The distance to the target could be decreased to approximately ten degrees even in the failed repetitions. It is first when subjects tried to tweak the posture, distances started to increase again. This suggests that subjects could not select the desired movements correctly, making the targets hard to reach.

The results were tested for statistical significance the same way as in experiment one, using a two-sided Wilcoxon signed rank test. All p-values calculated by the test are presented in table 7.2. The only case that produced a statistically significance between the two algorithms was the one DoF case. The path efficiencies were found to differ significantly ($p < 0.05$), and figure 7.8 shows that it is in the favor of the traditional algorithm. P-values were calculated using the mean value of the completed repetitions,

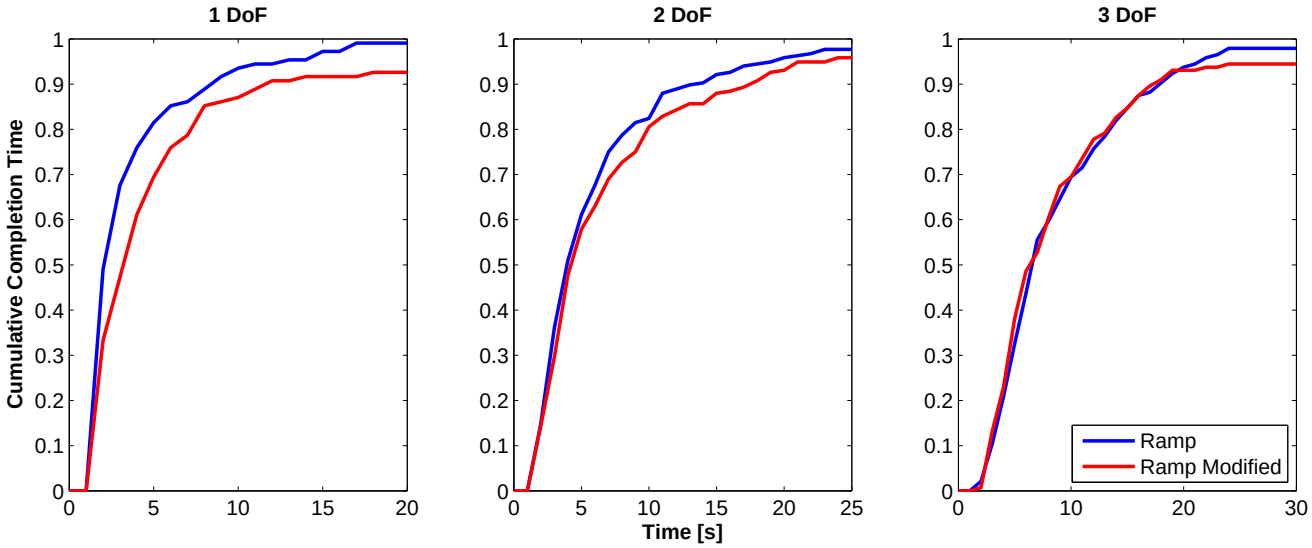


Figure 7.7: The cumulative completion time for the two proposed ramp algorithms separated in the cases with one, two and three DoF targets. The plot was constructed using all repetitions from all subjects and is not an average. The figure shows that more repetitions were completed using the traditional version of the ramp for all three cases. Subjects reached one and two DoF targets faster when they were using the traditional ramp. The fact that the distributions follow each other in the three DoF case means that the improved completion times with the modified algorithm cannot compensate the that fewer repetitions were completed.

DoF	1	2	3
Completion Time	0.094	0.1767	0.4073
Path Efficiency	0.039	0.338	0.331

Table 7.2: P-Values from the second experiment calculated using a two-sided Wilcoxon signed rank test. The performance metrics, completion time and path efficiency, were averaged per subject and movement and paired between the two versions of the algorithm. The only value that shows statistical significance is the path efficiency with one DoF movements ($p < 0.05$).

averaged across subjects and movements. All failed repetitions were left out of the calculation as they could not be used to extract either completion times or path efficiencies.

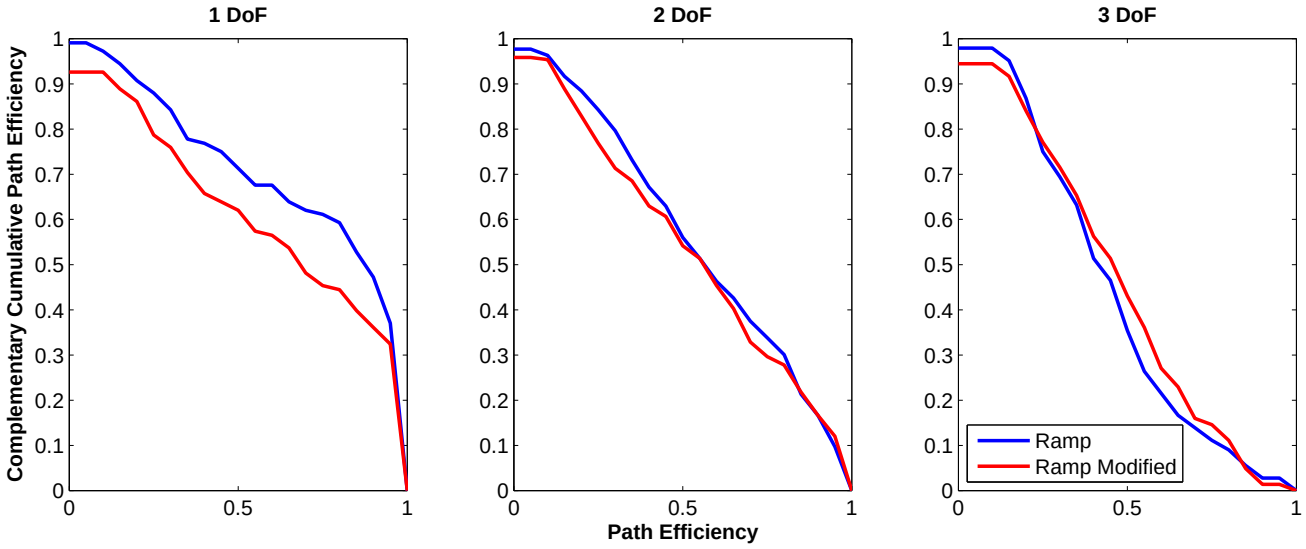


Figure 7.8: The complementary cumulative distribution of path efficiencies from experiment two. The x-axis shows the path efficiency that ranges from 0 to 1 and the y-axis shows the percent of completed targets that had greater path efficiency than corresponding x-value. The plot was constructed using all repetitions from all subjects and is not an average. The figure shows that one DoF targets were completed with higher path efficiencies when subjects were using the traditional ramp. The difference were also found to be statistically significant in a two-sided Wilcoxon signed rank test ($p < 0.05$). The two other cases were performed with similar efficiencies even though three DoF targets on average were reached more efficiently with the modified ramp.

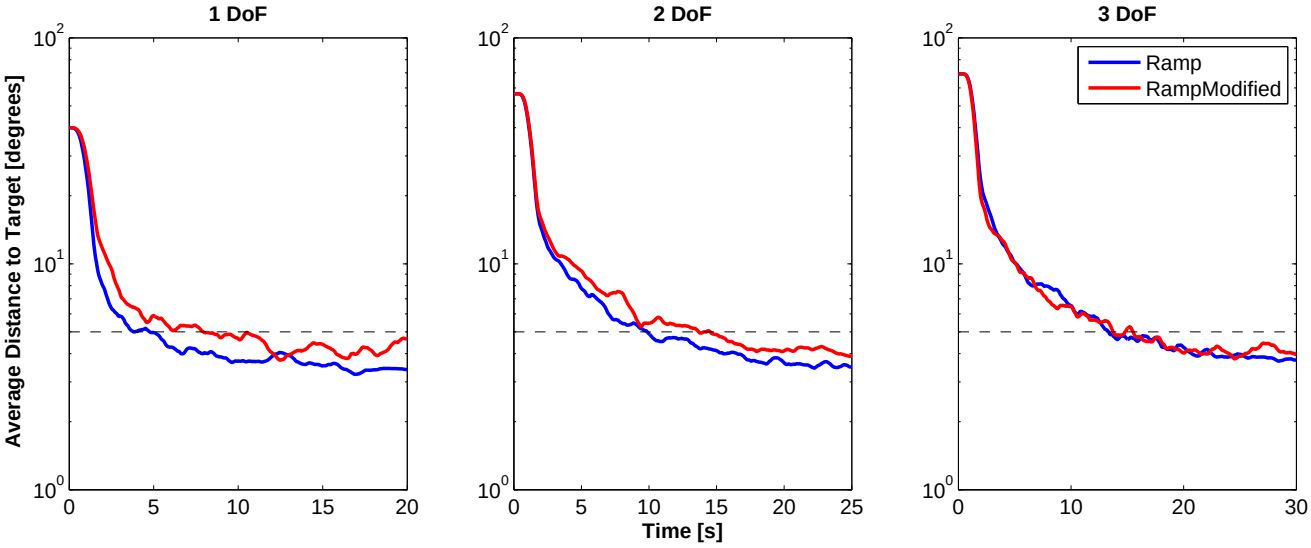


Figure 7.9: Average distance to target as a function of time from experiment two. Notice that the y-axis has a logarithmic scale as the distances quickly decreased initially. The plot was created by averaging all repetitions from all subjects, including the failed repetitions. Targets were given a five degree allowance during the experiment which is illustrated by the black line in the figure. If the distance between the virtual limb and the target posture was smaller than five degrees, the target was considered to be completed. Subjects were able to quickly position the virtual limb approximately ten degrees from the target postures. The distances decreases slower after that, which is an effect of the subjects stopping the virtual limb to tweak it into the target.

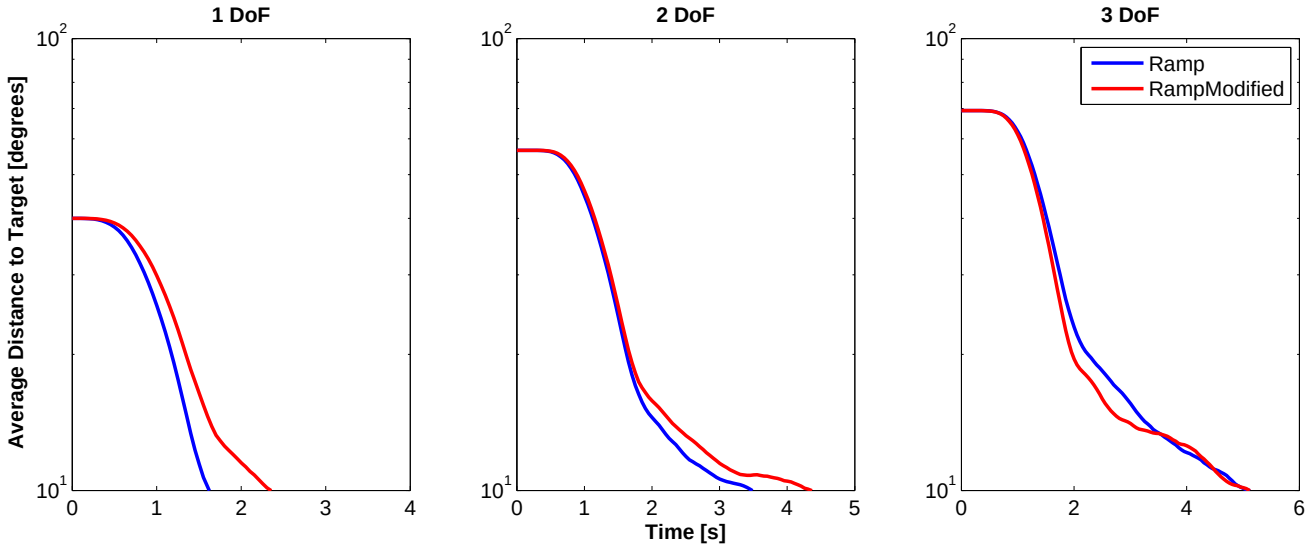


Figure 7.10: An enlarged version of the initial portion of figure 7.9. The figure shows that the traditional ramp closes the large distances to one DoF targets faster than the modified version while the other way around is true for three DoF targets. The bump on the curves are results from when the user stops the virtual limb to tweak the position. As the limb is stopped, the algorithms punishes the movement speeds and the limb has to be accelerated again making the distance decrease slower.

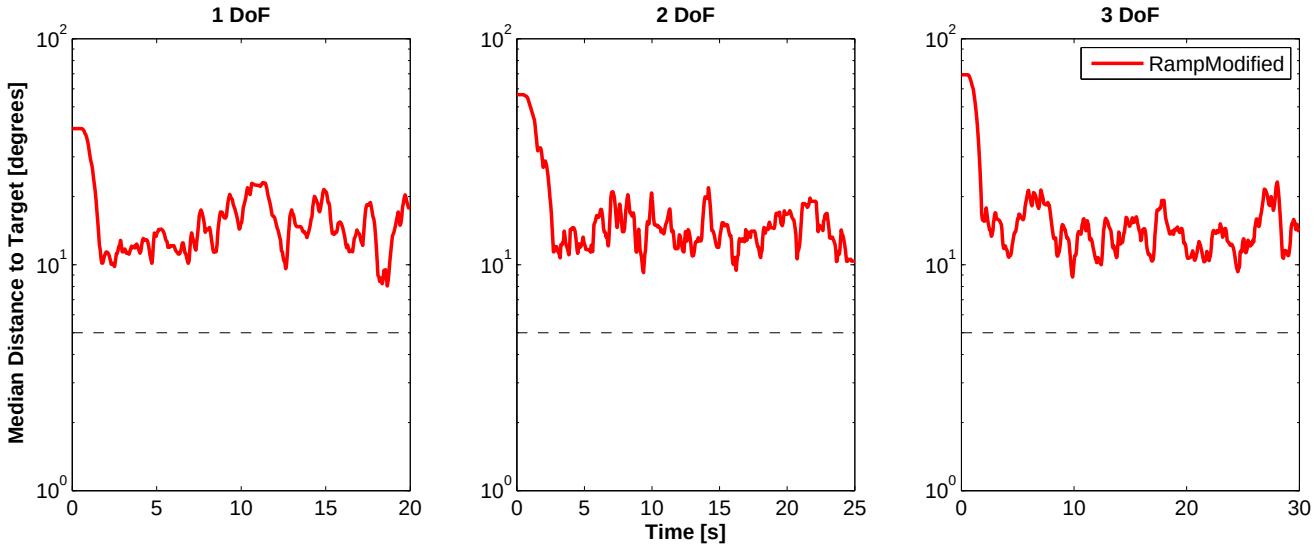


Figure 7.11: Median of the distances between target and virtual limb, calculated using only failed repetitions with the modified ramp. Eight, nine and eight repetitions were failed in the one, two and three DoF cases respectively. The figure shows that subjects easily could reach a posture approximately ten degrees from the targets even during the failed repetitions. The issue is that subjects could not tweak the limb into the target postures, highlighting a problem with the control during the on- and offset of motions.

7.3 Discussion

Both experiments involved experienced users of the myoelectric control system. This is important as it has been shown that experienced users can control myoelectric systems better than novice users (Bunderson and Kuiken, 2012). The fact that all subjects were experienced users means that the results rely less on the classifiers and more on the post-processing algorithms. The users were also experienced with the BioPatRec interface of the TAC-test which made them less prone to improve their results during the experiments.

The online classification can for several reasons be bad, even if the offline measurement in BioPatRec shows a good accuracy. For example, the electrodes require some time to create a stable interface with the skin, the user needs to perform all movements correctly during the recording session, there can be loose connections between the electrodes and the amplifier et cetera. Subjects were asked to test the system in real-time before starting the tests to make sure that the online classification was working good. If a subject felt that the classifier was working poorly, she or he underwent another recording session. The definition of working poorly was up to the subjects themselves, but as they were experienced users of the system they knew what performance to expect. This would further reduce the impact of the classifier and make the results rely more on the different post-processing algorithms that were used.

A bad classifier should impact the results from the first experiment more as two of the subjects choose to change classifier between the two sittings. They did therefor use two different classifiers for the ramp and the non-ramp systems. The other two subjects choose to perform the first experiment without a break, keeping the same classifier for all six TAC-test. The performance of the classifier can also degrade over time due muscle fatigue, electrode shifting or any other time related effect. As the non-ramp system was tested first and the ramp later, any time related degradation should have affected the results of the ramp system more. However, the ramp system still proved to be the better one.

In the second experiment, all TAC-tests were performed in one sitting by all subjects. The system was kept the same throughout the entire experiment, except for the version of the ramp algorithm that was applied. A bad classifier should not influence the results as much as experiment one as subjects used the same classifier for all tests. Any negative impact due to a bad classifier should be noted in the results of both algorithms. The results should also be less effected by transient changes in the classifier performance as the two algorithms were alternated throughout the experiment.

The systems were compared in both experiments using a two-sided Wilcoxon Signed Rank test. The test has proven to be appropriate when comparing classifiers on common data sets (Demšar, 2006). Pairs for the test were constructed by the averaging the performance metrics per movement and subject. Averages were calculated using only the completed repetitions as uncompleted repetitions do not have a completion time, and the path efficiency does not make any sense if the targets never were reached. Two of the subjects chose to change classifier between the two sittings in experiment one, meaning that the two classifiers might have had different performances on different movements. This would affect the statistical analysis as the performances were compared between subjects and movements. The completion times of one DoF targets could perhaps be found to be statistically significant if the pairs were constructed in another way, or if more subjects performed the test. Averages of the metrics has to be used as the individual repetitions are unrepresentative of the performance of the system.

The first experiment showed that the performance of the system can be increased by using the traditional ramp algorithm. All targets could be reached more directly which proves that the system better responded to the subjects intentions. In particular could subjects control one DoF movements much better using the ramp. The extra activations that system struggles with during these movements, figure 4.4, could effectively be suppressed by the ramp algorithm. The fact that target could be completed equally fast with and without the ramp is because the system could move faster without the ramp. Without the acceleration subjects could compensate any unintended movements the system made faster than they could with the acceleration. However, such a situation is not desired with a real prosthetic device as the user cannot control how the device will behave. The added acceleration of one second was based on the results found by Simon et al. (2011*a*). The completion times could perhaps have been improved further by decreasing the acceleration allowing subjects to move the virtual limb more quickly.

The second experiment showed that the traditional implementation of the ramp algorithm is to prefer over the modified version. Subjects could complete more repetitions with the traditional ramp which also proved to produce better path efficiencies with one DoF targets ($p < 0.05$). The modified algorithm

allowed three DoF targets to be reached faster and with higher efficiency, but the increased performance could not compensate the increased number of failed repetitions.

The traditional and modified algorithms differ in two aspects, the modified version allow some interruptions before penalizing the output speeds and it resets the counters when the rest-class is predicted. The first modification would allow the algorithm to perform better in cases when interruptions are the biggest problem, such as three with DoF movements. This trend can be noted in figures 7.6 and 7.5 as repetitions of three DoF targets were performed on average both faster and with higher efficiencies using the modified algorithm. The differences between the algorithms were however not found to be statistically significant, and the improvement could not compensate the fact that less repetitions could be completed.

The modified algorithm was also found to perform worse than the traditional with one DoF targets. This means that at least one of the two modifications were affecting the system negatively. The system has been shown to produce a lot of undesired activations during individual movements but it is not obvious that the modified version should perform worse in that situation. The modified algorithm still only outputs the predicted movements and as long as the misclassifications do not occur in an on/off fashion, the output speeds should be the same using either of the two algorithms.

Failed repetitions with the modified algorithm showed that subjects could position the virtual limb approximately ten degrees from the targets just as in the completed repetitions. The problem was found to be when subjects tried to tweak the limb to match the target, figure 7.11. Previous reports has shown that the classification accuracy during the on and offset of movements can be greatly reduced if no precautions is made (Lorrain et al., 2011). A reduced classification accuracy could possibly explain the problem subjects had trying to tweak the virtual limb. Tweaking is done by activating and deactivating movements within a short time frame, if wrong movements are predicted the virtual limb will move away from the target instead of towards. These additional misclassifications seem to have affected the modified algorithm more than the traditional. This could partly be due to fact that the counters are reset once the rest-class is predicted. Both misclassified and desired movements will make an equal impact on the output as all movements are starting from zero speed. Another possible explanation could be that subjects might have favored individual movements when they tweaked the posture. By adjusting the limb in one DoF at a time it is easier to see what adjustments are needed to reach the target. As one DoF targets showed a worse performance with the modified algorithm, this tweaking behavior would make the tweaking harder with the modified algorithm.

An interesting result is that larger completion times were noted in the second experiment than in the first, even though the same targets were used and subjects could move faster in the second experiment. As subjects could move faster they should have been able to reach the targets faster as well. On average could the large distances be closed very fast in the second experiment, figure 7.10, showing again that the problem is with the tweaking of the virtual limb. The extra speed made it harder for subjects to stop within the targets, possibly due to added classification errors during the offset of the movements.

The modified algorithm should not be disregarded totally. If the MLP thresholds are increased, the system would be less prone to make extra activations but it would interrupt more of the desired movements. In such a case the modified algorithm might perform better than the traditional as it allows for some interruption. The algorithm could further be fitted with a threshold that made sure that interruptions only are compensated for movements that have ramp up to a certain speed. This would make any spurious activations during the onset of movements to have less impact on the output, as the algorithm could force them to start over at zero speed again. The evaluation of such modifications is left for future work and it is concluded that the current system can benefit most from the traditional ramp.

7.4 Conclusions

Two experiments were conducted to evaluate the use of the Decision-Based Velocity Ramp for simultaneous myoelectric control using BioPatRec. Both experiments compared two different setups of the control system by having the subjects conduct six different target achievements test, two tests with one DoF targets, two with two DoF targets and two with three DoF targets. The first experiment was carried out by four experienced users and compared the system with and without the traditional implementation of the velocity ramp. The results showed that all types of targets could be completed with significantly

better efficiencies ($p < 0.05$) when the ramp was used. One DoF targets were completed faster using the ramp and three DoF targets faster without, but the results were not found statistically significant in a two-sided Wilcoxon signed rank test. The system should use the velocity ramp as it increases the path efficiency, making the virtual limb respond better to the indentions.

The second experiment compared the traditional version of the ramp against the modified version that was suggested in section 5.3. The results showed that more repetitions could be completed using the traditional ramp. Three DoF targets were on average completed faster and with higher efficiency using the modified algorithm, but the improvements were neither found to be statistically significant nor big enough compensate for the increased number of failed repetitions. The biggest issue when trying to complete targets is the final tweaking of the virtual limb. The tweaking was harder to do with the modified version which means that misclassifications during the on- and offset of movements are affecting the modified ramp more. The only statistically significant difference between the two algorithms was found with one DoF targets, which could be reached more efficiently using the traditional ramp ($p < 0.05$).

The control system can definitely be improved using a Decision-Based Velocity Ramp. The current system should use the traditional version as it showed best performance. However, if the system is further developed by e.g. increasing the MLP thresholds, the modified algorithm should be reevaluated. The modified algorithm could further be developed by implementing a threshold that only compensates interruptions of movements that has a ramped up to certain speed. This could prevent spurious activation to impact the output to the extent they are doing with the current algorithm.

Discussion

This work has studied the use of post-processing algorithms to improve the performance of myoelectric control systems for simultaneous movements. The problem of controlling an artificial limb that can move in three DoFs was studied for this purpose, but the results should be applicable to other movements as well. The myoelectric control system that was used to control the movements was developed by Chalmers, COO and Integrum AB, it is based on the BioPatRec software and uses a pattern recognition based approach.

When designing a post-processing algorithm for simultaneous control, the distributions of false positives and false negatives made by the classifier should be kept in mind. Positives here are the prediction of a movement, and negatives are the absence of a movement in the predictions. Classifiers that work with individual control will make an equal amount of false positives as false negatives. If a movement is incorrectly predicted by the classifier, the correct movement has to be absent as only one movement is predicted at a time. However, during simultaneous control the distributions of false positives and false negatives can vary.

In this work it was found that the system at Chalmers, COO and Integrum AB produces a lot of false positives during individual movements, while it only produces a few false negatives. The distribution changes when users are trying to perform three DoF movements. Then the system produces a lot of false negatives, but only a few false positives. This makes it hard to create a good post-processing algorithm for simultaneous control, as one cannot focus on one problem more than the other. This also makes it harder to evaluate the performance of the post-processing algorithms, since they might perform different with different classifiers. An algorithm that effectively removes false positives but not false negatives might work excellently with one classifier, but perform worse with another.

The post-processing algorithms that were studied in this work were taken from systems that control individual movements; the Majority Vote (Englehart and Hudgins, 2003) and the Decision-Based Velocity Ramp (Simon et al., 2011*a*). These two are not the only algorithms that has been suggested for myoelectric control, but they are two well-known algorithms in the field. It is possible that there exist other algorithms that had proven to perform better, but they are not known to the author.

As the post-processing algorithms originally were designed for the control of individual movements, it was hypothesized that some modifications were necessary for the algorithms to function properly. The Majority Vote algorithm were modified in two different versions that would be able to handle both false positives and false negatives. The results showed that both modifications allowed the algorithm to work with the simultaneous control, and either of the MV2 and MV3 versions of the algorithm can be used. The original version, MV1, proved to outperform the other during individual movements. This is because the classifier was prone to make a lot of false positives during these movements. Changing the classifier to not predict movements so easily, e.g. by increasing the MLP thresholds, the performance might not have differed so much between the algorithms. However, the MV1 version showed to perform really bad with three DoF movements. These results show that the original algorithm is good at handling false positives, but not at handling false negatives.

The Majority Vote algorithms showed that the performance can be kept at the same level, even when the analysis window is shortened, by using more predictions to make the decisions. This result might be of importance to future development, as it allows the pattern recognition based myoelectric control systems to be implement in real prosthetic devices where the computational power might be limited. The finding is consistent with previous results from the control of individual movements (Englehart and Hudgins, 2003). The time increment was fixed in this work and the performance might actually have improved if it had been optimized. The time increment is limited by the processing time of the analysis window, which in turn depends on both the hard- and software of the system. The optimal setup should

therefor be evaluated for each control system individually.

The decision-based velocity ramp was evaluated in two experiments, performed by four and three subjects respectively. The experiments should have been conducted with a larger number of subjects for the results to be really convincing, but as they agree with previous findings they should still be taken considered as a good indication (Simon et al., 2011*a*). The algorithm was found to improve the performance with individual movements but the improvements were not as big with combined movements. To allow the algorithm to work better with simultaneous control it was modified to handle false negatives better, by adding to the counters even when movements were absent in the predictions. The modification showed an improved performance with three DoF movements, but it reduced the performance with individual ones.The modified version performed overall worse than the original version, but the modification should not be neglected as it might be beneficial with another classifier. It all depends on the distributions of false positives and negatives. A threshold that switches between the original and modified implementation of the ramp could possibly be used to gain the benefits from both algorithms, that modification is left for future work.

The use of Majority Vote does not have to exclude the use of the Decision-Based Velocity Ramp. The two post-processing algorithm could be used as complements to each other as they can be used for different purposes. The Majority Vote algorithm can be used to generate a good output stream at a reduced computational cost. The output stream can then be used with the Velocity Ramp to ensure that user has a good controllability of the prosthesis. Previous results has compared the two algorithms as competitors to each other (Simon et al., 2011*a*) but as their purposes are different they should be viewed as complements instead.

Post-processing algorithms can only be used to improve the performance to some extent, the main performance is still up to the classifier. If the pattern-recognition based myoelectric control systems are to make a breakthrough, they have to be more robust than they are today. It has been shown that movements performed with the limb at different positions or with different force has can reduce the classification accuracy to levels where these systems become unusable (Scheme and Englehart, 2011). In such a case it does not matter whether or not post-processing algorithms are used. A suggested solution to this problem is often to include data from several positions/forces in the training phase (Scheme and Englehart, 2011), but is not realistic that the user should have to train the prosthesis this way. The simultaneous control using BioPatRec already requires the combined movements to be recored separately. The user already performs 26 movement combinations during the training phase, adding multiple positions and forces would make the already tiring process even more time consuming. The key to develop a robust pattern recognition based myoelectric control system seem instead to be the understanding of how the signal features are moving during real-time operation. If one knows how the features will move in feature space, the additional data can be sampled without the user actually having to perform the movements. Once the classifier is working robustly, post-processing algorithms can be developed to further improve the performance. These algorithms should be developed with the behavior of the classifier in mind, and the Majority Vote and Decision-Based Velocity Ramp should be thought of as complements rather than competitors.

Conclusions

This work has investigated the possibilities of using post-processing algorithms to improve myoelectric control system for simultaneous movements. The situation was found to be more complex than that for the control of individual movements, and that the performance of a post-processing algorithm depend more on the behavior of the classifier in this case. Classifiers that are designed to predict several movements simultaneously were found to struggle both with false positives and false negatives. The problems are not necessary equally severe, and they do not necessary occur with the same movements. For example, the classifiers that were analyzed in this thesis produced a lot of false positives during one DoF movements while they produced a lot of false negatives during three DoF movements. The fact that the issue depends of which movement the user tries to perform makes it harder to design one-size fits-all post-processing algorithms. The same algorithm could perform differently with two different classifiers, depending on how likely the classifier is to produce false positives or false negatives.

The two post-processing algorithms that has be used in this thesis were both brought from the control of individual movements, and the results suggest that some modifications to the algorithms are necessary. The first algorithm was the Majority Vote, originally proposed by Englehart and Hudgins (2003). The algorithm was found to improve the classification accuracy of one DoF movements even in the simultaneous control system. However, combined movements were classified with worse accuracy, suggesting that the algorithm is not suited for simultaneous control. To solve this problem, two modified versions of the algorithm were suggested in section 5.1, both which proved to be better suited for the simultaneous control. The classifications made by the system were not significantly improved with the modified algorithms, the performance was kept at the same level. The strength of the algorithms lie instead in the fact that shorter analysis windows can be used to achieve the same performance. Shorter analysis windows are computationally cheaper to processes, making it possible to implement the system in hardware with limited computational power. The algorithm might be necessary when building a real prosthetic device but it cannot be used in the original form, it has to be modified to work with the simultaneous control.

The second post-processing algorithm that was tested was the Decision-Based Velocity Ramp, originally suggested by Simon et al. (2011*a*). The algorithm also showed a worse performance with combined movements suggesting that a modification is necessary for the algorithm to operate as good as possible. In this work, the algorithms was modified to better handle false negatives, but the modification did not improve the overall performance. The modification should however not be neglected as it might work better with another classifier.

Post-processing algorithms can help improve the performance of simultaneous control systems, but for these systems to really make a breakthrough, the classification has to be more robust. Post-processing can only help to some extent and these systems are not yet robust enough for everyday use. The Majority Vote and the Decision-Based Velocity Ramp should not be viewed as competitors as they have been earlier, but instead they can be used as complements to each other.

Bibliography

Bunderson, N. E. and Kuiken, T. A. (2012), 'Quantification of feature space changes with experience during electromyogram pattern recognition control', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **20**(3), 239–246.

Childress, D. S. (1985), 'Historical aspects of powered limb prostheses', *Clinical prosthetics and orthotics* **9**(1), 2–13.

Dalley, S., Varol, H. and Goldfarb, M. (2012), 'A method for the control of multigrasp myoelectric prosthetic hands', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **20**(1), 58–67.

Demšar, J. (2006), 'Statistical comparisons of classifiers over multiple data sets', *The Journal of Machine Learning Research* **7**, 1–30.

Englehart, K. and Hudgins, B. (2003), 'A robust, real-time control scheme for multifunction myoelectric control', *Biomedical Engineering, IEEE Transactions on* **50**(7), 848–854.

Farrell, T. R. and Weir, R. F. (2007), 'The optimal controller delay for myoelectric prostheses', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **15**(1), 111–118.

Farrell, T. and Weir, R. F. (2008), Analysis window induced controller delay for multifunctional prostheses, Myoelectric Symposium.

Fougner, A., Scheme, E., Chan, A. D. C., Englehart, K. and Stavdahl, Ø. (2011), 'Resolving the limb position effect in myoelectric pattern recognition', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **19**(6), 644–651.

Fougner, A., Stavdahl, O., Kyberd, P., Losier, Y. and Parker, P. (2012), 'Control of upper limb prostheses: Terminology and proportional myoelectric control-a review', *Transactions on Neural Systems and Rehabilitation Engineering* **20**(5).

Hargrove, L., Englehart, K. and Hudgins, B. (2006), The effect of electrode displacements on pattern recognition based myoelectric control, *in* 'Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE', IEEE, pp. 2203–2206.

Hargrove, L. J., Englehart, K. and Hudgins, B. (2007), 'A comparison of surface and intramuscular myoelectric signal classification', *Biomedical Engineering, IEEE Transactions on* **54**(5), 847–853.

Jiang, N., Dosen, S., Muller, K. R. and Farina, D. (2012), 'Myoelectric control of artificial limbs - is there a need to change focus?[in the spotlight]', *Signal Processing Magazine, IEEE* **29**(5), 152–150.

Jiang, N., Vest-Nielsen, J. L. G., Muceli, S. and Farina, D. (2012), 'Emg-based simultaneous and proportional estimation of wrist/hand dynamics in uni-lateral trans-radial amputees', *Journal of NeuroEngineering and Rehabilitation* **9**(1), 42.

Khushaba, R. N., Kodagoda, S., Takruri, M. and Dissanayake, G. (2012), 'Toward improved control of prosthetic fingers using surface electromyogram (emg) signals', *Expert Systems with Applications* .

Kuiken, T. A., Li, G., Lock, B. A., Lipschutz, R. D., Miller, L. A., Stubblefield, K. A. and Englehart, K. B. (2009), 'Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms', *JAMA: the journal of the American Medical Association* **301**(6), 619–628.

Lock, B. A. (2005), *Design and interactive assessment of continuous multifunction myoelectric control systems.*, University of New Brunswick (Canada), Master Thesis.

Lock, B. A., Englehart, K. and Hudgins, B. (2005), Real-time myoelectric control in a virtual environment to relate usability vs. accuracy, Myoelectric Controls/Powered Prosthetics Symposium.

Lorrain, T., Jiang, N. and Farina, D. (2011), 'Influence of the training set on the accuracy of surface emg classification in dynamic contractions for the control of multifunction prostheses', *Journal of neuroengineering and rehabilitation* **8**(1), 25.

Micera, S., Carpaneto, J. and Raspopovic, S. (2010), 'Control of hand prostheses using peripheral information', *Biomedical Engineering, IEEE Reviews in* **3**, 48–68.

Mouzé-Amady, M. and Horwat, F. (1996), 'Evaluation of hjorth parameters in forearm surface emg analysis during an occupational repetitive task', *Electroencephalography and Clinical Neurophysiology/Electromyography and Motor Control* **101**(2), 181–183.

Muceli, S. and Farina, D. (2012), 'Simultaneous and proportional estimation of hand kinematics from emg during mirrored movements at multiple degrees-of-freedom', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **20**(3), 371–378.

Muceli, S., Jiang, N. and Farina, D. (2010), Multichannel surface emg based estimation of bilateral hand kinematics during movements at multiple degrees of freedom, *in* 'Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE', IEEE, pp. 6066–6069.

Nielsen, J. L., Holmgaard, S., Jiang, N., Englehart, K. B., Farina, D. and Parker, P. A. (2011), 'Simultaneous and proportional force estimation for multifunction myoelectric prostheses using mirrored bilateral training', *Biomedical Engineering, IEEE Transactions on* **58**(3), 681–688.

Ortiz-Catalan, M., Brånemark, R. and Håkansson, B. (2013*a*), 'Biopatrec: A modular research platform for the control of artificial limbs based on pattern recognition algorithms', *Source Code for Biology and Medicine* **8**(11).

Ortiz-Catalan, M., Brånemark, R. and Håkansson, B. (2013*b*), 'Evaluation of classifier topologies for the real-time classification of simultaneous limb motions', *Accepted in proceedings of the XXXV Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Osaka, Jul 3-7, 2013.

Ortiz-Catalan, M., Brånemark, R. and Håkansson, B. (2013*c*), 'Real-time classification of simultaneous hand and wrist motions using artificial neural networks with variable threshold outputs', *Accepted in proceedings of the XXXIV International Conference on Artificial Neural Networks (ICANN)*, WASET, Amsterdam, May 15-16, 2013.

Ortiz-Catalan, M., Brånemark, R., Håkansson, B. and Delbeke, J. (2012), 'On the viability of implantable electrodes for the natural control of artificial limbs: review and discussion', *BioMedical Engineering OnLine* **11**(33).

Peerdeman, B., Boere, D., Witteveen, H. J. B., Hermens, H. J., Stramigioli, S., Rietman, J. S., Veltink, P. H., Misra, S. et al. (2011), 'Myoelectric forearm prostheses: State of the art from a user-centered perspective', *Journal of Rehabilitation Research & Development* **48**(6), 719 – 738.

Scheme, E. and Englehart, K. (2011), 'Electromyogram pattern recognition for control of powered upper-limp prostheses: State of the art and challanges for clinical use', *Journal of Rehabilitation Research & Development* **48**(6), 643–660.

Sensinger, J. W., Lock, B. A. and Kuiken, T. A. (2009), 'Adaptive pattern recognition of myoelectric signals: exploration of conceptual framework and practical algorithms', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **17**(3), 270–278.

Simon, A. M., Hargrove, L. J., Lock, B. A. and Kuiken, T. A. (2011*a*), 'A decision-based velocity ramp for minimizing the effect of misclassifications during real-time pattern recognition control', *Biomedical Engineering, IEEE Transactions on* **58**(8), 2360–2368.

Simon, A. M., Hargrove, L. J., Lock, B. A. and Kuiken, T. A. (2011*b*), 'Target achievement control test: Evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses', *J Rehabil Res Dev* **48**(6), 619–28.

Smith, L. H., Hargrove, L. J., Lock, B. A. and Kuiken, T. A. (2011), 'Determining the optimal window length for pattern recognition-based myoelectric control: Balancing the competing effects of classification error and controller delay', *Neural Systems and Rehabilitation Engineering, IEEE Transactions on* **19**(2), 186–192.

Appendices

Functionally Partitioned Model of the Prosthesis Control Problem

The prosthesis control problem is a generic term for all the parts that has to be working in order for a user to control the movements of an artificial limb. There are many articles on the subject of myoelctric prostheses control, and in an attempt to create a coherent terminology and description of the problem Fougner et al. (2012) introduced a functionally partitioned model, see figure A.1. The proposed model has a general structure that identifies key aspects of the problem by braking it down into smaller sub-problems. The sub-problems can be solved in a variety of ways to achieve myoelectric control of a prosthesis. The model consists of eight layers, but the layers can be organized into three main categories; preprocessing, intent interpretation and output. The function of these layers are briefly described below along with some examples of how they can be implemented.

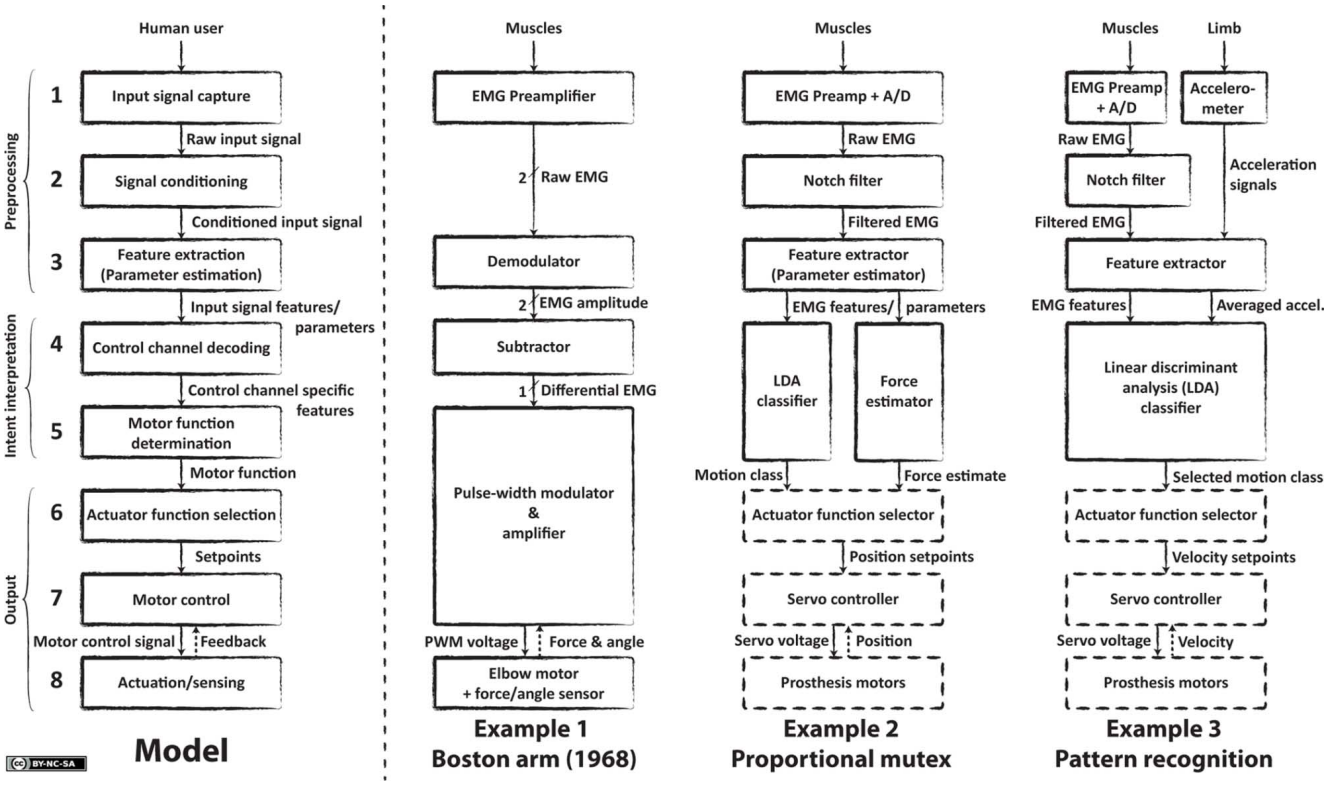


Figure A.1: A functionally partitioned model and corresponding taxonomy for the prosthesis control problem. This figure is taken from Fougner et al. (2012), where it was licensed under a Creative Commons BY-NC-SA license. The model they proposed can be viewed to the left of the dashed line. To the right of the line are three myoelectric control systems described using the model. The model consists of three main categories, preprocessing, intent interpretation and output which deals with various sub-problems of the general prosthesis control problem.

A.1 Preprocessing

In order for the user to be able to control a robotic prosthesis he must be able to produce some kind of inputs. The function of the preprocessing layers are to collect these inputs and process the signal(s) in

such a way that they can be used in the next layers. In the case of myoelectric control, the myoelectric signals produced by muscle contractions are used as input to the system. Processing of the signals includes applying filters and extracting signal features that can be used when determining the user's intention. A signal feature could be any kind of measurement representative to the recorded signal. Measurements such as the amplitude and the sign could be taken instantaneously whereas more advanced measurements such as the wavelength or frequency spectra requires a longer portion of the signal to be calculated. The preprocessing layers should be implemented depending on the desired result of the final system. Some examples on implementations that were found during the literature review are given below.

Single Signal Feature

The most simple strategies to control a myoelectric prosthesis makes use of only a single feature of the measured input signal, e.g. the amplitude. This feature can then be used together with threshold control, when the feature of the signal is larger than a set threshold, a function of the prosthesis is activated.

Multiple Signal Features

More complex control systems used multiple signal features as it provides more information. In particular are multiple signal features used together with pattern recognition algorithms to help them made the correct classifications. The choice of signal features to use is not trivial and several feature sets has been proposed together with pattern recognition based systems. Some examples will be given in next section that treats those systems.

Multimodal Approach

Multimodal approach is the idea of not only collecting input signals from the user but also from sensors in the prosthesis. The sensors would provide extra information that could be used by the intent interpretation layers. The strategy was proposed by Fougner et al. (2011) as a way to solve the problem that pattern recognition based systems performs differently if the prosthesis is located at different positions. By adding accelerometers to the prosthesis the performance of their system could be increased as the pattern recognition algorithm were provided with information of the position. Other researchers does also advocate the approach (Jiang, Dosen, Muller and Farina, 2012) as they think that it is necessary to create a myoelectric control system robust enough for everyday usage.

A.2 Intent Interpretation

Once the preprocessing layers have collected the input signals and extracted the desired signal features, the intent interpretation layers uses the features to decode what the users intends to do. This decoding can be performed differently depending on how advanced the control system is designed to be. If the prosthesis only allows the user operate one function, a threshold on the signal features could activate that function. More advanced systems can apply pattern recognition algorithm on the signal features to detect different intents using the same inputs. Some of the concepts that involves the intent interpretation layers are described below to the versatility when designing functional prostheses.

Single Function Systems

These are prostheses that only has one function that the user can control, e.g. the opening and closing of a hand prosthesis. These are the most simple functional prostheses and they can be controlled for instance using myoelectric signals. The myooelectric signals can be measured from opposite sides of the amputated limb, each side controlling one direction of the function, e.g. signals from the biceps could be used to close the hand while signals from the triceps to open it again. Activation of the functions could be determined using a threshold on the amplitude of the measured signal.

State Machine Systems

An effective way to offer the user more functionality without having to increase the number of inputs is to implement a state machine. The state machine works by allowing the user to activate different functions (states) with the same input depending on the state of the prosthesis. A state is in this sense could be a posture of the prosthesis and the inputs are used to make the transition between different states. A system constructed in this way allows the user to manipulate the prosthesis to far greater extent than just activating a function. The added functionality however requires that the user chooses the correct path to end up in a desired state. Depending on how many inputs that are available to the user, the states of the prosthesis can be linked together in networks of different complexity to allow the user to quickly reach any desired state. The state machine leaves a lot of the functionality for the designer to implement as she or he is free to build the networks that connects the states as she or he wishes.

A system that uses the finite state machine is e.g. the Vanderbilt MultiGrasp Hand Prosthesis presented in Dalley et al. (2012). In their system the user can cycle through six different grasp types and two hand postures using only two electrode channels. The states are divided into two different categories depending on the position of the thumb and are arranged in such an order that the prosthetic hand moves as smooth as possible when transitioning between the states. Flexors of the forearm are used to navigate through the states in one direction and the extensors in the other. A co-contraction of both the flexors and extensors allows the user to quickly switch between the two different grasp types. This system allows the user to perform approximately 85% of the different grasping tasks that are used in activities of daily living, without adding any more input channels.

The biggest problem with the state machine for prosthesis control is that it requires high attention from the user. The user has to always be aware of which the state the prosthesis is in to manipulate it as desired. If the prosthesis is in another state than the user thinks, the same inputs could have a totally undesired effect.

Pattern Recognition Based Systems

Some control systems have combined the fields of myoelectric control and pattern recognition in the search for a more intuitive control of functional prostheses. Pattern recognition based systems works by applying pattern recognition algorithms to the signal features that are extracted by the preprocessing layers. The main advantage that these systems have is that the control of the prosthesis can be made very intuitive. The system can be trained to recognize patterns of the signal features produced when the user tries to activate a specific function, and later activate that function when those patterns are observed. This way the user can choose how to activate a function by simply telling the system to recognize the patterns that are produced. These systems can for instance be trained to open an artificial hand whenever they recognize that user is trying to open his or her amputated hand, making the control as intuitive as possible. The control is established by teaching the system how the user wants to activate the functions rather than teaching the user how the system wants them to be activated . However there has been shown that some learning is required by the user as experienced users tend to operate them with more accuracy than novice users (Bunderson and Kuiken, 2012).

Kinematic Estimation

Kinematic estimation is a technique that applies machine learning algorithms to estimate the desired joint angles of the prosthesis by fitting the signal features to a function. Given the features from the preprocessing layers, the algorithm outputs the angles of each DoF of the prosthesis. The user is able to directly input a desired position of the prosthetic limb, much like a real limb is controlled. In contrast, pattern recognition based control only allows the user activate and stop the different functions of the prosthesis, making it crucial to deactivate a function at the correct movement to reach a desired position. Studies on kinematic estimation have, to the authors knowledge, all used a technique called bilateral mirrored training (Nielsen et al., 2011) (Muceli et al., 2010) (Muceli and Farina, 2012) (Jiang, Vest-Nielsen, Muceli and Farina, 2012). The user is asked to perform a continuous movement with both hands, amputated and intact. The position of the intact hand is tracked using a camera tracking system while the electromyogram is collected from the amputated limb. By synchronizing the information of the joint angles and the measured EMG, machine learning algorithms can be used to fit a function

that relates the two. This function is later used to estimate the desired joint position during real time operation. A drawback with this method is that it is currently only applicable to uni-lateral amputees, patients with one limb intact, as it is needed to track the kinematics.

A.3 Output

The final layers in the model presented by Fougner et al. (2012) are called the output layers. These layers are used to translate the intended motions into motor commands and collecting information from various sensors in the prosthesis. These sensors can be used for feedback control of the motor or as feedback to the user. The output layers should contain the control of the actuators, the motors that control the prosthesis. This could for example be position and velocity control of fingers and wrist. Sometimes these layers are implemented virtually, which has been the case for this thesis. This way systems can easily be tested and evaluated throughout the development process to make sure that they are performing good. The implementation of the output layers should however not be neglected as they have to be present to control an actual prosthesis.

Outlier and Adjacent Values in Boxplot

Many of the results in this work are presented with boxplots that has been generated using MATLAB (MathWorks, Inc.). In these plots the distribution of the results are presented as boxes, whiskers and outliers. The boxes in these plots ranges from the 25th to the 75th percentile, whiskers reaches to the upper and lower adjacent values and any outliers are plotted separately. The adjacent value is the most extreme result that is not considered as an outlier. Outliers has been calculated in this work using the standard *boxplot()* routine in MATLAB (MathWorks, Inc.) which follows equation (B.1).

$$\begin{aligned} X &> q_3 + 1.5(q_3 - q_1) \\ X &< q_1 - 1.5(q_3 - q_1) \end{aligned} \tag{B.1}$$

where X is considered as an outlier, q_1 the 25th percentile and q_3 the 75th percentile.