Temporal Segmentation of Human Motion for Rehabilitation

by

Jonathan Feng-Shun Lin

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2017

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner	Tamim Asfour Professor Institute for Anthropomatics and Robotics Karlsruhe Institute of Technology
Supervisor	Dana Kulić Associate Professor Electrical and Computer Engineering University of Waterloo
Internal Members	Otman Basir Professor Electrical and Computer Engineering University of Waterloo
	Fakhri Karray Professor Electrical and Computer Engineering University of Waterloo
Internal-External Member	James Tung Assistant Professor Mechanical and Mechatronics Engineering University of Waterloo

Abstract

Current physiotherapy practice relies on visual observation of patient movement for assessment and diagnosis. Automation of motion monitoring has the potential to improve accuracy and reliability, and provide additional diagnostic insight to the clinician, improving treatment quality, and patient progress. To enable automated monitoring, assessment, and diagnosis, the movements of the patient must be temporally segmented from the continuous measurements. Temporal segmentation is the process of identifying the starting and ending locations of movement primitives in a time-series data sequence. Most segmentation algorithms require training data, but *a priori* knowledge of the patient's movement patterns may not be available, necessitating the use of healthy population data for training. However, healthy population movement data may not generalize well to rehabilitation patients due to large differences in motion characteristics between the two demographics. In this thesis, four key contributions will be elaborated to enable accurate segmentation of patient movement data during rehabilitation.

The first key contribution is the creation of a segmentation framework to categorize and compare different segmentation algorithms considering segment definitions, data sources, application specific requirements, algorithm mechanics, and validation techniques. This framework provides a structure for considering the factors that must be incorporated when constructing a segmentation and identification algorithm. The framework enables systematic comparison of different segmentation algorithms, provides the means to examine the impact of each algorithm component, and allows for a systematic approach to determine the best algorithm for a given situation.

The second key contribution is the development of an online and accurate motion segmentation algorithm based on a classification framework. The proposed algorithm transforms the segmentation task into a classification problem by modelling the segment edge point directly. Given this formulation, a variety of feature transformation, dimensionality reduction and classifier techniques were investigated on several healthy and patient datasets. With proper normalization, the segmentation algorithm can be trained using healthy participant data and obtain high quality segments on patient data. Inter-participant and inter-primitive variability were assessed on a dataset of 30 healthy participants and 44 rehabilitation participants, demonstrating the generalizability and utility of the proposed approach for rehabilitation settings. The proposed approach achieves a segmentation accuracy of 83-100%.

The third key contribution is the investigation of feature set generalizability of the proposed method. Nearly all segmentation techniques developed previously use a single sensor modality. The proposed method was applied to joint angles, electromyogram, motion capture, and force plate data to investigate how the choice of modality impacts segmentation performance. With proper normalization, the proposed method was shown to work with various input sensor types and achieved high accuracy on all sensor modalities examined. The proposed approach achieves a segmentation accuracy of 72-97%.

The fourth key contribution is the development of a new feature set based on hypotheses about the optimality of human motion trajectory generation. A common hypothesis in human motor control is that human movement is generated by optimizing with respect to a certain criterion and is task dependent. In this thesis, a method to segment human movement by detecting changes to the optimization criterion being used via inverse trajectory optimization is proposed. The control strategy employed by the motor system is hypothesized to be a weighted sum of basis cost functions, with the basis weights changing with changes to the motion objective(s). Continuous time series data of movement is processed using a sliding fixed width window, estimating the basis weights of each cost function for each window by minimizing the Karush-Kuhn-Tucker optimality conditions. The quality of the cost function recovery is verified by evaluating the residual. The successfully estimated basis weights are averaged together to create a set of time varying basis weights that describe the changing control strategy of the motion and can be used to segment the movement with simple thresholds. The proposed algorithm is first demonstrated on simulation data and then demonstrated on a dataset of human subjects performing a series of exercise tasks. The proposed approach achieves a segmentation accuracy of 74-88%.

Acknowledgments

Research is rewarding but often difficult. The completion of this thesis was made possible only through the contributions of various people and organizations. I would like to thank...

My supervisor, Dr. Dana Kulić, for her support, guidance, and expertise. I am truly grateful for the constant mentorship, steady support, and enthusiastic advices, in both research and in life. I am constantly encouraged by your passion for research, and humbled by your perseverance in finding the n^{th} grammar mistake in my writing.

My doctorate examining committee, Dr. Tamim Asfour, Dr. Otman Basir, Dr. Fakhri Karry, and Dr. James Tung, for their time in reviewing my thesis, for providing insight into the research field, and where to go from here. The reason that I am able to claim any degree of expertise is due to the expertise of those who have came before me.

My research exchange supervisor at the Tokyo University of Agriculture and Technology, Dr. Gentiane Venture, for providing me an opportunity for a research exchange and see a little bit more of the world.

My research collaborators, who consistently expand my research horizons and push me towards new and challenging ideas. My project was only possible with the many hours of input and troubleshooting from Dr. Michelle Karg, Dr. Ali Samadani, Dr. Vincent Bonnet, Dr. Adina Panchea, Vladmir Joukov, and Brandon DeHart. I would also like to thank the other students of the Kulić lab for their continual ideas, support, and the extra pair of hands to start the motion capture system.

My research participants. This work would not be possible without the participants and clinicians from the University of Waterloo, Toronto Rehabilitation Institute, and St. Joseph Health Centre Guelph. I am indebted to each of you for the insight and data needed. It has gone a long way and has made all the difference.

My funding sources. This research would not have been possible without the support of the National Sciences and Engineering Research Council, the Ontario Centre of Excellence, the Japan Student Services Organization, the University of Waterloo, and Cardon Rehabilitation.

My infrastructure providers. Special thanks to Compute Canada for access to the SHARCNET high performance computation cluster, and to Thalmic Labs for access to the Myo armband.

My friends and family. Each of you, from both inside and outside of the University of Waterloo, Simply, Grad Cell, and elsewhere, thank you for your unwavering support and encouragement throughout my many years at Waterloo. I have skipped out on many social gatherings, but I am humbled by your continual forgiveness.

The research community at large. For my Master's thesis, I acknowledged Dr. Martin Schwartz [198] and Dr. Jorge Cham [46] for their respective work. In the years since, their work continue to remind me that research is worthwhile because it is hard.

In his heart a man plans his course, but the Lord determines his steps. - *Proverbs 16.9*

Dedication

This thesis is dedicated to my wife, Joanna.

Each encouragement, support, and lesson did much to get me to where I am today. Thank you for every decision and sacrifice made, for always thinking of me.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

***k*-NN**	*k*-nearest neighbour
AdaBoost	adaptive boosting
ANN	artificial neural networks
bagging	bootstrap aggregation
COP	centre of pressure
CR	continuous random gestures [of the UWT dataset]
CV	cross-validation
DOC	direct optimal control
DOF	degrees of freedom
DTW	dynamic time warping
EEG	electroencephalography

EM	expectation-maximization [algorithm]
EMG	electromyogram
FDA	Fisher's discriminant analysis
FN	false negative
FP	false positive
GRF	ground reaction force
IG	individual gesture [of the UWT dataset]
IMU	inertial measurement unit
IOC	inverse optimal control
KKT	Karush-Kuhn-Tucker [conditions]
LDA	linear discriminant analysis
LOMO	leave-one-motion-out
LOPO	leave-one-participant-out
LQR	linear quadratic regulator
MAV	mean absolute value [of EMG]

PC	Principal component
PCA	principal component analysis
PIP	pairwise inner-product
QDA	quadratic discriminant analysis
RBF	radial basis function
RMS	root-mean-square [of EMG]
RMSE	root-mean-square error
RP	RMS and PIP [of EMG]
RRP	raw EMG, RMS, and PIP [of EMG]
SJHCG	St. Joseph's Health Centre, Guelph
SVD	singular value decomposition
SVM	support vector machine
TE	Teager energy [of EMG]
TN	true negative
TO	trajectory optimization

TP	true positive
TRI	Toronto Rehabilitation Institute
URFI	University of Rome Foro Italico
UT	University of Tokyo
UW	University of Waterloo
UWT	University of Waterloo and Thalmic Labs
vfHMM	velocity feature HMM
WFL	waveform length [of EMG]
ZVC	zero-velocity crossing

List of Symbols

J	function, cost
L	function, Lagrangian
g	function, inequality constraints
h	function, equality constraints
$\mathbf{A}$	matrix, regressor
$\mathbf{C}$	matrix, Coriolis and centrifugal
$\mathbf{G}$	matrix, gravity
$\mathbf{M}$	matrix, inertial
$\mathbf{Q_{obs}}$	matrix, a combined variable representing observed trajectory, velocity, and acceleration
$\mathbf{V}$	matrix, Vandermonde spline fitting
x	matrix, input dataset or state variables

$\mathbf{q_{ck}^*}$, $\mathbf{\dot{q}_{ck}^*}$, $\mathbf{\ddot{q}_{ck}^*}$
: matrix, joint trajectory, velocity, and acceleration at spline control knots such that the trajectory is optimal

$\mathbf{q_{ck}}$, $\mathbf{\dot{q}_{ck}}$, $\mathbf{\ddot{q}_{ck}}$
: matrix, joint trajectory, velocity, and acceleration at the spline control knots

$\mathbf{q_{const}}$, $\mathbf{\dot{q}_{const}}$, $\mathbf{\ddot{q}_{const}}$
: matrix, joint posture, velocity, and acceleration constraints

$\mathbf{\hat{q}_{obs}}$
: matrix, estimated trajectory

$\mathbf{q_{obs}}$, $\mathbf{\dot{q}_{obs}}$, $\mathbf{\ddot{q}_{obs}}$
: matrix, observed trajectory

C
: scalar, SVM slack magnitude

γ
: scalar, SVM radial kernel scaling

k
: scalar, the number of neighbours in k-NN

n_{exp}
: scalar, number of data points around manual segemnts to convert to p_1 points

n_{stack}
: scalar, number of time points to be included in he temporal feature stacking

t_{err}
: scalar, tolerance for temporal error in segmentation

Acc_{bal}
: symbol, balanced accuracy

Acc_{F_1}
: symbol, F_1 score accuracy

∇
: symbol, the gradient operator, with ∇_x denoting that the gradient is taken with respect to $\mathbf{x}$

p_0
: symbol, non-segment point

p_1
: symbol, segment edge point

q, $\dot{q}$, $\ddot{q}$
: symbol, joint angle and higher order deriviatives

τ, $\dot{\tau}$, $\ddot{\tau}$
: symbol, joint torque and higher order deriviatives

x, $\dot{x}$, $\ddot{x}$
: symbol, Cartesian position and higher order deriviatives

$\mathbf{z}$
: vector, KKT multipliers

$\mathbf{b_i}$
: vector, the i^{th} pivot

$\hat{\mathbf{c}}$
: vector, estimated basis weights

$\mathbf{c}$
: vector, ground truth basis weights

λ
: vector, Lagrangian multiplier

$\mathbf{p}$
: vector, spline coefficients

$\mathbf{t_{ck}}$
: vector, spline control knots

Chapter 1

Introduction

Segmentation is the process of identifying the starting and ending locations of movements of interest, breaking a continuous sequence of time-series motion data into smaller components, termed *movement primitives*. If the subject is performing more than one type of motion in a given recording session, *identification*, or *labelling*, of each segment with the appropriate motion type is also required.

Time-series segmentation has been employed to address many different research problems. In imitation learning, segmentation is used to isolate movement primitives from a sequence of demonstration motions provided to robots in order to train and improve their movement repertoire. Imitation learning is desirable because the most intuitive way for humans to teach any given action is by demonstration, which would make robotics more accessible to non-expert users [42, 81, 83]. Other applications which require segmentation include gesture recognition [232, 182, 200], temporal video segmentation [230], and speaker-turn segmentation [58, 208].

The focus of this thesis is to develop accurate time-series segmentation for physiotherapy and rehabilitation. Current physiotherapy practice relies on subjective measures and visual observation of the patient for diagnosis, assessment, and progress monitoring. Visual inspection, goniometry [163] and questionnaires [80, 235] are common tools employed by the physiotherapist. These tools tend to be subjective, and can suffer from inter-rater variability [223, 55]. Human motion measurement technology, such as motion capture [91], vision and range sensors [243], or inertial measurement units (IMUs) [139, 27], can be used to automate the observation process to improve accuracy and reliability. These systems can also serve as a data logging tool, assisting physiotherapists in progress monitoring and diagnosis.

To enable automated measurement and analysis, the system must measure the human movement and identify exercise movement segments from the time-series data. Such a system consists of movement measurement, pose estimation [90], movement segmentation and identification [134], and movement analysis [79] (Figure 1.1). The system should also interact with the the physiotherapist and the patient to provide real-time feedback and report on patient progress [116]. The focus of this thesis is the segmentation component of this system.

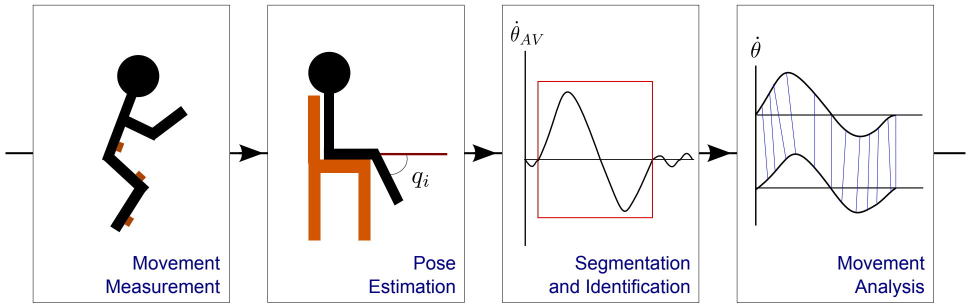
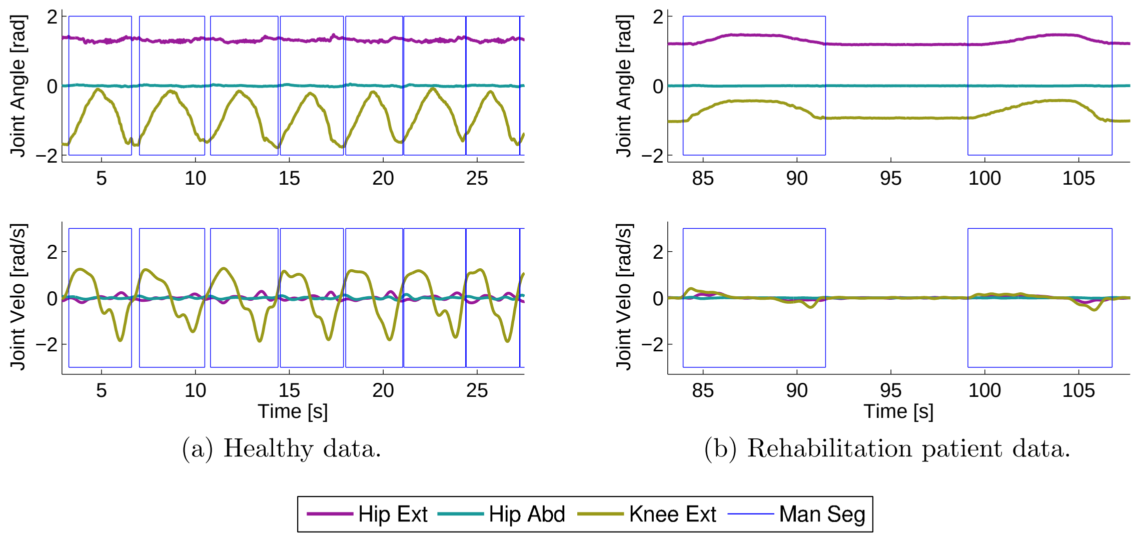


Figure 1.1: Components of an automatic rehabilitation supervision system. This thesis proposes algorithms for the segmentation component.

In order to enable continuous, online automated analysis of human movement data, accurate segmentation and identification of the movement is needed. One difficulty in performing accurate segmentation and identification is the large number of degrees of freedom (DOFs) in the movement data. Data streams used for single limb studies contain 4-6 DOFs [133], while full body human measurement can consist of 20-30 DOFs [171, 114], making scalability an important characteristic for any segmentation algorithm.

In addition, both segmentation and identification are made more difficult due to the variability observed in human movement. Motion can vary between individuals due to differing kinematics or dynamics characteristics, and also within a single individual over time [161], due to short term factors such as fatigue, or long term factors such as recovery or disease progression [227]. The participant may also start a subsequent movement before fully completing the previous one [149], leading to hybrid motion primitives. These factors introduce both spatial and temporal variability, which a successful segmentation algorithm must be able to handle.

Many approaches use healthy participant data to train the models used for segmentation. However, patient movements tend to be lower in velocity and range of motion, as well as exhibiting tremors and pauses during the motion (Figure 1.2). Inter- and intra-participant movement variability is a long-standing issue for segmentation, but is exacerbated by the changes in the patient movement patterns as they progress through the physiotherapy process [194, 186, 72]. Another major issue is *joint compensation* [59], where the patient uses joints and muscle groups different from the ones the therapist intended to perform a prescribed motion they find difficult, due to pain or disability [64]. These differences make it difficult for healthy movement templates to be generalized to rehabilitation patients.



(a) Healthy data. (b) Rehabilitation patient data.

Figure 1.2: Joint angle and velocity data of a healthy (left) and rehabilitation (right) participant performing a knee extension exercise, illustrating hip extension (hip ext), hip abduction (hip abd), knee extension (knee ext), and the manual segments (man seg). These figures, with both vertical and horizontal axes at equal scaling, show that motions performed by rehabilitation patients are slower, have smaller velocities, and smaller range of motion, when compared to healthy participants. In the patient movement, some evidence of joint compensation can also be seen, where the motion in the knee is accompanied by hip extension. These variations highlight the need to validate segmentation algorithms with patient data to assess generalization capability.

Lastly, some segmentation algorithms require labelled data for training, which can be time-consuming to generate. As a result, many publicly available datasets do not have manual segment data, making it hard to compare between existing segmentation algorithms.

In order for a given system to be suitable for physiotherapy applications, it should meet the following requirements:

Provide consistent and accurate segments.
This leads to a precise and reliable segmentation tool to provide physiotherapists with accurate range of motion and movement velocity data, as well as automatically counting the number of repetitions performed by the patient.

Process data and provide results in real-time
Real-time calculations, when paired with a suitable user interface, would allow for immediate feedback to patients, allowing them to correct any errors in exercise performance.

Be able to segment between repetitions of the same motion.
Rehabilitation and exercise data can include numerous repetitions of the same exercise, so an appropriate algorithm must be able to segment this type of data.

Be able to segment between different motion types.
Rehabilitation and exercise data can also include sequences of different movement types, so an appropriate algorithm must be able to handle sequences of different motion primitives.

Inter-primitive generalizability.
It is difficult for any template-based approach to be used on motions that it has not seen during training. However, since there are many key commonalities, such as a common resting posture, it is preferable for the algorithm be able to generalize to new motions. In this thesis, primitives that are included in the training are denoted as *known* or *seen* motions, while primitives absent from the training set are denoted as *unknown* or *novel* movements.

Inter-participant generalizability, particularly between the healthy and patient populations.
Since rehabilitation motions can vary greatly, a segmentation system for clinical applications must be able to accurately segment patient motion regardless of their degree of progress in the rehabilitation process.

Can utilize a wide range of sensor modalities.
A variety of sensor modalities, such as IMUs and force plates, can be utilized in the clinic. This allows for increased flexibility of deployment.

If training data is required, it should be easy to generate and update so that non-specialists can provide training data.

Ease of use is necessary to allow physiotherapists to create their own templates in order to extend the segmentation algorithm to handle customized exercises. Inter-primitive and inter-participant generalizability also reduce the necessity to generate a labelled template for each customized exercise.

1.1 Thesis Contributions

The contributions of this thesis are:

A framework for segmentation that can be used for algorithm development and comparison.

A comprehensive literature review of time-series segmentation was performed and used to formulate a segmentation framework.

While several review works surveyed the related fields of gesture [151] and activity recognition [3, 118], no previous reviews focused on time-series segmentation, which provides more granular temporal details than activity recognition. The proposed segmentation framework can be used to categorize and compare different segmentation algorithms, and serves as a starting point to guide new segmentation algorithm creation.

A motion segmentation algorithm that meets the requirements of a clinically-focused method.

To realize an algorithm that meets the requirements outlined above, a classifier-based segmentation technique that classifies all data points as segment edge points or non-segment points is proposed.

This classifier approach allows the algorithm to automatically extract segment edge characteristics, enabling inter-participant and inter-primitive generalization. This reduces the amount of template data required from clinicians in order to deploy the algorithm.

Systematic validation with multiple sensor modalities.

Unlike previous approaches, which mostly rely on a single modality, the proposed algorithm was tested using different sensor modalities, including joint angles, electromyogram (EMG), motion capture, and force plate data to investigate several modalities that have been used in both kinesiology research and the clinic. The results demonstrate that the proposed approach can be used with a variety of sensor platforms and features, and that low-cost sensors and features with limited computational requirements can be used to generate segments reliably.

The ability to utilize different sensor and feature sets allows clinicians to choose the most clinically appropriate sensing platform, or one that can accurately collect the motion data without impeding the participant.

Inverse optimal control for motion segmentation.

A novel feature set for segmentation motivated by human motor control theory is proposed. Theories of human motor control hypothesize that human motion trajectories are generated by optimization. The cost function, typically modelled as a sum of weighted basis functions representing features such as joint acceleration, linear acceleration, or torque, has been shown to change from task to task [213]. In this thesis, we propose to use the estimated basis function contributions over a sliding window as a feature for segmentation. The proposed approach is validated on a human movement dataset and promising early results are shown.

1.2 Thesis Outline

Chapter 2 provides background information on the mathematical tools discussed in the related work and throughout the thesis. An overview of different dimensionality transformation methods, classifiers, aggregators, and optimal control concepts is provided.

Chapter 3 proposes a segmentation framework that can be used for the development of a segmentation algorithm, as well as comparison between different segmentation algorithms. The four different components of a segmentation algorithm are discussed, and include data characterization, the definition of a segment, segmentation mechanics and the verification scheme.

Chapter 4 provides an overview of existing approaches to segmentation, as well as unsolved problems and issues with the existing approaches.

Chapter 5 details the different datasets that are used for the training and testing of the segmentation methods developed for this thesis. The datasets consist of both healthy and rehabilitation participants at differing stages of the rehabilitation process.

Chapter 6 proposes a segmentation algorithm that models the segment edge point to perform segmentation. Many segmentation methods rely on domain knowledge to set up the segmentation process, such as segmenting on velocity crossings. Instead of explicitly modelling the segment edge point characteristics, we use classifiers to learn the characteristics automatically, allowing for a more flexible segmentation algorithm. Different classifiers and feature selection approaches are compared to determine how they perform in time-series segmentation. This algorithm was tested using both healthy and rehabilitation participant data, as well as with features derived from different sensor modalities.

Chapter 7 proposes an algorithm for motion segmentation based on inverse optimal control. The proposed approach models the controller generating the human motion as a trajectory generator optimizing a sum of weighted basis functions, and estimates the basis weights over a sliding window. These basis weights are used for segmentation in a threshold-based approach. This algorithm was tested with healthy participant data performing various exercises.

Chapter 8 summarizes this thesis and directions for future work.

Chapter 2

Background

This chapter provides an overview of the existing algorithms commonly used for time-series modelling and segmentation, as well as key concepts in trajectory optimization.

2.1 Dimensionality Transformation

The algorithms in this section are designed to transform the feature set to allow subsequent algorithms to focus on latent features instead of the full feature set, which can reduce the number of correlated, redundant, or irrelevant features, and thus potentially improve separability and classification accuracy. With a reduced feature set, the computational cost is also reduced.

In this thesis, principal component analysis (PCA) and Fisher's discriminant analysis (FDA) are chosen for analysis as they are among the most common and well known dimensionality reduction techniques, and are simple to implement. PCA and FDA also provide examples of an unsupervised and a supervised dimensionality reduction technique, respectively.

2.1.1 Principal Component Analysis

The purpose of PCA [201] is to rotate the features of a dataset $\mathbf{X}$ into a space such that the new features $\mathbf{X}'$ are ordered by the amount of variance present in the dataset, as a linear combination of the original feature space. This is achieved by transforming the data set into a new set of orthogonal variables, the principal components (PCs), such that the first few PCs retain most of the variation in the original variables. The PCA projection matrix can be truncated in order to retain only the first k vectors, thus reducing the dimensionality of the resultant feature set. The PCA algorithm can be summarized as follows:

1. Calculate the principal components by taking the singular value decomposition (SVD), a method to factorize the matrix, of the original data: $\mathbf{X} = \mathbf{U\Sigma W}^T$, where $\mathbf{U}$ is the PCA transformation matrix that spans the basis of $\mathbf{X}$, $\mathbf{\Sigma}$ are the eigenvalues of $\mathbf{X}$, and $\mathbf{W}$ are the eigenvectors of $\mathbf{X}$, ordered by component variance.

2. To automatically determine k, the scree plot method [67] can be used. This method orders the fraction of total variance or eigenvalues in the data represented by each PC, allowing the designer to visually determine which PCs contribute the most variance and which PCs are of lesser importance. This allows for the selection of the top k PCs that contain a significant amount of variance. Alternatively, k can be hand selected.

3. The first k vectors of the $\mathbf{U}$ matrix will be retained. $\mathbf{X'} = \mathbf{U_k X}^T$ calculates the dimensionality-reduced dataset.

2.1.2 Fisher's Discriminant Analysis

FDA incorporates the class labels and generates a projection that maximizes the separation between the classes in the projection [85]. The FDA algorithm can be summarized as follows:

1. Calculate the within-class variance $\mathbf{\Sigma_w}$ by computing the variance of each set of labelled data, and add them together: $\mathbf{\Sigma_w} = \mathbf{\Sigma_1} + \mathbf{\Sigma_2}$

2. Calculate the total variance $\mathbf{\Sigma_t}$ by computing the variance of the whole dataset.

3. Calculate the between-class variance by subtracting the two calculated variances: $\mathbf{\Sigma_b} = \mathbf{\Sigma_t} - \mathbf{\Sigma_w}$.

4. Perform SVD on $\mathbf{\Sigma_w^{-1}\Sigma_b}$. The scree plot method (Section 2.1.1) can be used to determine a suitable number of k vectors to keep.

5. The first k vectors of the $\mathbf{U}$ matrix will be retained. $\mathbf{X'} = \mathbf{U_k X}^T$ calculates the dimensionality-reduced dataset.

2.2 Classifiers

Classification is the problem of identifying to which of a set of categories or labels a new observation belongs, on the basis of a labelled training set. These techniques are commonly used in pattern recognition, with statistical classifiers forming a large portion of ongoing research and application [85, 108, 180].

In this thesis, k-nearest neighbour (k-NN), decision tree, artificial neural network (ANN), and support vector machine (SVM) are chosen for analysis as they are wide spread and popular in the literature, and span a range of methods in terms of algorithm complexity.

2.2.1 *k*-nearest Neighbour

k-NN [85] is a simple algorithm where the label of an observation data point is determined by a majority vote of the closest k points to it. Any distance metric can be used, but typically, the Euclidean distance is employed [85].

k-NN is simple and intuitive, and requires no training time. The computation time increases with larger training sets, but k-NN works well when used in conjunction with feature selection, dimensionality reduction and data reduction algorithms, allowing k-NN to scale to larger datasets. However, the k parameter needs to be tuned [85].

2.2.2 Decision Tree

Decision trees [181] classify data points by propagating down a tree based on a sequence of feature-based thresholds. The label assigned to each data point is determined at the leaf node, and is dependent on the path taken through the tree. It can be paired with the bagging aggregator (Section 2.3.2) to create the random forest classifier [32]. Iterative dichotomiser 3 (ID3) [181] is a common implementation of the decision tree. Given the dataset $\mathbf{X}$, the ID3 is trained as follows:

1. A subsample of data $\mathbf{X_i}$ is randomly taken from $\mathbf{X}$.

2. Calculate the entropy of all unused attributes or DOF:

$$\mathcal{H}(i) = -\frac{p_i}{p_i + n_i} log_2 \frac{p_i}{p_i + n_i} - \frac{n_i}{p_i + n_i} log_2 \frac{n_i}{p_i + n_i}$$

where p and n denote the number of elements belonging to class p and n, respectively, if using attribute i.

3. Using the DOF that results in the lowest entropy, split $\mathbf{X_i}$ into subsets according to this DOF.

4. Using these subsets and DOF, create a decision tree node.

5. Repeat the above steps for other DOFs.

6. Once the tree has been constructed, the remainder of the data $\bar{\mathbf{X}}_\mathbf{i}$ is classified with the decision tree. A sample of the data points that are incorrectly classified are added to $\mathbf{X_i}$ and this process is repeated.

Decision trees are commonly employed as they have low computational cost and are intuitive to interpret in terms of individual features [25], but can easily overfit and tend to be sensitive to the quality of the initialization [85]. Various extensions to the ID3 algorithm have been proposed to improve noise rejection and robustness against missing information [181].

2.2.3 Artificial Neural Network

The ANN classifier [178] uses a large number of interconnected nodes to perform classification. At each given node, the inputs $\mathbf{x_i}$ are multiplied by a set of weights $\mathbf{w_i}$, and summed together. The weighted sum a is then passed through an activation function g. Various activation functions can be used, with the sigmoid function being the most common. The output of this node is then used as part of a weighted sum for a node in the next layer, until the propagation reaches the output layer and generates a final classification (Figure 2.1). Mathematically, this neuron behaviour function is characterized by:

$$a = \sum_{i=1}^{k} \mathbf{w_i}\mathbf{x_i} - \mathbf{w_0} \tag{2.1}$$

$$g = \frac{1}{1+e^{-a}} \tag{2.2}$$

ANN is trained by modifying the connection weights using the back-propagation method [86]:

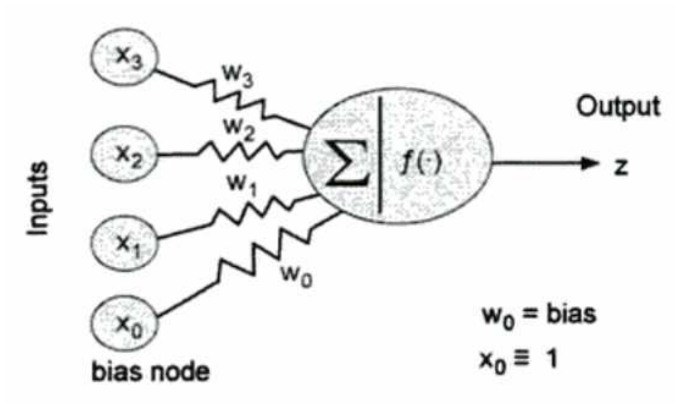


Figure 2.1: Artificial neural net activation function. Illustration from [178].

1. Initialize the weights w_i to small random values.
2. Choose one set of the training data, and apply the node computations through the network to obtain the ANN output d_i^u.
3. Calculate the partial derivative of the output error with respect to the output nodes: $\delta_i^l = g'(h_i^l)(d_i^l - y_i^l)$, where g' is the derivative of the activation function, y_i^l is the desired output value, and h is the neural input into the current i^{th} node at the l^{th} layer.
4. Propagate the error to preceding layers by accounting for all the j^{th} nodes that the current node feeds into: $\delta_i^l = g'(h_i^l)(\sum_j w_{ij}^{l+1}\delta_j^{l+1})$.
5. Update all the weights using $\Delta w_{ji}^l = \eta \delta_i^l y_j^{l-1}$ where η is the learning rate.
6. Repeat until the error in the output layer is below a threshold.

Unlike many other classification techniques, the neural network models do not provide easily interpretable intuition about the decision boundary, which can make tuning and training difficult. Once trained, the ANN performs classification at a rapid pace [195].

2.2.4 Support Vector Machine

SVM [35] is a deterministic binary classifier that determines a boundary that separates two classes based on the training data $\mathbf{X}$ (Figure 2.2). The decision hyperplane, defined by $\mathbf{w} \cdot \mathbf{X} + b = 0$, that has the largest margin from the closest training data point, is determined by finding the midpoint between the hyperplane that marks the boundary of one class ($H_1 : \mathbf{w} \cdot \mathbf{X} + b = +1$) and the boundary of the other ($H_2 : \mathbf{w} \cdot \mathbf{X} + b = -1$). $\mathbf{w}$ is the normal to the hyperplane, while $|b|/||\mathbf{w}||$ defines the perpendicular distance from the hyperplane to the decision hyperplane. To find the hyperplane, these constraints must be met:

$$\mathbf{x_i} \cdot \mathbf{w} + b \geq +1 \text{ for } y_i = +1 \tag{2.3}$$

$$\mathbf{x_i} \cdot \mathbf{w} + b \leq -1 \text{ for } y_i = -1 \tag{2.4}$$

which combines into:

$$y_i(\mathbf{x_i} \cdot \mathbf{w} + b) - 1 \geq 0 \ \forall \ i \tag{2.5}$$

Only key training data points, called the support vectors (SVs, circled in Figure 2.2), are used to specify the decision boundary, and are defined as training points that, if missing, will alter the shape of the boundary hyperplane. This allows the SVM to remain computationally efficient when compared to algorithms such as k-NN. If the classes are not linearly separable, kernel functions can be applied to move the data to a higher dimension so that the data becomes separable [35]. SVM is inherently two-class, but multi-class SVMs can be constructed with multiple two-class SVMs, or one-vs-all SVMs [35].

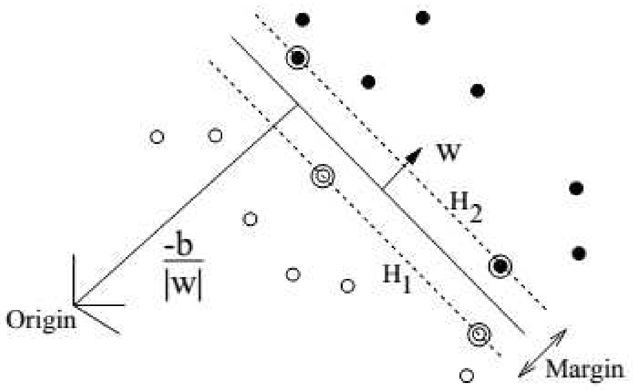


Figure 2.2: Separating hyperplanes for the SVM algorithm. The support vectors are circled. Illustration from [35].

2.3 Aggregators

Several weak or base classifiers can be combined via an aggregation scheme to outperform the individual classifier. Aggregators are techniques that combine multiple classifiers together, typically by weighting and voting schemes, to produce an overall label that focuses on incorrect labels that arose during training, or to smooth out overfitting.

2.3.1 Adaptive Boosting

The adaptive boosting (AdaBoost) [70] algorithm takes in an input training set $\mathbf{X}$, along with its labels $\mathbf{Y}$, and produces an array of weighted classifiers that are designed to be more accurate than a single classifier on its own. The goal of the AdaBoost algorithm is to train a series of weak learning algorithms using this training set and labels, and calculate a set of weights $\mathbf{D}$, over the training sample. The weights on the incorrectly classified labels are increased so that the weak classifiers contain more samples of the hard examples in the training set, aggregating multiple classifiers together to improve the overall performance [166]. Once $\mathbf{D_i}$ is trained, it can either be incorporated in the next iteration of training cycle, if the base classifier accepts weights, or $\mathbf{D_i}$ can be used to sample from the total pool of $\mathbf{X}$ for the next classifier training [199]. The training procedure for AdaBoost is as follows:

Given a training set $(\mathbf{x_1}, \mathbf{y_1}), ..., (\mathbf{x_k}, \mathbf{y_k})$, where $\mathbf{x_i} \in \mathbf{X}, \mathbf{y_i} \in \{-1, +1\}$:

1. Initialize the initial weights $\mathbf{D_1}$ to be a vector, with each element having a value of $1/k$.
2. A base classifier is trained on the sample data $\mathbf{x_i}$, and its accuracy is assessed against the training label $\mathbf{y_i}$. A classifier weight, α_i, is assigned to this classifier, based on its accuracy.
3. A weighing vector $\mathbf{D_i}$ is calculated based on the accuracy. Data that are misclassified are weighed more heavily.
4. The training data is re-sampled based on $\mathbf{D_i}$. Data with heavier weights, as they are classified incorrectly in previous iterations, have a higher chance of being selected. The re-sampled data is used to train a subsequent classifier.
5. This process repeats as often as required, until the training error is below some threshold, or an iteration limit k has been reached.

The overall label for an input data point is assigned by taking the weighted sum, based on α, of the individual labels from each classifier.

2.3.2 Bootstrap Aggregation

Bootstrap aggregation (bagging) [31] is a method to reduce overfitting by training a series of base classifiers using a small subset of the full training data, $\mathbf{X}$. A data point is classified by each classifier and the resultant label is decided by majority voting, or by averaging all the smaller classifiers together. The key concept is that if perturbing the training set can cause significant changes in the classifier output, then bagging can improve accuracy. However, if this does not hold, then the bagging algorithm would not result in a more accurate classifier than one that has been trained with the full $\mathbf{X}$.

The training procedure for bagging is as follows:

1. A subset of the total training data, $\mathbf{x_1}$ is randomly selected to be used in the base classifier training.
2. Another random subset of training data, $\mathbf{x_2}$, is drawn from $\mathbf{X}$ with replacement, and is used to train a second classifier.
3. This is repeated k times, creating k classifiers.

The overall label for a given data point is assigned by majority vote over all the individual classifiers.

2.4 Optimal Control Methods

Optimal control is a control strategy that synthesizes control actions which optimize a specified cost function. Conversely, *inverse optimal control* aims to identify the cost function, given an observed system trajectory. In this section, the optimal control problem formulation is presented. Methods to generate the optimal trajectory, as well as methods to recover the cost function given a trajectory, are briefly overviewed.

2.4.1 Optimal Trajectory Generation

A dynamic system can be modelled as a set of states, inputs, and outputs. The relationship between the state variables $\mathbf{x}$, control variables $\mathbf{u}$, and plant outputs $\mathbf{y}$ is described by the following model:

$$\dot{\mathbf{x}}(t) = f_x(\mathbf{x}(t), \mathbf{u}(t)) \tag{2.6}$$

$$\mathbf{y}(t) = f_y(\mathbf{x}(t), \mathbf{u}(t)) \tag{2.7}$$

The objective during optimal trajectory generation is to produce a trajectory $\mathbf{x}^*$ such that a given cost function J is minimized. These methods can be separated into direct optimal control (DOC) and trajectory optimization (TO) methods.

Direct Optimal Control

DOC formulates a controller function that optimizes the specified cost function. Since the controller is closed-loop, it also rejects noise, disturbances, and modelling inaccuracies during operation. Mathematically, this is achieved by finding the optimized control variable $\mathbf{u}^*$ that optimizes the cost function J_u:

$$\min_{\mathbf{u}} J_u(x, u, t); g(\mathbf{x}) \leq 0, h(\mathbf{x}) = 0 \tag{2.8}$$

where $g(\mathbf{x})$ and $\mathrm{h}(\mathbf{x})$ represent inequality and equality constraints, respectively. For lower-body exercises and gait applications, the cost function may include minimizing or maximizing the following:

- Time required to perform the action [172, 153].
- Tracking error [56, 205, 196].
- End-effector position, such as torso incline [37], orthogonal motion from the direction of movement [153], or step length [53].
- Linear velocity [103, 205].
- End-effector velocity [53].
- Joint velocity [103].

- Linear force and acceleration, assuming a point mass model [172].
- Torque, modelled as force exerted by actuating muscles [146], or as force exerted upon a skeletal model [191, 155, 197, 103]. Higher derivatives of torque have also been used [197]. If multiple joint torques are used, the torques may be grouped together by body limb [53].
- Power, the product of force and velocity [56, 38, 103].
- Energy, either by modelling the actuation as muscles, as a function of muscle length and velocity [172, 56, 205], or as heat expenditure [56, 38].
- Postural stability, measured by distance to metrics such as the foot placement indicator [38].
- Control effort, such as muscle activation [146, 206].

The constraints may include:

- Goal-based constraints, such as the starting and ending state values, and/or that the initial and final state must have zero velocity and acceleration.
- Plant-based constraints, such as state (*i.e.* joint range) or control (*i.e.* torque) limits.
- Physics-based constraints, such as the dynamics of the system, such as Equations 2.6 and 2.7.

For linear systems, the optimal controller has the form $\mathbf{u}^*(t) = \mathbf{K}\mathbf{x}(t)$ where $\mathbf{K}$ is the *control gain*, and is known as the linear quadratic regulator [101].

For non-linear control systems, numerous different approaches can be taken to obtain the optimal solution, depending on the system structure and dimensionality [24]. Generally, three categories of solutions are described: dynamic programming (DP), direct methods, and indirect methods [115].

DP methods break down the DOC problem into a sequence of smaller sub-problems based on the Hamilton-Jacobi-Bell equation [101]. However, DP methods scale poorly to higher dimensions [24]. Indirect methods formulate the DOC problem as a set of necessary and sufficient conditions for optimality, such as the Pontryagin maximum principle [24], to locate the point where the gradient of the model is stationary [24]. This means that indirect methods require an analytical form of the gradient to be available, which can be difficult to obtain. The gradient minimization also may not necessarily lead to a minimization in the cost function value, thus indirect methods may have small regions of convergence and require careful initialization [24].

As a result, direct methods are commonly employed over DP and indirect methods [24, 95]. Direct methods solve the DOC problem by minimizing the cost function directly and employ non-linear programming [115]. Direct methods include collocation methods [115], which replace the differential equation to be solved with a simpler equation, such as a polynomial or Taylor approximation, that matches the original on the collocation points. Another common class of solvers in this category are shooting methods [115], which solve small intervals of the differential equation at a time, propagating the error from one iteration to the next via defect constraints. The key difference between collocation and shooting methods lie in how the system dynamics are enforced: shooting methods simulate the full path to explicitly enforce the constraints and thus are more computationally expensive for complex problems, while collocation methods only enforce the constraints at key points so may result in infeasible trajectories [95]. In the literature, both collocation [38, 207, 37] and shooting [191, 155, 197] techniques have been applied to gait generation applications.

However, non-linear DOC is difficult to utilize in real-time control due to the heavy computational costs of the optimization.

Trajectory Optimization

Due to the computational costs of non-linear DOC, it is common for TO to be utilized instead. The objective of TO is to generate the optimized trajectory $\mathbf{x}^*$ which minimizes J:

$$\min_{\mathbf{x}} J(x, u, t); g(\mathbf{x}) \leq 0, h(\mathbf{x}) = 0 \tag{2.9}$$

where the constraints here can be formulated similar to that of Equation 2.8. Since $\mathbf{x}^*$ has been pre-optimized, this technique can be paired with a simpler controller, which can account for model inaccuracies or disturbances. While both DOC and TO can be formulated to minimize the same cost function, DOC produces a closed-loop solution in the form of a controller, while TO produces an open-loop solution in the form of a reference trajectory.

Similar to DOC, many techniques proposed for TO also employ collocation and shooting techniques [24]. A common approach in trajectory optimization is to set up, or transcribe, the target problem into the collocation or shooting framework, then apply a general purpose

NLP solver [95]. Trust region [30], a method where the cost function to be solved is simplified by Taylor approximation and the delta step is expanded or contracted depending if the step successfully obtained a more optimal solution, is commonly employed for this purpose. Interior point [40], a method that converts the constrained optimization problem into an unconstrained problem by adding the constraints into the cost function, is another. Active set [73], a method of turning constraints on and off to simplify the problem, is also common. The MUltiple Shooting CODe for optimization (MUSCOD), a software package that solves optimization problems via shooting, has been used in numerous locomotion [153, 154, 103] applications.

Other implementations of TO include the covariant Hamiltonian optimization for motion planning (CHOMP) [185], a technique that utilized gradient descent, Riemannian metrics, and distance fields to produce feasible and smooth trajectories. CHOMP was extended by stochastic trajectory optimization for motion planning (STOMP) [92], which creates perturbed versions of the trajectory and retains the components that lead to lower total cost. STOMP relies on randomized exploration instead of gradients, improving its ability to escape local minima, as well as allowing STOMP to handle cost functions that do not have a gradient.

2.4.2 Inverse Optimal Control

Inverse optimal control (IOC) is the process of recovering the cost function given the output trajectory. That is, the objective of IOC is to recover J used to generate $\mathbf{x}^*$. In the context of human motion analysis, the cost function is typically assumed to be composed of n_{bf} basis functions J_{bf} and the IOC is designed to recover the weights c_{bf} of these basis functions [154]:

$$J = \sum_{i=0}^{n_{bf}} c_{bf,i} J_{bf,i}$$

When examining its analog from the trajectory generation side, it may be more accurate to describe this technique as inverse trajectory optimization, To be consistent with literature, the technique described in this section will be referred to as IOC throughout the thesis.

Two major approaches are common, utilizing either the direct or indirect methods. The direct method is the *bi-level optimization* approach, where the basis weights are found by minimizing the root-mean-square error (RMSE) between the optimal path generated from the estimated weights and the observed data. Two layers of optimization are employed; one to generate the optimal trajectory given the weights and the task constraints, and the other to generate weights that minimize the RMSE given the trajectory. This method has been used in locomotion [154, 53], reaching [5, 22], and overhead assembly [209] tasks. The bi-level optimization approach is flexible as it does not require the optimization gradient in analytical form, but is computationally demanding as it must both optimize the weights and minimize the RMSE [209].

The second major technique formulates the IOC problem indirectly via optimality conditions, such as the *Karush-Kuhn-Tucker (KKT)* [30] criteria. This method has been applied to locomotion [179, 1, 170, 173] and box moving [63] tasks. KKT-based methods are faster than the bi-level optimization methods as they are only solving for the basis weights and the reconstructed trajectory is only calculated once to quantify the RMSE. However, the gradient must be modelled explicitly, which is not trivial [63].

The type of constraints (Equation 2.8) considered when performing IOC on human motion has varied in the literature. Although the majority of the IOC techniques consider only kinematic constraints [154, 179, 1, 209, 169, 63, 173], some of the prior work also includes dynamic constraints [5, 22, 53], improving the model fidelity. Another common approximation is to model the input trajectory as a spline [209, 1, 169] or radial basis function [63], in order downsample and smooth the input trajectory.

2.4.3 Karush-Kuhn-Tucker Conditions

The KKT conditions are a set of necessary conditions for a given non-linear programming solution $\mathbf{x}^*$ to be optimal. Given a constrained problem:

$$\min_{\mathbf{x}} J(\mathbf{x}); g(\mathbf{x}) \leq 0, h(\mathbf{x}) = 0$$

the KKT conditions state that, if the following conditions are met:

$$\begin{aligned} \nabla_{\mathbf{x}} J(\mathbf{x}^*) &= \sum_{i=0}^{n_f} c_i \nabla_{\mathbf{x}} f_i(\mathbf{x}^*) + \sum_{j=0}^{n_h} \lambda_j \nabla_{\mathbf{x}} h_j(\mathbf{x}^*) + \sum_{k=0}^{n_g} \nu_k \nabla_{\mathbf{x}} g_j(\mathbf{x}^*) = 0 \\ h_j(\mathbf{x}^*) &= 0 \\ g_k(\mathbf{x}^*) &\leq 0 \\ \nu_k g_k(\mathbf{x}^*) &= 0 \\ \nu_k &\geq 0 \end{aligned}$$

where $\nabla_{\mathbf{x}}$ denotes the gradient operator, while λ and ν represent the KKT multipliers for the equality h and inequality g constraints, respectively, then $\mathbf{x}^*$ is optimal.

Chapter 3

Segmentation Framework

This section proposes a framework for segmentation algorithm analysis[1]. The framework provides a structure for considering the factors that must be incorporated when constructing a segmentation algorithm. The lack of such a framework makes it difficult to compare different segmentation algorithms systematically, since different algorithms tend to be assessed against different criteria and with different procedures. The proposed framework provides the means to examine the impact of each algorithm component and allows for a systematic approach to determine the best algorithm for a given situation.

Depending on the target application, different types of segmentation may be required. *Gesture* [151] or *activity recognition* [3, 118] refers to segmentation where multiple repetitions of the same motion type are considered to be one activity or label, and thus segments are declared when the label transitions from one activity to another. The focus of this thesis is on *primitive segmentation*. Motion primitive segmentation is differentiated from activity recognition, as the algorithms are designed to provide more granular temporal detail by segmenting both repetitions of the same motion, as well as transitions between different motion types.

The proposed framework identifies five components that comprise any complete segmentation algorithm (Figure 3.1):

Segment definition (Section 3.1).

There is no common agreed-upon definition of a segment, and thus the segment definition must be clarified for each application.

[1]An early version of this chapter has been published in the IEEE Transactions on Human-Machine Systems [132].

Data collection (Section 3.2).
: Factors such as how the data are collected, as well as the availability of exemplar data or ground truth data for algorithm training and verification, should be determined.

Application specific requirements (Section 3.3).
: Once the source of data and the segment definition have been established, any application specific requirements on inter-participant or inter-primitive generalization, computational cost constraints and scalability requirements need to be defined.

Algorithm design (Section 3.4).
: Once the above factors are determined, the specific segmentation mechanism can be designed. The segmentation algorithm can be divided into four components: filtering and outlier rejection, feature space transformation, segmentation mechanism, and identification mechanism.

Verification (Section 3.5).
: Different validation schemes can serve to emphasize or obscure different aspects of the performance of a given algorithm, and must be selected carefully.

This proposed framework supports a systematic way to construct a segmentation algorithm by ensuring critical details, such as the segment definitions and data sources, are determined early in the design process. The proposed framework also facilitates algorithm comparison and verification.

3.1 Segment Definitions

Segment definitions refer to how the segment boundaries are characterized, to allow a human or algorithmic observer to identify the segment boundaries from the measured data. These definitions are often subjective, algorithm-dependent, application-driven, and tend to fall into three categories:

Physical boundaries.
: For movement applications, the definition of a segment typically refers to physical changes that occur when the movement starts or ends. These natural physical boundaries can be defined by joint movement direction changes [66], or contact changes such as at heel strike [237], during object pickup [229], or during a change in the set of support contacts [142]. These domain knowledge characteristics may be specific to a particular movement (*e.g.*, heel strike during gait) or may generalize to multiple

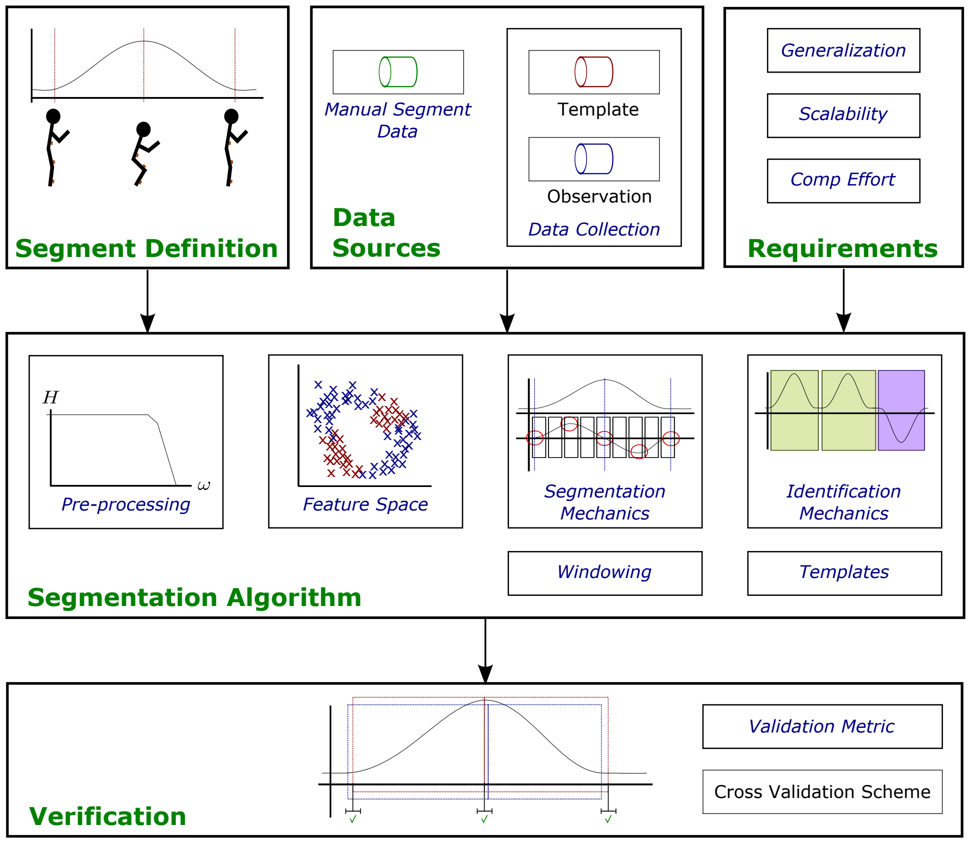


Figure 3.1: Overview of the segmentation framework. The bold labels denote major sections, while italicised labels denote subsections.

motions (*e.g.*, joint movement direction changes). Relying only on contact changes limits the segmentation approach to movements that involve these types of physical interactions. For joint direction changes, it can be unclear which joint should be used, or how to segment if multiple joints are changing directions. Segments defined using joint angle direction changes can separate flexion and extension, or include both in a single primitive (Figure 3.2).

Derived metrics boundaries.

The segment can also be defined by a change in a metric or derived signal. Segment boundaries can be signalled by changes in variance [104], at metric thresholds [18, 96], or at hidden Markov model (HMM) state transitions, as determined by the Viterbi algorithm [216]. Segments can then be determined by either unsupervised [104, 18] or supervised [216] algorithms. Unsupervised algorithms reduce the need for manual data labelling, and can identify segments similar to those denoted by domain experts [66, 41]. However, unsupervised segment identification may be less suitable for rehabilitation applications, where the segments of interest may be clinically defined and may not coincide with the derived metric.

Template boundaries.

Segments can be defined based on a user-provided template. Template-based algorithms include template matching [134], dynamic time warping (DTW) [82], or classifiers [20]. Defining the primitive by a template allows for maximum flexibility, allowing the user to define the template to suit requirements. This approach requires preparation time to generate the templates and generally requires more computationally intensive algorithms.

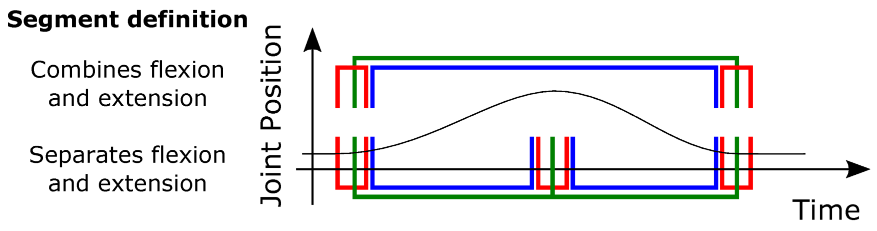

Segment definition
Combines flexion and extension
Separates flexion and extension
Joint Position
Time

Figure 3.2: The location of the segment edge region (red) and the within-primitive region (blue), based on the segment primitive definition used (green), illustrated on a joint angle time series. This image shows two alternate segment definitions: a segment definition with extension and flexion combined (top), and a segment definition where extension and flexion are considered separately (bottom).

3.2 Data Sources

This section summarizes the main approaches for data collection and discusses feature space dimensionality.

3.2.1 Data Collection

Common sensory systems for human motion analysis utilize motion capture systems [156, 219], ambulatory sensors such as IMUs [189, 36], or cameras [152, 240]. Other modalities such as electroencephalogram (EEG) [97, 157] and EMG [50, 193] have been used to study motion in kinesiology and biomechanics studies [226] but have rarely been utilized for segmentation purposes.

Motion capture.

Motion capture systems are considered the gold standard in biomechanics research; they rely on infrared cameras to determine the absolute positions of reflective markers placed on the body, and can be used to calculate joint position or angle data [233, 11]. These multi-camera systems are accurate for collecting gross movement but can suffer from marker swapping and marker occlusions [10, 77]. They tend to be expensive, require unobstructed line-of-sight in the capture space, and are time-consuming to set up, limiting their use to the laboratory. Examples of segmentation algorithms applied to motion capture data include [18, 82, 140, 114, 113].

Cameras.

Video and depth cameras have found widespread usage due to their price and size. Cameras provide light and colour intensity data, and can be used to calculate joint angle data via pose detection [152] and skeleton tracking algorithms [202]. Similar to the motion capture systems, cameras require clear line-of-sight which limits applications to environments where occlusions are not a concern. Segmentation algorithms that rely on camera data include [78, 184, 19, 125].

Inertial measurement units.

IMUs, consisting of accelerometers, gyroscopes, and magnetometers, are lightweight and inexpensive, and measure linear acceleration, angular velocity and orientation with respect to the Earth's magnetic field, respectively. The measured data can be used directly for segmentation, or converted to joint angles via the Complementary [27] or Kalman filter [139, 133]. An IMU-based measurement system makes minimal assumptions about the deployment environment and does not require line-of-sight.

However, IMUs suffer from integrational drift, leading to accuracy issues over time. Magnetometers are also ineffective indoors, where metallic objects, such as steel framing in walls, interfere with the sensors. Segmentation techniques that have been applied to raw IMU data include [20, 47, 125, 236], while techniques utilizing post-processed IMU data include [134].

Force plates.

Force and pressure plates measure the interaction force between an object and the plate and are used to provide dynamic data information. Traditional force plates are immobile, but smaller form factors, such as the Nintendo Wii Balance Board [162], have enabled force plate measurements in a larger range of settings. These modalities provide ground reaction force and centre of pressure measurements, and can be used to calculate joint torque and dynamic parameter profiles [13]. Segmentation algorithms that utilize this modality include [190, 231, 2].

Electromyogram.

EMG measures the electrical impulses signalling muscle contractions. EMG does not have the same environmental limitations as camera-based systems, and does not necessarily need to be on the limb of the joint being measured [211]. For example, since the electrical impulses that control hand movements flow through the forearm, forearm EMG can be used to detect hand and finger movements [175]. The EMG sensors are placed on the forearm and not the hand, so this approach does not impede finger movement or the performance of any hand gestures, allowing for more natural interactions. However, EMG data collection is a function of neuronal and tissue conductivity [225], and water and body fat introduces noise that attenuates the EMG signal [225]. Sensor placement must be consistent between participants for comparison studies and needs normalization to minimize inter-participant variability [164]. Segmentation approaches based on EMG include [94, 44], but many other works examine activity recognition using EMG as well [54, 62, 100, 239, 50, 193].

3.2.2 Manual Segment Data

Manually segmented data is required for training labels or as ground truth for algorithm verification. Various techniques have been employed to obtain ground truth, manual segment, or labelled data. *Training* and *testing data* are used to denote the data series used for training and testing, respectively.

Video playback.

In this approach, the ground truth is generated by having a human observer generate labels by observing video playback of the recorded data. The video can be collected simultaneously with data collection [114, 18] or by playback via regeneration from measurement data, such as animating motion capture markers [134]. An analyst observing the video indicates when segments begin and end, thus introducing subjectivity. Disadvantages of this approach are inaccuracies in the segment points caused by the expert's reaction speed, limitations in the viewing angle while displaying the movement, and effort.

Annotation.

The manual segments can be generated by reviewing the collected data as a time-series graph and annotating directly the regions that contain motions of interest [47, 125]. It takes less time to generate the manual segments than with video playback, but this approach relies on an expert rater that can interpret the time-series data to extract the motions of interest [33].

Proxy sensors.

Ground truth can be generated by a secondary data source. For example, gait cycles can be segmented by locating the impact acceleration [237], when an accelerometer determines that the lower leg posture is parallel to gravity [121], or via a gait mat [52].

Counting primitive occurrences.

It is possible to only identify the number of primitives that occur, either by collecting this information during data collection, or by video playback [236]. This is the fastest method to provide ground truth but also provides no information about the temporal segmentation accuracy of the algorithm.

3.2.3 Public Databases

Several movement databases exist for algorithm training and verification. The use of public databases reduces the effort needed for data collection and allows different algorithms to be compared using the same data. The following databases contain both temporal identification labels, and movement data from multiple participants performing multiple actions:

Brown University HMDB51 Database [112].

The HMDB51 database contains temporally segmented data of various videos available on the Internet, composing of 51 primitives with at least 100 exemplars each.

These primitives include facial actions, full body actions, participant interactions with objects, and multi-person interactions.

Carnegie Mellon University Multimodal Activity Database [57].
The Carnegie Mellon University (CMU) database contains temporally synchronized video, audio, motion capture and IMU data from 26 participants cooking 5 different recipes in a kitchen. This dataset consists of common activities of daily living (ADL), with significant variations in how each primitive is performed. However, the participants were heavily instrumented, which may have impeded natural motion.

Hochschule der Medien HMD05 Database [158].
The HMD05 database contains the motion capture and video data of 5 participants performing various locomotion, object grasping, and exercise tasks. The data is provided in long continuous sequences spanning roughly 3 hours. The data has also been cropped into 1500 motion clips totalling 50 minutes of motion data, composing of 10-50 exemplars for the primitives examined.

Technische Universität München Kitchen Dataset [210].
The Technische Universität München Kitchen (TUM) dataset contains video and motion capture data of 4 participants performing kitchen tasks. The TUM dataset aimed to provide data that contained less intra-primitive variability than the CMU database [210].

University of Glasgow Body Movement Library [141].
The University of Glasgow library contains motion capture data of 30 participants performing locomotion, and arm movements such as knocking and throwing with different emulated emotions. 40 exemplars were collected from each participant. 5 repetitions of the arm movements were performed in sequence, with manual segments provided.

University of Southern California IEMOCAP Dataset [39].
The IEMOCAP dataset contains motion capture, video, and audio data of 10 participants during conversation, spanning over 12 hours of data. The data is segmented to syllable level segments.

University of Tokyo Full-body Movement Dataset [114].
The University of Tokyo (UT) dataset contains video and motion capture data of 1 participant performing 49 different types of full body motions for a total of 751 segments. This dataset provides well-defined full-body movements, but only contains data from 1 participant.

Yale Human Grasping Dataset [34].

The Yale dataset contains video data of 4 participants performing housekeeping and machining tasks over 27 hours of hand movement tasks.

Other movement databases do not provide temporal segmentation data: the University of California [165, 17], the CMU Graphics Lab [43], the Technische Universität Darmstadt [228], the Karlsruhe Institute of Technology [144, 176], and the Kungliga Tekniska Högskolan [218]. A review of databases for computer vision can be found in [49], while motion databases relevant for robotics can be found in [145].

3.3 Application Specific Requirements

The algorithm requirements for the specific application, such as template generalizability, scalability, and computational effort constraints, are considered in this section.

3.3.1 Generalizability

This refers to an algorithm's ability perform on data different from the training set. Two types of generalizability can be considered: (1) participant variability, subdivided into intra-[2] and inter-participant variability[3], and (2) inter-primitive variability.

Participant variability.

Several prior works [216, 134] have examined intra-participant and inter-participant variability. Stochastic techniques, such as the HMM [134], model the variability between exemplar motions inherently. Aggregator techniques have been used for activity recognition applications [110, 12], which can reduce the impact of overfitting from training data, but have only seen limited success in segmentation applications. To date, few techniques successfully generalize between subjects that have very different motion characteristics while performing the same type of motion, such as training on healthy subjects and segmenting on rehabilitation subjects. This is a difficult task due to the differences between the two populations [134].

[2]Data from the same participant but at different instances. Variability is due to effects such as random muscle recruitment and fatigue [188].

[3]Data from a different set of participants not observed during training. In addition to the intra-personal effects, variability is due to effects such as stature and physiological differences [188].

Inter-primitive variability.

Inter-primitive variability considers the case when the training data are obtained from one set of movement primitives, while the testing data consists of a second set of unknown or novel primitives. Algorithms built to segment based on domain knowledge features, such as segments that are defined via velocity characteristics [66, 237], or contact condition changes [244, 229], can be flexible against inter-primitive variability, as they define segment edge points that share common characteristics across all primitives of interest. To reduce the reliance on domain expertise, learning approaches can be utilized to determine these common characteristics (Section 6).

3.3.2 Computational Effort and Causality

Another important consideration is whether the algorithm is capable of running online. This will be influenced by the computational complexity and runtime of the algorithm.

Runtime constraints come from two sources: (1) computational effort for online operation, or (2) the algorithm is non-causal and requires the full observation sequence to be available. The training component of many algorithms tends to be computationally expensive, such as the Baum-Welch algorithm for the HMM [124, 134], or the back-propagation method for the ANN [136]. For the observation component, DTW [184, 82] and the Viterbi algorithm (Section 4.1.1) [216, 218, 15] are two common techniques that require a large computational effort. However, it is difficult to assess computational effort accurately if it is not explicitly reported, since it is a function of both algorithm design and implementation methodology.

Non-causal algorithms can only run offline as they require the full observation sequence before algorithmic processing can begin. Examples of non-causal algorithms applied to motion segmentation include the Viterbi algorithm [216, 218, 15], regression modelling techniques [47], or dimensionality reduction tools [88, 214]. Although the Viterbi algorithm is a non-causal technique, some applications run the algorithm against shorter segments of the observation data instead of requiring the full dataset, or run a truncated version [45, 61], allowing the Viterbi algorithm to be operated online.

Considering computational effort and causality, segmentation algorithms can be separated into three broad categories:

Fully online approaches.

Algorithms that fall into this category may or may not consist of a training phase. If training and template generation is required, it is computationally fast enough to be performed online. The segmentation component is also performed online.

Semi-online approaches.

Algorithms that fall into this category are trained offline, either because the training computational effort is expensive, or full sequences of the training data are required. Once trained, the segmentation algorithm can be performed online.

Offline approaches.

Algorithms that fall into this category may or may not consist of a training phase. If training is required, then the training is performed offline. Once trained, the segmentation is also performed offline, due to computational runtime requirements, or causality requirements.

A related concern to computational effort is scalability. For many algorithms, the computation time does not scale well with increasing feature set dimensionality or the number of templates in the motion library. Data streams used for full-body human modelling can consist of 20-30 DOFs [171]. Algorithms that iterate or rely on dynamic programming, such as HMM or ANN, can become computationally intractable when the feature space is too large, or if there are too many motion types.

3.4 Algorithms Taxonomy

The method for performing the segmentation can be designed after the various constraints in the previous sections have been considered. This section examines different components of the segmentation algorithm, such as any pre-processing or feature space transformations, as well as key algorithm design criteria, such as windowing and algorithm supervision (Figure 3.3). The detailed description of the related work including the description of online, semi-online, and offline approaches can be found in Chapter 4.

3.4.1 Pre-processing

Filtering and Outlier Rejection

Pre-processing of the data is often required to remove sensor noise, due to varying sensor characteristics and limitations in the digitization process [226]. A common procedure is to pre-process the observation data with a properly tuned low-pass [19, 218, 134, 125, 12], or median [218, 12] filter to remove high-frequency noise. High-pass [12] filters have also been employed to reduce the impact of drift. Low-pass filters with a cutoff frequency (f_c) of

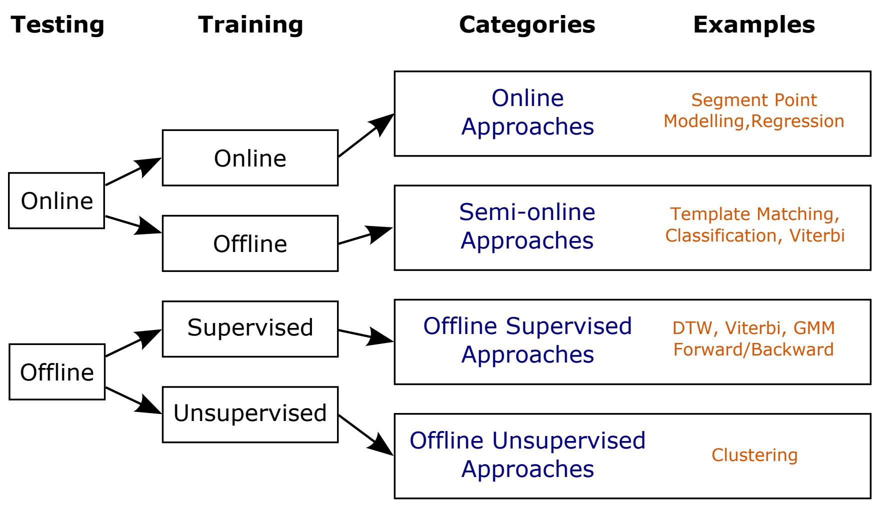

Testing
Training
Categories
Examples
Online
Offline
Online
Offline
Supervised
Unsupervised
Online Approaches
Segment Point Modelling,Regression
Semi-online Approaches
Template Matching, Classification, Viterbi
Offline Supervised Approaches
DTW, Viterbi, GMM Forward/Backward
Offline Unsupervised Approaches
Clustering

Figure 3.3: Overview of the segmentation mechanics.

0.1-4.0 Hz [12, 236, 134, 237] have been applied to IMU and joint angle data. Fast moving signals may require low-pass filters up to the 4^{th} order and a f_c of 6.0 Hz to sufficiently remove noise [226]. For high-pass filters, filters with a f_c of 0.1 Hz have also been applied to accelerometer data [12]. Normalization of the data [137, 19] may also be necessary.

Feature Space

The feature space to be used by the segmentation algorithm is dependent on the type of data available from the data collection phase. The measurement data can be processed to extract proxy features or statistics, or projected into some latent space, to allow for easier processing when compared to the original observation space.

Different feature space manipulation techniques have been used for these purposes, but generally fall into four categories: (1) no transformation, (2) transformation without dimensionality reduction, (3) transformation with dimensionality reduction, and (4) kernel methods. These methods can be characterized based on the resultant DOFs and the complexity of the mapping algorithm (Figure 3.4).

No transformation.

The segmentation is performed directly on the input space. Techniques that use this approach typically rely on data directly from sensors, such as accelerometer signals [236]. This technique requires no pre-processing but can suffer from scaling

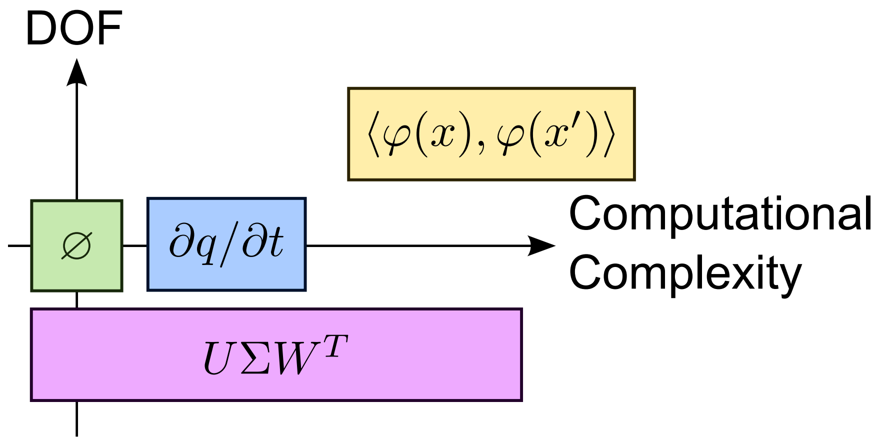


Figure 3.4: Feature space mapping can be conceptualized as a two dimensional space of dimensionality and computational complexity. The origin denotes the original number of DOFs with no transformation (green, $\varnothing$). Transformation methods without dimensionality reduction (blue, $\delta q/\delta t$), methods with dimensionality reduction (purple, $U\Sigma W^T$), and kernel methods (yellow, $\langle \varphi(x), \varphi(x') \rangle$) require increasing computational complexity.

issues, as it is difficult to perform segmentation on high dimensional data due to high computational cost and the existence of correlated and uninformative dimensions.

Transformation without dimensionality reduction.
A transformation of the original features is used as the new representative feature space. The representation retains approximately the same dimensionality as the original feature space. Common techniques include differentiation [66], or the calculation of joint angle data from motion capture [114] and IMUs [133]. Statistical and spectral features, such as mean, or entropy, can be extracted from the data.

Transformation with dimensionality reduction.
The observation data are mapped to a lower dimensional space where segmentation may be easier to perform. This can be achieved using dimensionality reduction tools like PCA [19], feature selection [134], coefficients of frequency transforms [29, 236], or distance metrics [157].

Transformation with dimensionality increase.
Kernel methods map data to an implicit higher-dimensional feature space [45, 61] via the *kernel trick*, where the higher dimension is obtained by taking the inner product between all pairs of data in the feature space. This is computationally cheaper than explicitly determining the higher dimensional coordinate space. This is common in

algorithms that employ SVM [35], to generate a higher dimensional space where the data are better separable. Higher-dimensional embedding [105, 114] is also a way to increase the dimensionality.

Windowing

Rather than considering raw measured data, some algorithms instead use summary statistics computed over windows of measured data. The two main variables for windowing are the size of window, and the amount of overlap between adjacent windows. *Fixed window* algorithms use a sliding fixed-length window, while *variable length window* algorithms employ windows that change their length dynamically to fit the incoming data. Fixed window algorithm performance tends to be sensitive to window length and the amount of overlap between subsequent windows [104, 105]. Overlap between the current window and the subsequent window ranges from 0-50% [7, 134, 12]. A special case of the fixed window approach is a window with the length of one data point [177, 66]. Variable length window algorithms tend to be more computationally expensive as additional computation is required to determine window size. However, the window size is more targeted to the underlying movement which potentially improves the segmentation quality. Typically, variable length windows do not overlap with each other [134, 20].

A Priori Knowledge Requirements

The need for labelled data, and the amount of effort required to generate such labels, separates segmentation algorithms into three distinct groupings:

Unsupervised.
No labelled data are provided to the algorithm and no pre-trained models are generated *a priori* [114, 125]. Unsupervised approaches tend to identify the segment edge points directly by relying on domain knowledge [66] or changes in features [104]. They tend to be computationally faster than supervised algorithms.

Supervised.
Labelled data are provided *a priori* to the algorithm, leading to supervised model training [134, 47]. The key characteristics that describe a segment are determined automatically by the algorithm based on the training data.

Adaptive.

Adaptive solutions update the model online as new data are observed, which allows existing models to be contextualized to the observation data, and potentially increasing segmentation accuracy. Two possible methods to achieve adaptive modelling are: (1) template modification, where an existing primitive model is modified by new observation data [114], and (2) template insertion, where new templates are created from the observations and added to the primitive library [114].

3.5 Verification

Verification techniques are used to determine how well an algorithm performs on a given dataset compared to ground truth. The selection of the verification technique can highlight or obscure the strengths and weaknesses of an algorithm. In the following, it is assumed that ground truth data consists of manual segment edge points provided by the expert rater (Section 3.2.2), while the algorithm being evaluated generates algorithmic segment edge points.

An algorithmic segment edge point is labelled as a true positive (TP) if it corresponds to a manual segment edge point or false positive (FP) otherwise, whereas the absence of an algorithmic segment edge point can be labelled as a true negative (TN) or false negative (FN), if a manual segment edge point is present or absent, respectively, at the corresponding algorithmic segment edge point. This general scheme conforms to common statistical measures of performance, but in some of the assessment schemes, for example, when comparing point pairs, the TN set would result in an empty set. Alternative assessment metrics, such as ones based on shape similarity between templates and observations, may exist, but have not been used for segmentation. In the following, *Acc* is used to denote the different scores used to represent accuracy. Ver is used to denote the verification methods used to calculate the algorithm accuracy.

Two common accuracy metrics, precision ($Acc_{precision}$) and recall (Acc_{recall}) can be computed from the comparison of the algorithmic and manual segmentation labels. Of the labels declared by the algorithm, *precision* reports the percentage of points that received the correct label. *Recall* computes the ratio of correct labels to the total number of labels. These two scores can be aggregated together into the F_1 (Acc_{F_1}) score. These metrics are formulated as follows:

$$Acc_{precision} = \frac{TP}{TP + FP} \tag{3.1}$$

$$Acc_{recall} = \frac{TP}{TP + FN} \tag{3.2}$$

$$Acc_{F_1} = \frac{2 \cdot TP}{2 \cdot TP + FN + FP} \tag{3.3}$$

The balanced accuracy is also utilized in the literature. This metric serves as a measure that aggregates both sensitivity and specificity and limits inflated accuracy scores in imbalanced dataset cases:

$$Acc_{bal} = \frac{1}{2} \cdot \frac{TP}{TP + FN} + \frac{1}{2} \cdot \frac{TN}{TN + FP}$$

The normalized mutual information (Acc_{NMI}) has also seen use in the literature. This metric is a measure of the mutual dependence between two variables using their marginal $H(x)$ and joint $H(x, y)$ entropies, and has been used to compare the algorithmic segment accuracies against the manual ground truth. In this metric, 0 indicates independence, while 1 corresponds to full dependence:

$$Acc_{NMI} = \frac{H(x, y)}{(H(x)H(y))^{\frac{1}{2}}}$$

In addition to these metrics, the allocation of data between training and testing is important. The process of training on one set of data and testing on another set of data is called k-fold cross-validation, where the dataset is divided into k folds, and one fold is used to validate, while *k-1* folds are used to train. In cases where the dataset cannot be easily divided, or there are insufficient samples, leave-one-out cross-validation is employed, where only one sample is used for validation, and the rest are used for training. This technique can be applied to validate the inter-participant variability, where one participant is left out of the training set in leave-one-participant-out (LOPO) validation. It can also be used in leave-one-motion-out (LOMO) validation, where one type of motion primitive is excluded from the training set.

The accuracy of an algorithm can be measured by temporal tolerance or by classification by data point labels.

3.5.1 Temporal Tolerance

One approach to assess algorithmic segment edge points is to declare TPs when they are temporally close to existing manual segments. An algorithmic segmentation point is declared correct if it falls within $\pm t_{err}$ of a manual segment edge point. A FP error is declared if an algorithmic segment edge point was identified when there is not a corresponding manual segmentation point within the $\pm t_{err}$ region. A FN error is declared if a segment edge point was not found algorithmically for a manually identified segment edge point within the $\pm t_{err}$ region. Alternatively, a distance-based metric can be utilized, where a score of 1 is assigned if a manual segment coincides perfectly with an algorithmic segment and decreases towards 0 as the algorithmic segment approaches $\pm t_{err}$ [117]. An inverted version can also be defined where the algorithmic segment edge point is deemed correct if a manual segment edge point falls within $\pm t_{err}$ of it [18].

This verification method is typically used by segmentation algorithms that search for the entire primitive at once or otherwise determine the segment boundaries in a direct fashion and assess the accuracy of the segment boundaries [114, 134] (Figure 3.5).

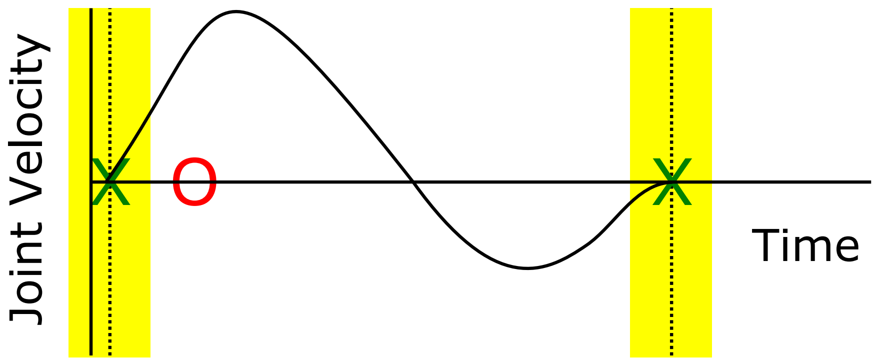


Figure 3.5: An example of the temporal tolerance approach, where the dotted line denotes a manual segment edge point and the yellow region is error tolerance ($\pm t_{err}$). The green (X) algorithmic segment edge points are declared correct and the red (O) is declared incorrect. This algorithm is declared to have an $Acc_{precision}$ of 50%.

This method is sensitive to the selection of t_{err}. t_{err} ranges widely and should be considered in relation to the length of expected primitives. Authors have used t_{err} from 0.2-1.0 s [114, 134, 238]. This method will be denoted as $Ver_{temporal}$.

3.5.2 Classification by Data Point Labels

Instead of only declaring algorithmic segment edge points when they occur, all data points can be assigned a label and compared against ground truth.

This method is less stringent than the temporal tolerance approach. Incorrect segment edge points do not heavily influence the results since there are many more within-segment data points available to smooth out poor segment boundaries (Figure 3.6). Note that, even though this result looks similar to Figure 3.5 in that the resultant segment bounds are in the same place, the accuracy is higher using this verification method, because of how the accuracy is being reported. Although FPs are penalized more lightly, FNs are penalized more heavily. If a segment is completely missed, 3 FNs would be counted in this example, as compared to only 1 FN for the temporal tolerance scheme.

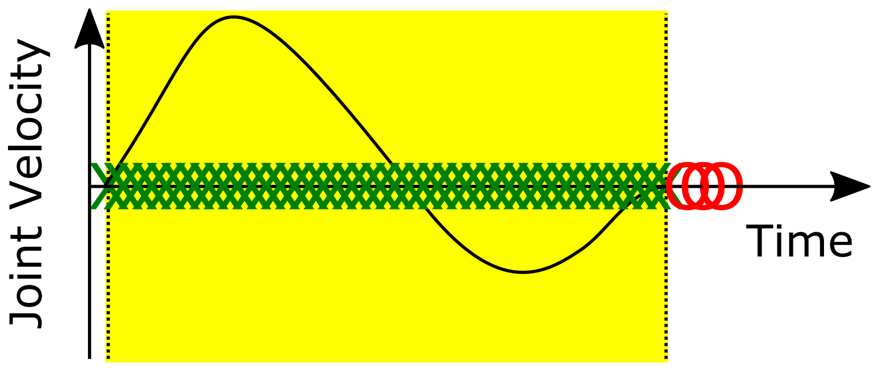


Figure 3.6: An example of the classification by data point labels approach, where the dotted line denotes a manual segment edge point and the yellow region is the correct region for the primitive under examination. Of the 43 labels, 40 are declared correct (green X) and 3 are declared incorrect (red O). This algorithm is declared to have an $Acc_{precision}$ of 93%.

This verification method has been used by segmentation algorithms that label each timestep as a given label and segment when the label changes from one primitive to another [138]. Usually, this is done by assessing windowed data rather than individual points. This method treats time-series segmentation as a classification problem as each data window is assigned a label. Acc_{Class} is defined as the number of data windows with the correct label when compared to the ground truth. In these cases, segmentation and label accuracy are the same. This method will be denoted as Ver_{labels}.

3.6 Summary

Considering applications such as human movement analysis for rehabilitation or imitation learning for robotics, algorithms that detect the start and end points of movement primitives with high temporal accuracy are required. Movement primitive segmentation enables learning detailed motion models for analysis of motion performance and robotic motion imitation.

The proposed framework provides a structure and a systematic approach for designing and comparing different segmentation and identification algorithms. This framework outlines key points of consideration for the segment definition, data collection, application specific requirements, segmentation and identification mechanisms, and verification schemes. The analyzed algorithms can be separated into online, semi-online and offline approaches, and will be detailed in the next Chapter.

Chapter 4

Related Work

This section aims to survey time-series segmentation, and assess the applicability of these algorithms to human motion segmentation[1]. Algorithms are classified based on the framework developed in Chapter 3, separating online, semi-online and offline approaches.

In the event that a given algorithm has multiple components, the algorithm will be classified based on the components that are used directly for segmentation. For example, an algorithm that identifies segment points by using an unsupervised technique such as thresholding, then subsequently uses a supervised technique such as HMMs for labelling, is considered an unsupervised technique for the purposes of segmentation.

4.1 Online Segmentation Approaches

Online algorithms refer to methods that train and segment in an online fashion. These approaches tend to be computationally light, and do not require the full observation sequence to be available before performing segmentation.

4.1.1 Segment Point Modelling

Algorithms in this class are designed to identify segment points, as opposed to recognizing the motion being performed. They tend to be simple, but some prior knowledge of the

[1]A version of this chapter has been published in the IEEE Transactions on Human-Machine Systems [132].

nature of the motion is required. These algorithms do not require template training, and thus are considered unsupervised.

Thresholding on Feature Vectors

Many algorithms declare a segment point when observed features cross a threshold. This method of segment point declaration is simple when only small feature sets are examined. It becomes more difficult with larger feature sets as it becomes more difficult to determine which feature subset to threshold on. A common method of segmentation by thresholding is segmentation by zero crossings [177, 66, 126]. Local maxima and minima can also be thought of as an instance of zero-crossings, since local maxima in one dimension are crossings in the derivative of that dimension.

Many motion-based segmentation algorithms rely on velocity-based features. These methods assume that a change in body link direction denotes a natural segment point. These direction changes are accompanied by local turning points in the joint angle space, resulting in zero-velocity crossings (ZVC). Pomplun and Matarić [177] apply the ZVC concept to motion segmentation to study the effects of rehearsal in motion imitation in humans. The algorithm assumes that motions have clear velocity-crossings to denote the start/end points. To reduce over-segmentation, minimum segment time lengths are enforced. Fod *et al.* [66] segment when the sum of all velocities is lower than some threshold. Lieberman and Breazeal [126] replace the static threshold for ZVCs with a dynamic one, which allows the algorithm to adjust to mean velocity changes and to movements that do not cross zero in joint space.

The velocity crossings concept has been applied in other motion-based applications [41, 74]. Other feature crossings examined include joint acceleration [74, 187], linear acceleration [236], and angular jerk [232]. For movement data, these crossings represent turning points in the movement, and thus serve as a logical segment point.

Despite their popularity, thresholds and zero-crossing approaches suffer from a number of shortfalls. They have a tendency to over-segment, particularly with noisy data or with an increasing number of DOFs [66]. It is difficult to determine which crossing points are actual segment points, or spurious crossings. The threshold value requires tuning, which becomes more difficult as the number of DOFs increases. Additional algorithms such as HMMs [41, 19] or stochastic context-free grammar (SCFG) [232, 74] must be used to provide movement labels following segmentation. Lastly, using thresholds assumes that primitives have well-defined thresholds and crossings in the input space.

Thresholding on Distance Metrics

Derived metrics can be used to denote the degree of separation or difference between two sets of data. When the derived metric reports a value above some threshold between two sequential windows of data, or between a window of data and a template, a segment is declared. Common metrics are the Euclidean distance [157, 75], Mahalanobis distance [18], calculated by normalizing the Euclidean distance by the signal variance, and the Kullback-Leibler (KL) divergence [105, 114], a measure of the difference between two probability distributions. Other distance metrics can be employed, including the generalized likelihood ratio (GLR) [19], and signal variance [104, 105].

Changing contact conditions, which can be considered as a specific case of Euclidean distance or velocity-based segmentation, are also common. These methods include segmenting by surface distance [222] and spatial relation [4] of objects to each other, velocity of body link when contact is occuring [142, 143, 28], and the type of object manipulation task performed [244, 229].

Vögele *et al.* [220] segment by creating a self-similarity matrix, where the distance between the current frame and future frames is calculated. The main diagonal of this matrix is removed, since a given frame will always be the most similar to itself. Segments are declared by finding isolated regions, suggesting that the movement in one region is very different from the next. Similar activities can be clustered for identification.

Bashir *et al.* [19] apply the GLR [8] as a distance metric for curvature data segmentation. The GLR is a statistical test used for hypothesis testing. The null hypothesis is that two different segments were generated from the same model. If the ratio is above some threshold, then the null hypothesis is rejected, and thus a segment should be declared. This method has been applied to ADL accelerometer data [29].

Mandery *et al.* [142] declared segments based on contact to determine different support poses with the floor and environmental elements in gait and object interaction tasks. If an environmental element is closer than a given threshold to the feet, hands, knee, or elbow, and has low velocity for a given length of time, a segment is declared.

Using signal variance as a distance metric is also a common technique. When a feature set suddenly exhibits a large change and the variance becomes very high, it indicates that the underlying motion may have changed to another motion; thus a segment point should be declared. Using variance for movement data segmentation was first proposed by Koenig and Matarić [104].

Instead of calculating the variance directly, different representations of the signal variance can be used. Barbič *et al.* [18] apply PCA to a window of the observation data and retain the top r PCs. They apply this truncated PCA transformation matrix to the subsequent frames of the observation data. The reconstruction error between the pre-transformed data and the post-transformed data is calculated. If the underlying motion changed at time t, the PCA-projected data will differ from the pre-transformed data, causing a spike in the reconstruction error at time t when compared to the error at previous timesteps, and a segment is declared. However, this method is sensitive to the r and error threshold used.

Another alternative to calculating the variance directly is to segment based on changes exhibited by a signal's probability density function (PDF), typically via the HMM and the Viterbi algorithm. The HMM [183] is a modelling technique where the movement data is represented by a stochastic dynamic model. The motion is represented by an evolving unobservable state that obeys the Markov property. Given an HMM, the Viterbi algorithm finds the best state sequence for a given observation sequence.

Kohlmorgen and Lemm [105] use the HMM and an online version of the Viterbi algorithm [183] to perform segmentation. A sliding window is used to calculate the PDF of the windowed data. The PDFs are used to train a HMM. The online Viterbi algorithm is applied to determine the state transitions of the HMM, thus producing segment bounds for a given observation set. This algorithm has been applied to human joint angle data [87]. Kulić *et al.* [114] extended Kohlmorgen and Lemm's algorithm [105] by clustering previously segmented sequences to generate new templates in real-time. Once a new primitive has been identified, the primitive is modelled as a HMM. The novelty of the primitive is determined by its KL distance from existing HMMs. Novel primitives are added into the movement library, whereas non-novel observations are clustered into existing HMMs. A hierarchical HMM can also be employed, where a higher-level HMM contains each primitive as a state to determine the transition between primitive types [113]. This method has also been applied to IMU data [9].

Distance metrics allow segmentation to be performed on a wider range of feature vectors and do not require *a priori* knowledge. Using variance as a distance metric allows these particular algorithms to scale to higher DOFs. Distance-based segmentation shares many of the same weaknesses as the direct feature thresholding approaches. They do not have mechanisms to reject false segments and do not provide segment labels. Tuning is required to determine threshold values [203, 238, 174, 18, 104, 120, 125], or the algorithm is sensitive to the width of the sliding window, or cost function.

4.1.2 Regression

The algorithms examined in Section 4.1.1 model the segment point explicitly. Online regression approaches consider a different conceptual model by modelling the primitive itself.

Keogh *et al.* [96] assume that the observation data can be described by piecewise linear fitting. A large sliding window, sufficient to fit 5-6 segments, is used to window the observation data. This sliding window is divided into small sub-windows and separate linear regression models are calculated from the start of the sliding window to the end of each of the sub-windows. The error for each sub-window is defined as the error between the regression line and the underlying data with a cost function that penalizes long sub-windows, to keep each segment small. The window edge that results in a model with the smallest error that exceeds a defined error threshold is declared a segment. The sliding window is advanced to the end of this segment and this process is repeated. Keogh's algorithm [96] has been applied to segment for gesture recognition, with an HMM for motion identification [7].

Lu and Ferrier [137] use an auto-regressive (AR) model to represent the data over a two-timestep sliding window. When the model from the previous window to the current one is sufficiently different, which is determined by the Frobenius norm, a segment is declared. To reduce over-segmentation, segment points that are close together are removed.

The regression algorithms described above are suitable for segmenting both repetitions of the same primitive, as well as separating different motion primitives. However, these algorithms are very sensitive to parameter tuning, such as the cost function or window size [96]. The resultant regression functions can overfit and do not generate high quality segments. Similar to variance-based approaches, these algorithms do not include methods to reject trivial motions such as tremors and other noise.

4.2 Semi-online Segmentation Approaches

Semi-online approaches encompass methods that require an offline training component but can segment online following the training phase.

4.2.1 Template Matching

Performing segmentation on a single feature's zero-crossings or threshold levels as proposed in Fod *et al.* [66] is often too simplistic and leads to over-segmentation. A sequence of such features can be used as patterns to identify in the observation data. Requiring a sequence of feature thresholds to be matched in a specific pattern reduces over-segmentation in comparison to single threshold approaches.

Kang and Ikeuchi [93] use curve fitting to the volume swept out by a polygon formed by fingertip Cartesian coordinates and hand velocity thresholds to segment grasping tasks. Lin and Kulić [134] propose using sequences of velocity and acceleration crossing points to coarsely locate segment points. HMMs are employed in a fine-tuning step to further reduce over-segmentation. Feature sequence templates and the HMM templates are trained *a priori* from labelled data. Zhang *et al.* [237] use sequences of velocity features to denote heel strikes during gait.

Ormoneit *et al.* [167] examine the signal-to-noise (SNR) ratio of the observation data where the signal is denoted as data amplitude and the noise is the variance. The SNR is used to determine the optimal window for segment searching, and this window length is used to perform curve fitting to pre-made templates.

Feature matching may be able to handle small movement variability but may not generalize sufficiently to large inter-personal variabilities. For example, when healthy templates are used to segment patient templates, the performance is noticeably worse, suggesting generalizability issues [134].

4.2.2 Segmentation by Classification

In many of the algorithms examined so far, a segment bound is declared when the signal passes a threshold, either considering the signal directly, or some distance metric computed from the signal. A separate labelling algorithm is required to classify the motion segments found between each pair of identified segment points. An alternative approach is to label the observation data based on the patterns in the observation vectors, thus transforming the time-series segmentation problem to a multi-class classification problem. This method can be performed by employing sliding windows and pre-trained classifiers [85] and is commonly used in activity recognition contexts [12, 110, 147, 119, 76] where each data point is labelled as a primitive type by the classifier. It has also been applied for primitive segmentation [20, 241].

Berlin and van Laerhoven [20] monitor psychiatric patients using accelerometers on a wrist watch and apply piecewise linear approximation [96]. The slope of the linear segments is converted to angles and binned. The degree of discretization was determined *a priori*, via tuning. Symbols are assigned to sequential pairs of bins, creating motifs. The motifs are generated by inserting training data into a suffix tree, and common chains are used as the motif templates. Segmentation and identification are performed simultaneously by using a bag-of-words classifier.

Zhao *et al.* [241] cluster n-DOF manually segmented data together to create a template dictionary. Each dictionary entry only contains one DOF from the data, resulting in n times more models but less overall computational cost. A time-warping distance feature set is calculated between the training data and each entry in the template dictionary that corresponds to the correct DOF. This feature set is used to train a linear classifier. Observed data are segmented by determining the optimal window via DTW, converted into the distance features, and labelled via the linear classifier.

Classifier approaches reduce the need for domain expertise but are poor at handling temporal variability.

4.2.3 Online Supervised Viterbi

The algorithms described here modify the traditional offline Viterbi algorithm so that it can be operated online, assuming that the model has been trained *a priori*. This approach has been applied to segment human joint angle data as the human guides a robotic arm through a pre-determined trajectory [124, 109].

Castellani *et al.* [45] use pre-trained SVMs to classify sub-tasks during robotic teleoperation tasks. A one-vs-all SVM is used to classify each subtask, which form the HMM states. The SVM hyperplane is translated into a sigmoid function and used as the HMM state emission probability. An online Viterbi algorithm is used to segment the whole data sequence. A 'peg in hole' telerobotic task was used to verify the segmentation accuracy, which consists of several smaller sub-tasks. The state transitions, denoting the change from one sub-task to another, are defined as the segment points. Ekvall *et al.* [61] apply this method to other telerobotic tasks.

Hong *et al.* [78] use the finite state machine (FSM), a deterministic version of the HMM, for video data segmentation. The training data are represented by spatial Gaussians. The number of Gaussians is calculated by dynamic k-means without the temporal data and is done offline. Once the spatial information is segmented, the temporal data are included in the training of the FSM. One FSM is trained for each gesture. When a new observation vector arrives, each FSM decides if the state should be advanced, based on spatial and temporal distances between the observation and the FSM state model. When an FSM reaches the final state, a segment is declared. The approach is verified using a 'Simon Says' system, where the program requests the user to perform a given a task, and the program verifies that the task is performed correctly.

The online Viterbi approach enables online application, but can provide different results than the standard Viterbi algorithm. Ignoring the back propagation component means that the online Viterbi algorithm does not have the full data set to calculate its likelihood values, only the data up to the current timestep, and can result in the algorithm suggesting an incorrect segment, when the standard Viterbi performs optimally, as demonstrated in [45].

4.2.4 Other Stochastic Methods

The Kalman filter has been applied for segmentation purposes. Meier *et al.* [149] constructed multiple dynamical movement primitive (DMP) templates for segmentation. The algorithm assumes that the start of the observation is the start of the first segment, so the segmentation task is to find the end of the movement primitive. It does so using the pre-trained DMPs and uses the Kalman filter to estimate the segment length (τ_i) and posture of the observed segment. The observed segment is identified by an expectation-maximization (EM) procedure that is used in the Kalman filter. Once the elapsed time (t_{curr}) exceeds τ_i, the end of the segment is assumed to be found. The segment is declared, and the algorithm restarts again at $t = t_{curr}$.

4.3 Offline Supervised Segmentation Approaches

Offline methods refer to techniques that perform both training and testing offline. These algorithms, such as the Viterbi algorithm, require the full observation sequence to be available before segmentation. Other algorithms, such as boosted HMMs [140], are too computationally expensive to run online.

4.3.1 Dynamic Time Warping

A major challenge of segmentation is the temporal and spatial variations between the template and the observation. DTW [192] overcomes the temporal variations between motion data sequences by selectively warping the time vector to minimize the spatial error between the observation data and the movement template. DTW has been applied to segment Cartesian gesture data [184], full-body exercise data [217], as well as EEG data [97].

Ilg *et al.* [82] employ DTW in a multi-tier fashion. The observation signal is downsampled by removing all data points that are not at a ZVC. The downsampled points are used to warp to downsampled versions of templates. The sum of spatial error is minimized between the observation and the template to ensure alignment, allowing for segmentation and identification to be performed simultaneously.

DTW-based algorithms are computationally expensive at higher dimensionalities, preventing them from being utilized online. Poor warping can also lead to singularity issues, where large portions of the motion are warped to small portions of the template. The severity of the singularity issue can be mitigated by constraining the warping path [192], or using the derivative of the data instead of the Euclidean distance [97].

4.3.2 Viterbi Algorithm

The Viterbi algorithm overcomes temporal and spatial variations by using an HMM to model each motion template, thus explicitly modelling these variances by the HMM observation variances, and the state transition matrix. The Viterbi algorithm has been used to segment movement data in different applications, such as joint angle data for tele-operative surgeries [216], hand gesture Cartesian data [122], ground reaction force from shoe inserts [190], and joint angle and tactile data for hand grips [21].

Ganapathiraju *et al.* [71] and Vicente *et al.* [218] both use SVM and HMM hybrids in a similar fashion. The training data are used to train SVMs, and the SVM is used to label windows of data. The SVM label sequences are used as the feature vectors for the HMM, and the primitive sequences are represented by the HMM state evolution. The Viterbi algorithm is then employed to determine the segment points.

In computational linguistics, a commonly used technique is the n-gram model, a *n-1* ordered Markov model [208]. Ivanov and Bobick [84] combine the n-gram model with the stochastic context free grammar (SCFG), starting from the current position of the observation data and hypothesizing the possible continuations of the input by tracing down branches on the SCFG tree. The observation input is then compared against expected states and a likelihood of state advancement is generated. When a grammar branch is exhausted, a primitive may have been completed, and the Viterbi algorithm denotes the state path taken and thus the segmentation result.

Baby and Krüger [15] apply HMM merging to improve Viterbi performance. Given an existing HMM template λ_M, new observation data are formulated into a new HMM $\lambda_{\mathbf{c}}$, and merged into $\lambda_{\mathbf{M}}$ by merging similar states between $\lambda_{\mathbf{c}}$ and $\lambda_{\mathbf{M}}$. If there are states in $\lambda_{\mathbf{c}}$ that do not have a corresponding state in $\lambda_{\mathbf{M}}$, these states are inserted as new states into $\lambda_{\mathbf{M}}$. Once all the observation data are merged into the $\lambda_{\mathbf{M}}$, the Viterbi algorithm is used to trace through the motions, and common paths are removed from the Viterbi paths via the longest common substrings method until no common paths exist between any components. Each of these components becomes a primitive; segments are detected on switches between components. This approach was applied to human interactions with objects from the object's point of view [16].

Chamroukhi *et al.* [47] segment movement data employing multiple regression models and segmenting on model switch. The observation data are represented by a regression model, $y_i = \beta_{z_i} t_i + \epsilon_i$, where the regression coefficient β_{z_i} is a function of the logistic hidden state z_i. z_i controls the switching from one activity to another, for k different activities. That is, the regression model describes a different motion according to the state of z_i, while the logistic model captures the higher level stochastic dynamics of the transitions between motions. When the state of z_i changes, a segment point is declared. The parameters of the regression models and $\mathbf{z}$ are trained by the EM algorithm. The segments are produced by estimating z_i at each y_i in a similar fashion to the Viterbi algorithm.

Although the usage of the Viterbi algorithm is widespread, it suffers from several key issues. It is expensive to use, and requires the full observation sequence to be available. The Baum-Welch and the Viterbi algorithm are also local optimizations, so the solutions provided may not be globally optimal. For the HMM, the modelled data are assumed to be Gaussian, which does not hold for human movement in general [161]. Tuning is required to find the suitable number of states to represent the model and to prevent overfitting.

4.3.3 Gaussian Mixture Models

Gaussian mixture models (GMMs) are parametric PDFs represented as the weighted sum of Gaussians. For segmentation, the boundaries between each Gaussian are used to denote the segments. The number of Gaussians needed for the GMM is typically determined *a priori* [18]. GMMs have been used to segment exercise data [18], and for imitation learning [123].

Like the HMM, GMM modelling assumes that the modelled data are Gaussian, or near Gaussian in nature. The number of Gaussians needed to model the data requires some degree of tuning; the Bayesian Information Criterion (BIC) can be used to assist the tuning effort [123].

4.3.4 Forward/Backward Algorithm

This algorithm is a technique for determining the likelihood that a given sequence of observation data is generated from a given HMM [183] and is typically used for primitive identification. However, it has also been applied to primitive segmentation.

Wilson and Bobick [224] utilize pre-trained parametric HMMs of hand pointing gestures. A fixed-length sliding window is used on the observation data, and the EM algorithm is used to estimate the parameters of the parametric HMM over the windowed data. The corresponding likelihood value is determined by the forward/backward algorithm. Windows that result in a high likelihood are declared segments.

Lv and Nevatia [140] use HMM templates as classifiers in an AdaBoost algorithm. The observation data are split into two windows, with the first window starting at some minimum length l_{min}, and increasing at each iteration. The two windows are run through the AdaBoost classifiers, and the window length that results in the highest likelihood is selected, forming a segment at l_{maxLL}. The algorithm is run multiple times, with the starting point of the first window advancing to the end of the previous segment at each run.

Both of these methods incur a large computational cost for both training and segmenting and cannot be used online.

4.4 Offline Unsupervised Segmentation Approaches

Offline unsupervised methods are techniques that do not require labelled data to be provided *a priori*, but the algorithms either require the full dataset to be available or have large computational costs.

An approach that belongs to this category assumes that the observed data evolves according to an underlying deterministic model that has been contaminated with time warping and additive noise. Probabilistic methods can be used to approximate both the parameters of the underlying model and find the segmentation locations. Chiappa and Peters [51] estimate the underlying model, warping terms and noise model via EM. This approach requires the full sequence for action fitting, making it unsuitable for online applications.

Zhou *et al.* [242] use an extension of DTW to produce similarity measures between two temporally aligned segments. Given an initial set of segments, this measure is used in a kernel k-means framework to determine segment cluster centres and to assign each segment to a cluster. For each segment, a search is used to determine the segment boundaries that would result in minimal distance between the segment and its cluster centre. These two steps are repeated to iteratively converge to a solution.

Lan and Sun [117] model motion data as a written document with unknown topics (the motion), composed from a vocabulary of words (key poses). Hierarchical clustering is used to extract key poses, then all data frames are discretized to these key poses. Latent Dirichlet allocation, a topic discovery generative modelling technique, is used to group the key poses into motion primitives. A sliding window is used to calculate between-window topical similarity, and a segment is declared using a threshold. Newly observed key poses and primitives can be incorporated to update the model.

Fox *et al.* [69] examine multiple sets of time series data simultaneously to extract global movement characteristics over all movements. Individual time series are assumed to exhibit only a subset of these characteristics, over certain lengths of time. These characteristics and behaviours are modelled as autoregressive HMMs (AR-HMMs), trained by a Markov Chain Monte Carlo process, and can be thought of as features of the movement. Features that describe a given time-series data are selected via a beta process model. Segments are declared when the time series shift from one AR-HMM to another, signifying a shift in the underlying movement.

Wächter *et al.* [221] propose a segmentation method that divides the trajectory when the Euclidean distance between two objects is below a threshold as a coarse segmentation method. Once segmented, a hierarchical approach is employed, where a metric calculated from jerk normalized by its peak-to-peak acceleration amplitude is proposed. Between each existing segment point, a candidate segment point is inserted. The metric value of the window to the left and the right of the candidate is calculated as the score of that candidate point. The candidate point is then incremented in time. The candidate point with the highest score is declared as a segment. This approach is repeated with all pairs of segment points recursively while enforcing segment length and score thresholds. This method serves as a way to capture both segment edge point characteristics, as well as avoid the need for pre-determined segmentation templates.

Murali *et al.* [159] perform segmentation on video as outputs from neural networks, as well as kinematic features, over several stages. The video features are dimensionally reduced, and augmented with the kinematic features. The augmented features are then coarsely segmented via GMM clustering [111]. These segments are then refined by clustering with different subsets of the features, as well as time. Different trajectories of a given primitive are then examined to see if the segment point candidates appear in the same location over several different trajectories.

Although most distance and threshold-based algorithms are computationally light enough to be computed online, some approaches employing computationally expensive derived features can require offline implementation. Isomap, a dimensionality reduction technique, has been combined with thresholding on Cartesian maxima [88, 89] and joint angle crossing points [214] to segment.

4.5 Validated Algorithms

A majority of the approaches examined do not report their verification methods or scores, report only the identification accuracy instead of the segmentation accuracy, or do not provide a complete set of validation parameters, making it difficult to compare between methods. Methods that reported some form of validation results are summarized in Tables A.1, A.2 and A.3.

For online approaches, the highest temporal accuracy was reported by Barbič *et al.* [18], with an Acc_{F_1} of 93%. This paper segments by calculating the Mahalanobis distance between a window of data against subsequent windows, and thresholds on distance peaks. The distance metrics scale well to higher dimensions as the segmentation focuses on a single feature and is lightweight to use. This method successfully separated action sequences consisting of 7 different primitives, but is sensitive to the tuning parameters, such as the window length and the threshold. The primitives examined consist of highly different full-body movements, such as jumping, walking, kicking, or punching, which proved to be dissimilar enough for the distance metric, but may be difficult to generalize to primitives that are very similar to each other, or primitives that are not very correlated [18].

For semi-online approaches, the highest temporal accuracy was reported by Lin and Kulić [134], with an Acc_{F_1} of 85%. They applied a two-tier algorithm where pre-trained velocity peak and crossing templates provide a set of segment candidates from the observation data. The candidates are passed into HMMs where the forward algorithm is used to verify the segments. They tested with both individualized velocity/HMM templates, as well as generalized templates. HMM-based methods scale well to higher dimensions but require long training times, which need to be completed *a priori*. However, velocity features rely on relatively simple primitive types, and may not scale well to complex or unsteady movements [134].

For offline approaches, none of the examined algorithms reported temporal segment accuracy. The highest reported label identification accuracy was reported by Chamroukhi *et al.* [47], with an Acc_{class} of 90%. They modelled the observation data as a set of regression models and switched between the models via a Viterbi-like algorithm. This method was applied to various postures and ADL. The Viterbi algorithm shows high accuracy, scales well to higher dimensions, but requires both training and testing to be carried out offline, which needs to be completed *a priori*.

4.6 Outstanding Problems

While many approaches have been proposed for motion primitive segmentation, active areas of research remain.

4.6.1 Algorithm Verification and Public Databases

The majority of the algorithms examined do not explicitly report segmentation accuracy, in part due to the difficulties of obtaining labelled data. Testing algorithms with publicly available datasets with labelled data is recommended as that would provide both a common ground to compare different algorithms, as well as reduce the amount of post-processing work that researchers must do to carry out algorithm verification.

Currently available databases (Section 3.2.3) tend to focus on healthy populations, omitting populations which may have significantly different movements. These alternative populations would provide a wider spectrum of data for inter-participant testing and are particularly important for rehabilitation applications.

4.6.2 Inter-participant Generalizability

Inter-participant generalizability remains an outstanding problem. Although techniques such as HMM can be applied to generate multiple-participant templates and account for spatial and temporal variabilities, only a few algorithms generate such templates and test against multiple participants [134, 216]. Applications such as physical rehabilitation often do not have access to patient movement data *a priori* and thus must rely on template generalization.

If the segmentation algorithm is to be applied to participants of different demographics or capabilities, large variations can be expected, and pose a significant challenge for algorithm generalizability. This problem is especially significant in cases where few training samples are available.

4.6.3 Inter-primitive Generalizability

Inter-primitive generalizability also remains an outstanding problem. A few algorithms, such as those reliant on domain-knowledge [66], classifiers (Section 6), or parametrized models [224] provide potential solutions, but do not tend to explicitly explore or report inter-primitive generalizability.

Generalizability is an important issue in rehabilitation since the exercises are typically modified slightly in order to suit the patient's capabilities, and an algorithm that can provide some degree of generalizability would increase the utility of any given algorithm. In exercise applications, movements that vary in only direction should be considered the same movement, but may pose a challenge for the algorithm [224], and thus require techniques that are robust to these types of variability.

Generalizability is also an important concern for applications where the motions to be performed are not known *a priori*, such as in online human machine interaction. The existing segmentation work can be divided into techniques that model the primitive explicitly, or techniques that model the segment point directly. Techniques that model the segment point directly using domain-knowledge [66] or learned automatically via classifiers (Section 6) provide the means to segment based on common characteristics over all the primitives of interest and warrant further investigation.

4.6.4 Dependence on Sensor Modality

A majority of the examined algorithms were verified with a single sensor modality or feature type. The lack of validation with multiple modalities reduces the clinical applicability of these segmentation algorithms, where an algorithm that is capable to handling different types of features can improve real-life deployment sucess if the sensing environment is different from clinic to clinic.

Algorithms that have been applied to multiple modalities do not tend to rely on any specific feature of a given modality and typically model the motion or segment using abstracted features. These methods include the HMM-based approaches applied to both motion capture and camera data [15, 16], classification on joint angles, EMG, and force plates (Section 6), and DTW on joint angle and Cartesian data [184].

Algorithms that have been tested with multiple datasets include algorithms using curvature data [19], the Viterbi algorithm [114, 113], AR modelling [137], and DMP [149, 150].

Chapter 5

Experimental Datasets

To validate and analyze the performance of the algorithms developed in this thesis, several movement datasets were collected and used:

1. Lower-body rehabilitation movements from healthy subjects, collected at the University of Waterloo (UW), described in Section 5.1 [134].
2. Whole body exercise movements from a publicly available dataset from the University of Tokyo (UT), described in Section 5.2 [114].
3. Squat movements from healthy subjects from the University of Rome Foro Italico (URFI), described in Section 5.3 [26].
4. Hand gesture movements from healthy subjects, collected at the University of Waterloo in collaboration with Thalmic Labs (UWT), described in Section 5.4.
5. Lower-body rehabilitation movements from knee or hip total joint replacement inpatients, collected at the Toronto Rehabilitation Institute (TRI), described in Section 5.5.
6. Lower-body rehabilitation movements from knee or hip total joint replacement outpatients, collected at the St. Joseph's Health Centre Guelph (SJHCG), described in Section 5.6.

5.1 University of Waterloo Dataset

The UW dataset consists of 30 (17M, 13F, $\mu_{age} = 24.4$) healthy participants performing 10 to 20 repetitions of 14 lower-body exercise motions [133, 134]. This dataset serves as a key benchmark as it provides a large amount of healthy movement data. A key characteristic of this dataset is the consistency in both range of motion and velocity between participants, even though the only instructions given to the participants were that the motions should be executed in a controlled fashion. Unlike some of the later datasets, there is no ambiguity in the number of repetitions performed, and thus the manual segments are of high quality. The dataset was collected over two different sessions.

The first component consists of 20 repetitions of 5 rehabilitation motion types from 20 healthy participants. It was collected via 3 Shimmer IMU sensors [36], worn at the right ankle, knee, and hip, transmitting at 128 Hz. Simultaneous motion capture data were also collected for a 10-marker lower-body system via Motion Analysis cameras [156], transmitting at 100 Hz. The exercises performed are outlined in Table 5.1.

Table 5.1: First set of exercises in the UW dataset.

Abbreviation	**Motion**	**Initial Posture**
HFEO-SUP	hip extension	Supine
KEFO-SIT	knee extension	Sitting
KHEF-SUP	knee/hip flexion	Supine
SQUA-STD	squats	Standing
STSO-SIT	sit to stand	Sitting

The second component was collected to match the TRI exercises (Section 5.5). It consists of 10 repetitions of 9 rehabilitation motion types from 10 healthy participants. The data was collected via 3 Shimmer IMU sensors [36], worn at the right ankle, knee, and hip, transmitting at 51.2 Hz. Motion capture data were also collected using a 10-marker lower-body system via Motion Analysis cameras [156], transmitting at 50 Hz. The exercises performed are outlined in Table 5.2.

The features computed from this dataset are:

- 3 DOF planar joint angles from motion capture via inverse kinematics.
- 5 DOF 3D joint angles from IMU via Kalman filter and forward kinematics [133].

Table 5.2: Second set of exercises for the UW dataset.

Abbreviation	**Motion**	**Initial Posture**
ARRT-STD	Ankle raise	Standing
GAIT-STD	Walking	Standing
HAAO-STD	Hip adduction	Standing
HAAO-SUP	Hip adduction	Supine
HEFO-STD	Hip extension	Standing
HFEO-STD	Hip flexion	Standing
KFEO-STD	Knee flexion	Standing
KHEF-STD	Knee/hip flexion while standing	Standing
LUNG-STD	Lunges	Standing

For this dataset, manual segments of the exercises were generated and labelled by an expert watching the motion in video playback of motion capture data that was collected simultaneously. All data were collected with the approval of the University of Waterloo Research Ethics Board and signed consent was obtained from all participants.

5.2 University of Tokyo Dataset

The UT dataset consists of 1 healthy participant performing an average of 20 repetitions of 45 full-body exercise motions [114]. This dataset also serves as a key benchmark, as it is the only dataset that includes a mixture of different primitives performed in sequence. One key issue with this type of movement sequence is that the subject may blend two primitives together, if they do not complete one primitive fully, pause, then begin the next primitive. This dataset is a publicly available [114].

The dataset was collected as a 26-marker full body system via motion capture. Video data was also collected, so the movement labels are unambiguous. The exercises included in this dataset are outlined in Table 5.3.

The features used from this dataset are:

- 20 DOF joint angles from inverse kinematics [114].

The dataset also contains manual labelling of the segment edge points of all performed motions, obtained via video playback [114]. The data was collected in compliance with the research ethics requirements of the University of Tokyo.

Table 5.3: Exercise set for the UT dataset. The different components of each exercise are grouped together by the overall motion. For example, both arms to 180°, arms raising (BAR180) and arms lowering (BAL180) are grouped together here.

Abbreviation	**Motion**
BAL180, BAR180	Both arms raise to 180°, lower
BAL90, BAR90	Both arms raise to 90°, lower
BAD, BAU	Bowing
LAL180, LAR180	Left arm raise to 180°, lower
LAL90, LAR90	Left arm raise to 90°, lower
LKAL, LKAR, LKE, LKR	Left kick
LPAL, LPAR, LPE, LPR, LPUAD, LPUE, LPUR	Left punch
MLL, MLR, MRL, MRR	Marching
RAL180, RAR180	Right arm raise to 180 °, lower
RAL90, RAR90	Right arm raise to 90°, lower
RKAL, RKAR, RKE, RKR	Right kick
RPAL, RPAR, RPE, RPR	Right punch
SQD, SQU	Squat
WWL, WLR, WRL, WRR	Walking in spot

5.3 University of Rome Foro Italico Dataset

The URFI dataset consists of 8 ($\mu_{age} = 30.2$) healthy participants performing 20-30 repetitions of squat motions [26]. The squats were performed at a self-selected speed, with the starting and ending posture kept consistent. The participants were asked to keep their arms straight along their sides, while the feet remained flat on the ground.

The data was collected via VICON cameras [219] at 100 Hz. Simultaneous force plate data was also collected via a Bertec force plate [23].

The features used from this dataset are:

- Joint position data from motion capture.
- Planar 7 DOF joint angle data from motion capture via inverse kinematics [26].
- Ground reaction force and centre of pressure from force plate.
- Joint torque data from joint angle and force plate data computed via inverse dynamics [99] using dynamic parameters obtained from and anthropometric tables [60].

Manual segment data were generated and labelled by time-series annotation. The data was collected in compliance with the research ethics requirements of the University of Rome Foro Italico and signed consent was obtained from all participants.

5.4 University of Waterloo and Thalmic Dataset

The UWT dataset consists of 10 ($\mu_{age} = 26.8$) healthy participants performing 9 different types of hand gestures. Two datasets were collected. The first set consists of 5 repetitions of 9 different hand gestures (individual gesture (IG) dataset). The second set consists of continuous motions where the 9 primitives are performed in random order (continuous random (CR) dataset). All motions started at the hand-open resting posture, then moved into the gesture, then back to the resting posture. No specific directions on how to perform each motion were given to the participants, so some inter-participant variabilities were observed in how each gesture was executed. Paddle out was used as the sync motion. Only 6 of the 10 participants contributed randomized motion sequences.

The data was collected via a 8 channel forearm surface EMG using a Thalmic Myo [211] at 100 Hz, placed on the working forearm of the participant, without any specific instructions on armband location or orientation. Kinematic data was also collected by a Measurand Shapehand dataglove [148] at 100 Hz. The dataglove provides 15-channel joint angle data, corresponding to each joint in the fingers. The hand gestures performed are outlined in Table 5.4.

Table 5.4: Hand gestures for the UWT dataset.

Abbreviation	**Motion**
FISP	Finger spread
FIST	Fist
GUNM	Gun
PDIN	Paddle in
PDOU	Paddle out
POIN	Pointing with index finger
POIM	Pointing with index and middle finger
SNAP	Finger snapping
THPK	Thumb-pinky touch

The features used are:

- 8 DOF EMG data from the forearm EMG. From the EMG data, a large number of typical EMG and signal processing features can be computed. These features considered were selected due to their prevalence in EMG analysis [225, 106], for their ability to extract useful information from a noisy signal, as well as their channel agnostic characteristics to reduce the impact of EMG sensor placement variability. The features examined include RMS EMG, mean absolute value, waveform length, slope sign changes, skewness, kurtosis, channel-pair inner-product, channel-pair angle, RMS ratio and peak-to-peak time between the two most active EMG channels, Hurst exponent, Hjorth parameters, Teager energy, entropy, relative entropy, mutual information, peak frequency, band width, peak width as measured at quarter power point from the peak frequency spectral power, and relative spectral power.

- 15 DOF joint angles from the dataglove.

The experiment was approved by the University of Waterloo Research Ethics Board and signed consent was obtained from all participants.

5.5 Toronto Rehabilitation Institute Dataset

The TRI dataset consists of 18 (4M, 14F, $\mu_{age} = 73.8$) lower body total joint replacement (TJR) in-patients, performing post-operative rehabilitation exercises. The patients were tracked from the first day of admission until discharge, with the average patient's treatment lasting 5.7 days. The patients engaged in rehabilitation every day during the course of their treatment, and data was collected during each weekday session, for roughly an hour per session. The patients were also instructed to perform exercises outside of these sessions, but these unsupervised exercises were not captured. All the exercises collected were part of their prescribed rehabilitation treatment; no exercises were modified or added for the data collection. This dataset of in-patient data is likely the most challenging dataset for validation. In-patient data consists of numerous pauses in movement, as the patient cannot complete the motion without significant pain. Range of motion and velocity are also significantly lower, which can make it difficult for healthy data templates to generalize to this dataset.

The dataset was collected via 3 Shimmer IMU sensors [36], attached by Velcro straps to the hip, knee and ankle of the patient. The patient was asked to verify that the straps were not uncomfortable or hampered their movement before the data collection began. A wide variety of motions were collected from TRI, and are outlined in Table 5.5.

Table 5.5: Exercises for the TRI dataset.

Abbreviation	Motion	Initial Posture
HAAO-STD	Hip adduction	Standing
HAAO-SUP	Hip adduction	Supine
HEFO-STD	Hip extension	Standing
HFEO-STD	Hip flexion	Standing
HFEO-SUP	Hip extension	Supine
KEFO-SIT	Knee extension	Sitting
KEFO-SUP	Knee extension	Supine
KEFO-SUP	Knee extension	Supine
KFEO-STD	Knee flexion	Standing
KHEF-STD	Knee/hip flexion while standing	Standing
KHEF-SUP	Knee/hip flexion	Supine
SQUA-STD	Squats	Standing
STSO-SIT	Sit to stand	Sitting

The features used are:

- 5 DOF 3D joint angle data, computed from IMUs via Kalman filter and forward kinematics [133].

Manual segment data were generated and labelled by time-series annotation. This data was collected with the approval of the University of Waterloo Research Ethics Board, and the University Health Network Research Ethics Board. Signed consent was obtained from all participants.

5.6 St. Joseph's Health Centre Guelph Dataset

The SJHCG dataset consists 26 (10M, 16F, $\mu_{age} = 66.7$) lower body TJR out-patients [116], performing post-operative rehabilitation exercises. A single session of the out-patient exercise regime was collected from all participants, spanning roughly an hour per session. These out-patients may have had several sessions of in-patient care prior to out-patient visits, or were only provided with home rehabilitation. When compared to the in-patient population, the out-patient population is generally able to perform the exercises with less supervision. This dataset of out-patient data serves as an intermediate point between healthy data and in-patient data. The out-patients do not have the full functional range and velocity of healthy subjects, but they are generally much healthier and in less pain than in-patients. The exercises performed at SJHCG were similar to those performed at TRI, and can be found in Table 5.5.

Many of the exercises performed by the SJHC and TRI patients were modified in some way: springs, slings, weights, and other physical devices were added to modify the exercise difficulty, providing test cases for of inter-exercise generalization. The main difference between the two patient datasets was the health status of the patients. In-patients were being treated in hospital due to additional health concerns or co-morbidities that prevented discharge, while out-patients were healthy enough to recover at home.

The features used are:

- 5 DOF 3D joint angle data computed from IMUs via Kalman filter and forward kinematics [133].

Manual segment data were generated and labelled by time-series annotation. This data was collected with the approval of the University of Waterloo Research Ethics Board, and the St. Joseph's Health Centre Guelph Research Ethics Board. Signed consent was obtained from all participants.

5.7 Summary

To validate and analyze the performance of the algorithms developed in this thesis, several motion datasets collected from both healthy and rehabilitation participants performing various exercises were employed. These datasets span over 90 participants, including 44 rehabilitation patients. They also span 5 types of sensor modalities: motion capture, IMU, force plate, and EMG.

Chapter 6

Segmentation based on Segment Point Modelling

6.1 Introduction

This chapter develops an algorithm for online motion segmentation[1]. A classifier approach for segmentation is proposed, where machine learning techniques are used to automatically extract segment edge characteristics. All data points are classified as either a *segment edge point* p_1 or a *non-segment point* p_0, converting the difficult task of temporal segmentation into a simpler data classification task. If segment edge points share common features which are equivalent between different exercises, such as changes in velocity directions or changes in contact conditions, classifying all data points into either a p_1 or p_0 point allows the algorithm to handle motions that have not been observed before, and be applied to novel participants and exercises. This reduces the number of templates that a clinician must provide in order to operate the algorithm, and allows the optimal classification boundary to be determined automatically without the need for hand selected features.

[1]Components of this chapter have been published in the International Conference of the IEEE Engineering in Medicine and Biology Society [130], the IEEE-RAS International Conference on Humanoid Robots [129], the EAI International Summit of Smart City 360 [135], and the NIH-IEEE Strategic Conference on Healthcare Innovations and Point of Care Technologies [127]. A journal paper is currently under review at the Journal of Rehabilitation and Assistive Technologies Engineering [131].

6.2 Proposed Approach

In this section, a classifier for discriminating between p_1 and p_0 points is developed (Figure 6.1), evaluated at each time-step of observed motion data. Several unique problems arise for segmenting via classifiers [85]:

Appropriate generation of segment edge point training data.
In order to provide training data to the classifier, exemplars of p_1 and p_0 points need to be provided. Ground truth generated from expert observation typically provides a single time point to denote the start or end of a segment. For a classifier, this is not a suitable approach, as there is likely minimal difference between the data at a given time t_n and the data at time t_{n+1} near the declared segment edge point. The data described by the p_1 points denotes characteristics such as joint angle turning points, and these characteristics are shared with more data points than the ones denoted by the manual p_1 points.

Integration of temporal information.
Typical classifiers also do not account for temporal effects, which can hurt segmentation performance, as it is an integral part of movement data.

Unbalanced datasets.
Unbalanced datasets are also an issue, where there are significantly more data samples of a given label when compared to the other labels [234]. For example, in a given set of training data, data points labelled as a p_1 are uncommon when compared to p_0.

To address these issues, normalization, manual segment point expansion, input vector stacking, and downsampling are used, as described below. This procedure is used to prepare the data for both classifier training and observation classification.

6.2.1 Pre-processing

Normalizing

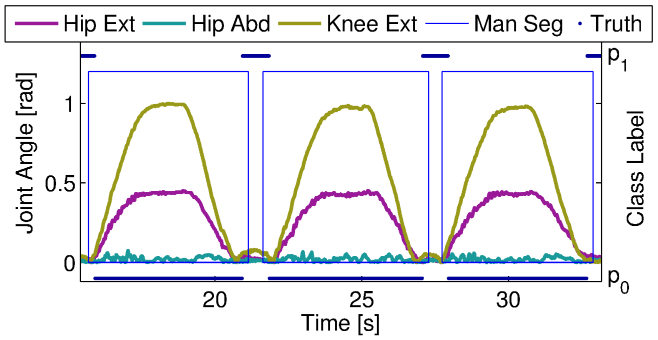


Figure 6.1: A time-series waveform of joint angles during a squatting exercise. This figure shows the hip extension (hip ext), hip abduction (hip abd), knee extension (knee ext), the manual segments (man seg), and p_1/p_0 ground truth data generated from the manual segments (truth). The segments provided by a human observer (blue boxes) are expanded into multiple p_1 points (top blue lines) and p_0 points (bottom blue lines). Data points between exercises are p_1 points, while data points within exercises are p_0 points.

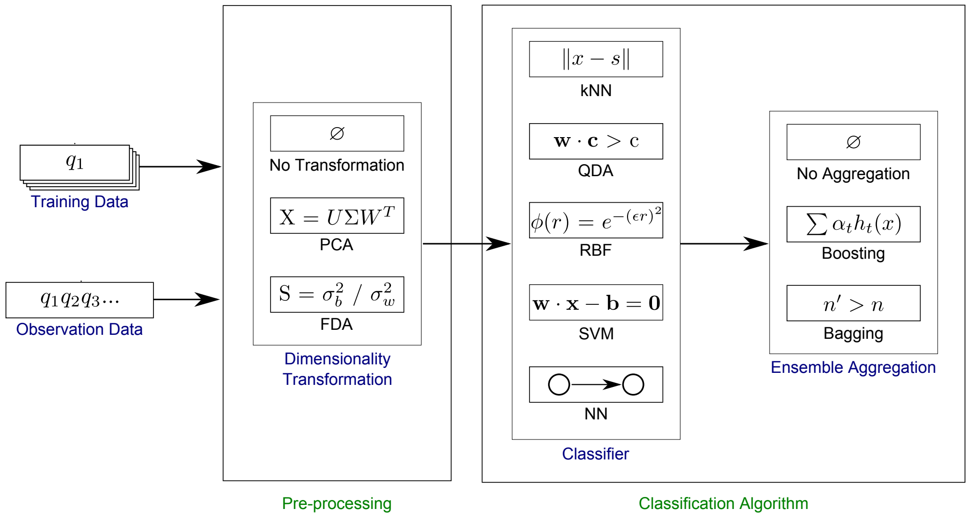
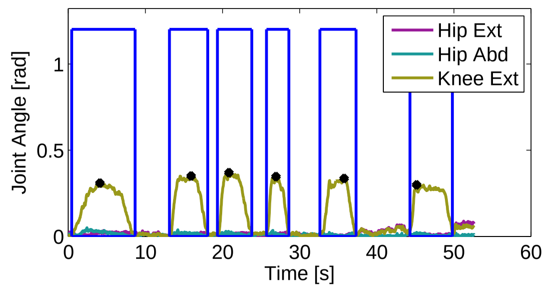
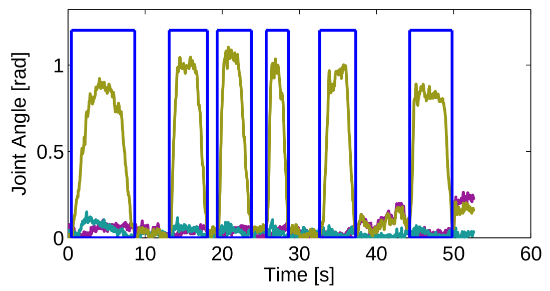


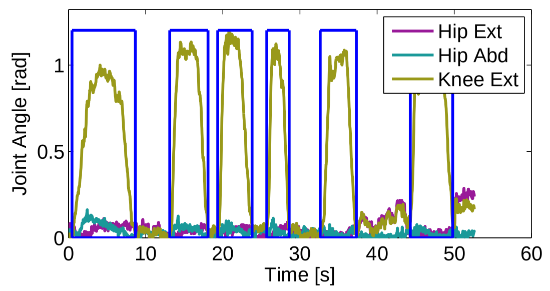
Figure 6.2: Flow diagram for the classifier-based segmentation algorithm. A given instance of the algorithm will consist of a dimensionality transformation component, a base classifier, and an aggregation component.



(a) Unnormalized data. Peak points of all segments highlighted (black dots).



(b) Normalized data by sampling from all 5 segment windows, used for algorithm training.



(c) Normalized data by sampling from the first segment window, used for algorithm testing.

Figure 6.3: Knee extension data from the TRI dataset showing magnitude scaling normalization. These figure denote the original unnormalized data (top), as well as the same trajectory after normalization where the value for normalization is derived from averaging over all the peak points (bottom left) or only the first one (bottom right).

Initial value removal.

The data can have its initial value removed. Most rehabilitation motions follow the pattern of starting from a resting posture, moving into a maximal flexion or extension pose, then returning to the resting posture. However, the exercises may start from a wide range of initial poses, depending on the posture of the participant and the equipment used. For example, supine heel slides are a common rehabilitation exercise, but the initial posture could be with the leg flat against the bed, or held up by a sling. Therefore, the initial value of each time-series can be removed such that the resting pose is always at zero.

Rectification.

The data can also be rectified. Taking the absolute value of the joint angle data would allow motions that involve the same joints but moving in opposite directions to be recognized as one exercise, improving generalization.

Magnitude scaling.

Magnitude scaling, by dividing by the mean of the rectified peak values over all the segments in the time-series data, compensates for the varying joint angle ranges between healthy and patient data, as well as enables exercise generalization. However, the determination of this value requires the full observation set, which is impractical in clinical applications because the full observation sequence is not available *a priori*. For training, the scaling factor is calculated from the full dataset, while for testing, the scaling factor can be calculated from only the first segment (Figure 6.3).

For EMG, a normalization coefficient can be calculated in a variety of different ways: (1) channel-wise normalization, where the coefficient is the maximum magnitude of each channel in each gesture, (2) motion-wise normalization, where the coefficient is the maximum magnitude over all the EMG channels, or (3) participant-wise normalization, where the coefficient is the average of the maximum motion-wise magnitude of different trials of the sync motion.

Position normalization.

For certain modalities, further normalization may be necessary. EMG data require placement normalization to reduce the impact of the inter- and intra-participant variability resulting from variations in sensor placement and EMG signal magnitude. For EMG data, a known sync motion is collected for baseline purposes. Sensor placement normalization is carried out by finding the appropriate rotations such that the maxima of the EMG norm of the sync motion are always in a given channel, and applying this rotation to all movements by this participant in a given session.

Temporal

Data points are interpolated to ensure even temporal spacing between time points, then data points that are n_{stack} samples around the time point being examined (t_n) are concatenated into the feature vector to incorporate temporal information before and after t_n into the classifier. That is, the input feature vector into the classifier at time t_n is $d(2n_{stack}+1)$ wide, where d is the dimensionality of the original feature set, creating a feature vector $\mathbf{t_{n,use}} = [t_{n-n_{stack}} \cdots t_{n-1}, t_n, t_{n+1} \cdots t_{n+n_{stack}}]$.

Balancing

Imbalanced datasets can skew the classifier towards the class that has the most samples. To minimize the impact of this problem, data points n_{exp} around manual segment points, as well as the data points between segments, are converted into p_1 points, as they are similar to the manual segment point. This also increases the number of p_1 points available for training and classification.

During training, downsampling is also applied to p_0 to further balance the dataset. A Gaussian sampling method was employed, where p_0 points closer to p_1 points have a higher chance of being sampled than p_0 points that are not close to any p_1 points. This approach was used to include more data points that are close to the segmentation boundaries and thus are more likely to be misclassified (Figure 6.4). During testing, no downsampling is applied.

6.2.2 Segment Definition

In this thesis, the segment is defined by the user. Two types of segments are defined and examined in this thesis. The first type is the full segment, which defines the flexion and extension of a motion as a single primitive, while the second type, the half segments, defines the flexion and extension of a motion as two different primitives (Figure 3.2).

6.2.3 Feature Extraction

After the pre-processing steps, a feature extraction technique can be applied. The primary purpose of the extraction is to reduce the feature space dimensionality and decrease computational time. The feature embedding from Section 6.2.1 can increase the dimensionality

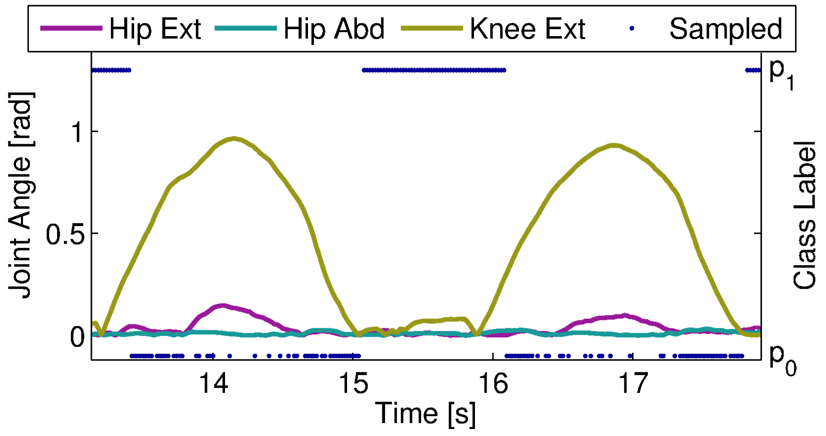


Figure 6.4: Knee extension training data, denoting the hip extension (hip ext), hip abduction (hip abd), knee extension (knee ext), and the sampled data for training (sampled). The blue points at $y = 0$ denote the training data selected for p_0, sampled by Gaussian resampling, favouring non-segment points closer to the boundary. The blue points at $y = 1$ denote the training data selected for p_1, which has no downsampling. The Gaussian covariance coefficient is reduced to show this effect more clearly.

of the system significantly, depending on the choice of n_{stack}. An extraction technique that can reduce the dimensionality, and thus the computational time, is desirable.

6.2.4 Discriminative Classifier

The data can now be used for training and classification. Classifiers can only handle classes that have been seen during training, but this is not a limiting factor for the proposed approach, as all data points can only be either p_1 or p_0. Different classifiers can be used for this purpose and are further explored in this chapter. The different combinations for the various algorithm components that are compared are as follows (Figure 6.2):

1. Dimensionality reduction
   - No transformation applied
   - Principal component analysis [85] (PCA)
   - Fisher's discriminate analysis [85] (FDA)
2. Classifier

- k-nearest neighbour [85] (k-NN)
- Quadratic discriminate analysis [85] (QDA)
- Radial basis function network [168] (RBFN)
- Support vector machine [35] (SVM)
- Artificial neural network [86] (ANN)

3. Aggregation
    - No aggregation algorithm
    - Boosting [70]
    - Bagging [31]

6.2.5 Analysis via the Segmentation Framework

This algorithm can be analyzed via the segmentation framework proposed in Section 3.

Segment definition.

The segment definition is specified by domain experts in physiotherapy. These generally consist of either an extension and a flexion movement defined together as one segment, or the two components defined separately.

Data collection.

The algorithm is designed to be applicable to a variety of sensing modalities. The primary sensor modality used during validation are IMUs, with the derived feature set of joint angles. Other sensor modalities examined include motion capture, EMG, and force plates.

The source population for the training are healthy participants performing exercise data. The target population for the testing are both healthy participants and rehabilitation patients. The manual segment data are sourced from human observers from video playback and annotation.

One public dataset with manual segments was used, while several other datasets were collected.

Application specific requirements.

To meet the target application of physiotherapy applications, the algorithm must be able to handle inter-participant and inter-primitive variability. There are no training time requirements, but testing time should occur online.

Algorithm design.

The training and testing data are handled differently based on the sensor modality and feature set. The sensor data may be used as-is as the feature set, such as motion capture joint position or force plate GRF. Alternatively, techniques such as extended Kalman filter, inverse kinematics, or inverse dynamics can be used to convert the data to joint angles or joint torques.

This algorithm is categorized as a sliding window semi-online supervised segmentation technique. There is an offline training component that calculates the transformation matrix for a dimensionality reduction component, as well as the template data for the statistical classifier. Both of these components can test incoming data in an online fashion. The algorithm classifies between segment-edge and within-segment time points. This algorithm has no segment identification component.

Verification.

The validation set includes both healthy and rehabilitation datasets, to reflect the target population. The validation employs leave-one-out validation when the dataset is small and partitioning if the dataset is large. $Acc_{precision}$, Acc_{recall}, Acc_{F_1}, and Acc_{bal} accuracy measures are used with classification by data point labels to produce the segmentation accuracy.

6.3 Experiments

Several sets of experiments were performed to verify the algorithm in order to examine the impact of different classifier methods, normalization techniques, and feature sets. 12 sets of experiments were carried out over 6 datasets and can be broadly grouped into 3 different categories. For each experimental category, the dataset used, the algorithm settings, and the validation configurations are detailed in Tables 6.1 and 6.2, and are summarized below:

Classifier testing (Section 6.3.1).

The first set of experiments employs 2 healthy datasets over 4 different experiments to test the segmentation accuracy of different dimensionality reduction techniques, classifiers, and aggregators (Figure 6.2). These experiments were conducted to identify the best performing dimensionality reduction, aggregation and classifier configurations.

Normalization testing (Section 6.3.2).

The next set of experiments examined the impact of feature normalization using 3 different datasets. While many algorithms were designed with clinical applications in mind, few to date have been tested with patient populations or report specific segmentation results. Motion data from patient populations can vary greatly from healthy data. Segmentation algorithms should be able to generalize from healthy data to rehabilitation data as healthy data can be collected in a supervised environment and be labelled. The use of healthy data only for training is preferred, to avoid the need for manual labelling of clinical data. The impact of normalization is evaluated with this set of experiments.

Feature testing (Section 6.3.3).
To investigate the suitability of the proposed algorithm to different sensor modalities, the final set of experiment compares the performance using different sensor modalities and features using 2 different datasets. Algorithm flexibility to multiple sensors allows the method to be deployed more easily in different sensing environments.

All processing and algorithm implementation were done in MATLAB [212], along with the libsvm Toolbox [48], the Toolbox for Dimensionality Reduction [215], and the Bayes Net Toolbox [160].

Table 6.1: Overview of experiments conducted. The experiment sets (E) are grouped by their categories: classifier (C), normalization (N) and features (F). The features examined are joint angles (q), joint positions (x), GRF (F), COP (C), and joint torque (τ). Normalizations (norm) considered are initial offset removal (O), rectification (A), and magnitude scaling (M). The number of data participants (n_s) and primitives (n_p) are denoted, as well as the number of cross-validation (CV) sets performed.

Name		**Features**		**Training**		**Testing**		
E	**Dataset**	**Features**	**Norm**	n_s	n_p	n_s	n_p	**CV**
C	UW Individual	q	O	1	5	1	5	20
	UW Grouped			5	5	5	5	4
	UT Full			1	9	1	19	1
	UT Half			1	23	1	45	1
N	UW Testing	q	A, M, O	10	5	20	8	4
	SJHCG Testing			-	-	26	7	-
	TRI Testing			-	-	18	10	-
F	UWT Normalization	EMG	M	10	9	6	9	2
	UWT Classifier			10	9	6	9	2
	UWT LOMO			1	8	1	1	9x10
	UWT LOPO			9	9	1	9	6
	URFI LOPO	x, q, F, C, τ	A, M, O	7	1	1	1	8

Table 6.2: Overview of experiments conducted. The experiment sets (E) are grouped by their categories: classifier (C), normalization (N) and features (F). The different dimensionality reduction (DR), classifier, and aggregator configurations considered are listed, as are the n_{exp}, n_{stack}, and sampling rate ([Hz]) used.

Name		**Classifier**			**Temporal Factors**		
E	**Dataset**	**DR**	**Classifier**	**Aggregator**	n_{exp}	n_{stack}	[Hz]
C	UW Individual	None,	k-NN, QDA, RBFN,	None,	25	15	128
	UW Grouped	PCA,	SVM, ANN	Bagging,			
	UT Full	FDA		Boosting	4	15	33.3
	UT Half						
N	UW Testing	PCA	SVM	-	10	0, 5, 10, 15, 30	50
	SJHCG Testing						
	TRI Testing						
F	UWT Normalization	PCA	Threshold	-	5	15	50
	UWT Classifier	PCA	k-NN, QDA, RBFN, SVM, ANN				
	UWT LOMO	PCA	ANN				
	UWT LOPO	PCA	ANN				
	URFI Testing	PCA	SVM	-	0, 2, 5, 10	15	50

6.3.1 Classifier Experiments

This set of experiments examined the impact of varying dimensionality reduction, classifier, and aggregator components to determine suitable methods for motion segmentation, and tested against 2 healthy datasets.

Datasets

The first dataset examined in this set of experiments was the UW dataset, using the joint angles and velocities calculated from IMUs. The UW dataset contains a large number of participants data performing the same exercises, and thus serves to test the algorithm's ability to handle inter-participants variability.

The second dataset examined in this set of experiments was the UT dataset, using the joint angles and velocities calculated from the motion capture data. The UT dataset contains a large number of primitive types, and thus serves to test the algorithm's ability to handle different types of primitives.

Algorithm Settings

The dimensionality reduction, classifier, and aggregators that were varied in this experiment set are summarized in Figure 6.2.

For the PCA and the FDA, the optimal number of PCs to include was determined by the scree plot method, with the threshold set to 80%. For k-NN, k was set to 3 or 9. The kernels for the soft-margin SVM considered were linear, polynomial, radial. A feedforward ANN was examined, with three different layers and neuron configurations: $[10, 10]$, $[10, 10, 10]$, $[20, 20, 20]$. For boosting, AdaBoost [70] was employed. However, the AdaBoost algorithm is formulated to work with classifiers that can accept weighted data points, which does not apply to all classifiers. Instead, a resampling scheme, based on the data weights, is employed [199]. 3-stage and 5-stage boosting and bagging methods were tested.

Validation Configurations

A total of 4 different experimental configurations were used in the classifier experiments. 2 experiments used the UW dataset to explore intra- and inter-participant segmentation accuracy. 2 other experiments were conducted using the UT dataset to explore the impact of different segment definitions and the algorithm's ability to classify similar primitives, as well as its ability to generalize to novel primitives.

The first experiment investigated intra-participant segmentation accuracy using the UW dataset (Table 6.1, UW Individual), where the classifier was trained from 5 primitives from 1 participant, and tested with the other half of the available data of the same participant. The results were averaged over all 20 participants. The second experiment investigated inter-participant accuracy using the UW dataset (Table 6.1, UW Grouped), where 5 participants formed the training data, while the testing dataset consisted of the 5 other participants. A 4-fold cross-validation was performed.

The third experiment investigated the classification and generalization accuracy while using full segments from the UT dataset (Table 6.1, UT Full), where the last 10 segments of each action primitive were used for training, over 9 different primitives. The last experiment investigated the classification and generalization accuracy while using half segments from the UT dataset (Table 6.1, UT Half), where 23 different primitives were used. All primitives over the whole dataset were then used to test the segmentation accuracy of the proposed algorithm.

The validation metrics used in this section were the $Acc_{precision}$, Acc_{recall}, and Acc_{F_1} scores. The Acc_{F_1} score is a metric that combines precision and recall, and is designed to reduce the impact of bias in unbalanced datasets. The labels assigned are as follows:

Table 6.3: Label conditions for the classifier error verification method.

Category	**Condition**
TP	An algorithmic p_1 point matches up with a manual p_1 point, or if an algorithmic segment point falls within $\pm t_{err}$ of a manual segment edge point.
TN	An algorithmic p_0 point matches up with a manual p_0 point.
FN	An algorithmic p_0 point matches up with a manual p_1 point, or if an algorithmic segment point is missing from within $\pm t_{err}$ of a manual segment edge point.
FP	An algorithmic p_1 point matches up with a manual p_0 point, or if an extra algorithmic segment point is found outside of $\pm t_{err}$ of a manual segment edge point.

For the temporal assessments in this experiment, t_{err} is set to 0.2 seconds unless otherwise specified. This time was chosen as human reflexes averages to 0.25 seconds [107], thus as long as the algorithmic segment edge point is declared within 0.2 seconds of the manual segment edge point, it can be considered as being online. Clusters shorter than n_{exp} in length are removed. Each cluster is then converted into an ending point for the n^{th} segment, and a starting point for the $n+1^{th}$ segment. These segment edge points are declared n_{exp} from the two edges of the cluster, or one-third of the length of the cluster, if the cluster is shorter than n_{exp} in length. The Acc_{F_1} scores of the temporal clusters were assessed.

Two types of Acc_{F_1} score was used. The standard definition of the Acc_{F_1}, denoted here as $Acc_{F_{Seg}}$, is defined as follows:

$$Acc_{F_{Seg}} = \frac{2 \cdot TP}{2 \cdot TP + FN + FP}$$

$Acc_{F_{Seg}}$ does not include TN points in its definition. To include the p_0 points, a second definition of the Acc_{F_1}, the $Acc_{F_{Class}}$ was introduced:

$$Acc_{F_{Class}} = \frac{2 \cdot (TP + TN)}{2 \cdot (TP + TN) + FN + FP}$$

6.3.2 Normalization

This set of experiments examined the impact of varying normalization methods for motion segmentation. All training data from this section are drawn from healthy data in order to represent real-life situations where labelled healthy data are easily obtainable, while labelled rehabilitation data are difficult to find. Testing was done on both healthy and rehabilitation data. A total of 1 healthy and 2 rehabilitation datasets were used.

In addition, an analysis into the PCA projection and phase plots of healthy and rehabilitation data to compare the differences between unnormalized data and normalized data was performed. This analysis provides an overview of the impact of the feature manipulation via n_{stack} and serves as a way to obtain insight into the features that PCA is selecting. Lastly, a comparison to existing algorithms is also conducted in this section.

Datasets

The healthy dataset used in this set of experiments was the UW dataset, using joint angles and velocities calculated from IMUs. The UW dataset provides a large number of healthy motion data and forms the main training set for this set of experiments.

The rehabilitation datasets used in this set of experiments were the SJHCG and TRI datasets. These two datasets cover 44 rehabilitation patients with wide ranging health conditions, from shortly post-surgery and requiring constant physiotherapist supervision, to out-patient status and requiring only occasional supervision. These two datasets serve to validate the algorithm's effectiveness with the intended target population.

Algorithm Settings

The normalization methods that were varied in this experiment set are summarized in Figure 6.5.

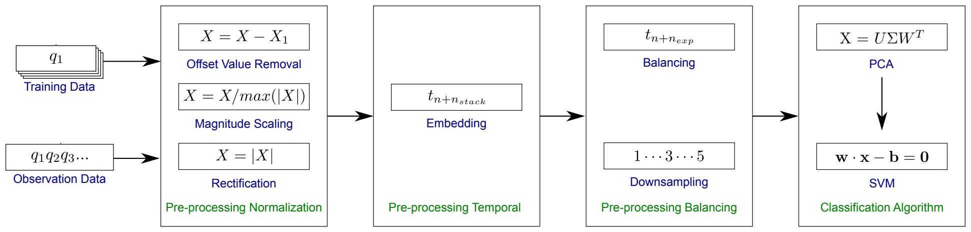


Figure 6.5: Flow diagram of the experimental routine used for normalization testing. This experiment set varied the normalization techniques: offset value removal, magnitude scaling, and data rectification.

For PCA, a scree plot threshold of 80% was used. For SVM, 2-class soft margin radial SVM was used. C was set to 1. For the radial kernel function, γ was set to the inverse of the number of DOFs post-PCA.

Validation Configurations

A total of 3 experimental configurations were used in the normalization experiments. All 3 experiments used the UW dataset to train a template consisting of 5 participants from the first UW dataset, and 5 from the second UW dataset, with 5 different primitives in total, and were tested on the remaining UW (Table 6.1, UW Testing), SHJHC (Table 6.1, SJHCG Testing), and TRI (Table 6.1, TRI Testing) data. In all experiments, no training data overlapped with the testing data. 4-fold cross validation was performed. The testing was separated into classifying known motions and generalizing to novel motions. The primitives selected was selected by totalling the amount of rehabilitation data available, and selecting the top 5 exercises that had exercise examples from all three datasets (Table 6.4).

Table 6.4: Known (K) and novel (N) exercises used for normalization testing, with their corresponding active joints, direction of movement, typical initial postures, and the total amount of data collected. The directions of motion include extension (ext), flexion (flex), and abduction (abd). The initial postures include supine (sup), sitting (sit), and standing (std).

	Names		**Characteristics**			**Data Collected [s]**		
	Long [102]	**Short**	**Joints**	**Direction**	**Posture**	**UW**	**SJHCG**	**TRI**
K	Standing hip abduction	HAAO-STD	Hip	Abd	Std	832	976	3535
	Standing hip extension	HEFO-STD	Hip	Ext	Std	799	1945	7734
	Seated knee extension	KEFO-SIT	Knee	Ext	Sit	1207	1327	8119
	Marching	KHEF-STD	Knee/Hip	Ext/Flex	Std	836	1934	6892
	Supine heel slides	KHEF-SUP	Knee/Hip	Ext/Flex	Sup	952	515	21277
						4625	6697	47558
N	Standing hip flexion	HFEO-STD	Hip	Flex	Std	799	-	3817
	Quad over roll	KEFO-SUP	Knee	Ext	Sup	-	238	10954
	Standing hamstring curls	KFEO-STD	Knee	Flex	Std	-	2921	7118
	Resisted knee flexion over roll	KFEO-SUP	Knee	Flex	Sup	733	-	6102
	Squats	SQUA-STD	Knee/Hip	Ext/Flex	Std	1370	-	6002
						2902	3159	33993

The validation metrics used here were the $Acc_{precision}$, Acc_{recall}, and Acc_{bal}. The metrics were changed from the previous experiments as using two different measures, the $Acc_{F_{Seg}}$ and $Acc_{F_{Class}}$, produced some redundancy. Acc_{bal} provides an accuracy score that accounts for both TN and TP scores equally, is suitable for unbalanced datasets:

$$Acc_{bal} = \frac{1}{2} \cdot \frac{TP}{TP + FN} + \frac{1}{2} \cdot \frac{TN}{TN + FP}$$

6.3.3 Features

This set of experiments examined the impact of varying sensor modalities and feature sets to assess algorithm flexibility performance with various input modalities. 2 healthy datasets were tested.

Datasets

The first dataset examined in this set of experiments was the UWT dataset, using different signals calculated from the EMG. EMG serves as an interesting feature to examine since motor neuron signals serve as the control signal of the body, which can potentially signify motion before it is noticeable in joint angle space.

The second dataset examined in this set of experiments was the URFI dataset, using both kinematic (joint position and angles from motion capture) and dynamic (joint torque, ground reaction force, and centre of pressure from force plate) data.

Algorithm Settings

For the UWT dataset, different algorithm combinations were considered, and are detailed as follows: for the PCA, the scree plot method was used, with the threshold set to 80%. The classifiers considered were k-NN, SVM, ANN, LDA, QDA, and thresholding. For k-NN, k was set to 1 or 3. The kernels for the soft-margin SVM considered were linear, polynomial, and radial. A feedforward ANN was examined, with three different layers and neuron configuration: [10], $[10, 10]$, $[10, 10, 10]$, $[20, 20, 20]$. For LDA, QDA, and thresholding the Euclidean distance was used.

For the URFI dataset, PCA SVM was used. The dimensionality retained by PCA was automatically determined by scree plot method, retaining 80% of the variance. Radial, soft-margin SVM was used. The testing data was multiplied by the PCA warping matrix and classified using SVM.

Validation Configurations

A total of 5 different experimental configurations were used in the feature experiments. 4 experiments used the UWT dataset to explore suitable configurations for EMG datasets. The last experiment used the URFI dataset to explore segmentation accuracy using different sensor modalities.

The training data for the UWT experiments were always drawn from the IG dataset, while testing was always done with the CR dataset. The first experiment investigated EMG normalization (Table 6.1, UWT Normalization), since EMG data show high inter-personal variability and require normalization to generalize. Secondly, different classifiers was examined to determine the one that produces the best segmentation result (Table 6.1, UWT IG Classifier). Lastly, leave-one-motion-out (LOMO, Table 6.1, UWT LOMO) and leave-one-participant-out (LOPO, Table 6.1, UWT LOPO) cross-validations were performed, where one motion and one participant was removed from the training set, respectively.

The last experiment investigated different sensor modalities using the URFI testing set (Table 6.1, URFI Testing). It was performed by training on 7 participants and testing on the remaining participant in a LOPO scheme.

The validation metric used here was the Acc_{bal} score.

6.4 Results

Table 6.5: Overview of the results for the experiments conducted for the classification-based algorithm. The experiment sets (E) are grouped by their categories: classifier (C), normalization (N) and features (F). The features (feat) examined are joint angles (q), RMS + pairwise inner-product (RP), mean average value (MAV), and joint position (x). The dimensionality reduction (DR), classifier (cla), and aggregator (agg) selected are also listed. The normalization (norm) examined are initial offset removal (O), magnitude scaling (M), and rectification (A). The accuracy metric (met) denote the training (tra), classification (class) and generalization (gen) accuracy refer to intra-primitive classification and inter-primitive generalization.

Name		**Features**		**Classifier**			**Validation**			
E	**Dataset**	**Feat**	**Norm**	**DR**	**Cla**	**Agg**	**Met**	**Tra**	**Class**	**Gen**
C	UW Individual	q	O	PCA	SVM	-	Acc_{F_1}	97	97	-
	UW Grouped	q	O	PCA	SVM	-	Acc_{F_1}	97	95	-
	UT Full	q	O	PCA	k-NN	Boosting	Acc_{F_1}	100	100	97
	UT Half	q	O	PCA	SVM	Bagging	Acc_{F_1}	99	95	88
N	UW Testing	q	AMO	PCA	SVM	-	Acc_{bal}	-	91	87
	SJHCG Testing	q	AMO	PCA	SVM	-	Acc_{bal}	-	89	88
	TRI Testing	q	AMO	PCA	SVM	-	Acc_{bal}	-	85	83
F	UWT Normalization	RP	M	PCA	ANN	-	Acc_{bal}	-	79	-
	UWT Classifier	MAV	M	PCA	ANN	-	Acc_{bal}	-	83	-
	UWT LOGO	MAV	M	PCA	ANN	-	Acc_{bal}	-	-	76-87
	UWT LOPO	MAV	M	PCA	ANN	-	Acc_{bal}	-	72-91	-
	URFI Testing	x	AMO	PCA	SVM	-	Acc_{bal}	-	97	-

6.4.1 Classifier

UW Dataset

Three sets of results were generated:

1. Training results, where the Acc_{F_1} scores for the data used for the classifier training were assessed.
2. Classifier results, where the Acc_{F_1} scores for the full observation data for primitives of the same type as the training data were assessed.
3. Temporal results, where each cluster of p_1 points is converted into a single temporal segment edge point by clustering the p_1 and p_0 points.

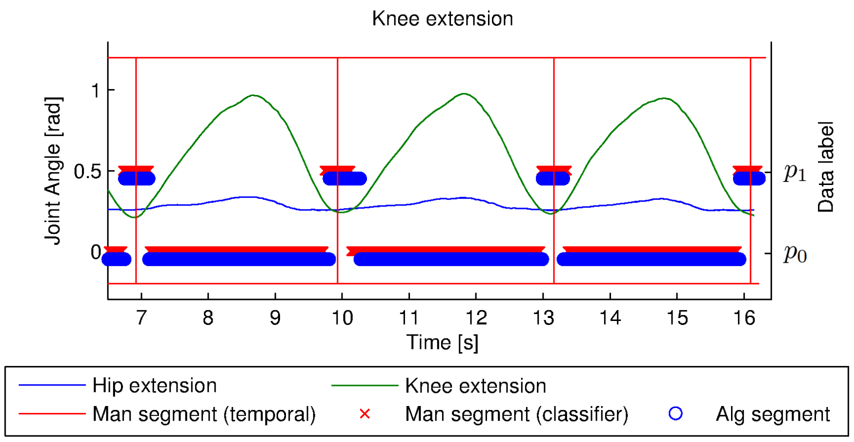


Figure 6.6: Segmented knee extension motion. Points around the $y = 0$ area are manual (red) and algorithmic (blue) points denoted as p_0, while points at around $y = 0.5$ are p_1 points.

Results for UW Individual and Group Overwhelmingly, the top performing classifiers utilize PCA. PCA selects between 30 to 40 dimensions to represent 80% of the total variability, confirming that the data is redundant and correlated, so that a lower dimensionality feature vector can be used.

The classifiers that reported the highest $Acc_{F_{Seg}}$ were the SVM, the ANN and the k-NN. These classifiers provided high accuracy in both the individualized template and the generalized template tests, suggesting that these classifiers are suitable for both intra- and inter-subject segmentation. The high processing costs of k-NN makes it unsuitable for online applications, however.

The tables show that aggregated techniques do not improve segmentation accuracy, as the top 10 classifiers include both non-aggregated and aggregated variants. Note that some aggregation is already performed by the sampling technique, which preferably selects p_0 points closer to the segmentation boundary for training. These results suggest that the training data sampling scheme is sufficient, since the sampling scheme emphasizes the p_0 points close to the p_1, which are likely to have a higher chance of misclassification. Table 6.10 shows the results from Tables 6.8 and 6.9, with the aggregation variables removed. This table shows that PCA no longer dominates the top performing classifiers. For k-NN, no performance difference was observed between k-NN and PCA k-NN, suggesting that k-NN is not sensitive to the input features. The SVM classifier exhibits some performance improvements when paired with PCA.

The misclassifications generally stem from the tendency of the classifiers to over-declare p_1 points, leading to a high FP score.

Individualized templates report higher accuracy than generalized templates, in both classifier and temporal F_1, underscoring that intra-subject variability is easier to handle than inter-subject variability.

The temporal accuracy is lower than the classifier accuracy, and is due to the temporal localization component. The simple conversion approach used generates algorithmic segments that are wider or narrower than the manual segments, thus leading to a poorer result, even though the algorithmic segments approximately overlap the manual segments. The improved performance between $t_{err} = 0.2s$ and $0.3s$ shows that many algorithmic segments sit just outside the boundary of the manual segments, and are flagged as FN instead of TP. In several instances, the manual segments are delayed due to the reaction speed in the expert performing the labelling, and the segments suggested by the algorithm may be more suitable to denote the actual location of the segment.

Table 6.6: Segmentation results using individualized templates, where the input vector consists of q and $\dot{q}$, $n_{exp} = 25$ and $n_{stack} = 15$. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), $Acc_{F_{Seg}}$ (F_S), and $Acc_{F_{Class}}$ (F_C) accuracies [%] for training and classifier testing results are reported. For brevity, only the top 10 results are reported.

	Classifier Parameter			**Training**				**Classifier Testing**			
	DR	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_C	F_P	F_R	F_S	F_C
1	PCA	SVM, radial	None	96	98	97	99	88	88	88	97
2	PCA	SVM, radial	Bagging, 5	95	97	96	98	88	88	88	97
3	PCA	SVM, radial	Bagging, 3	95	97	96	98	88	88	88	97
4	PCA	SVM, radial	Boosting, 5	95	98	96	98	88	89	88	97
5	PCA	SVM, radial	Boosting, 3	95	98	96	98	88	88	88	97
6	PCA	k-NN, 9	Bagging, 5	98	98	98	99	87	89	88	97
7	PCA	ANN, 20-20-20	Bagging, 5	99	100	100	100	85	79	82	95
8	PCA	ANN, 10-10-10	Bagging, 5	99	99	99	100	85	78	81	95
9	PCA	ANN, 10-10	Bagging, 5	99	99	99	100	84	78	81	95
10	PCA	ANN, 10-10	Bagging, 3	99	99	99	100	82	78	80	95

Table 6.7: Segmentation results using individualized templates, where the input vector consists of q and $\dot{q}$, $n_{exp} = 25$ and $n_{stack} = 15$. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), and $Acc_{F_{Seg}}$ (F_S) accuracies [%] for temporal testing results at different t_{err} are reported. For brevity, only the top 10 results are reported. The training time accounts for classifier training time, while the testing time accounts for classification time.

	Classifier parameter			**Temporal, t_{err} = 0.2s**			**Temporal, t_{err} = 0.3s**			**Time Taken [s]**	
	DR	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_P	F_R	F_S	**Training**	**Testing**
1	PCA	SVM, radial	None	98	86	92	99	92	95	183	114
2	PCA	SVM, radial	Bagging, 5	98	87	92	98	93	95	217	141
3	PCA	SVM, radial	Bagging, 3	97	86	91	99	93	95	290	143
4	PCA	SVM, radial	Boosting, 5	97	87	92	98	93	95	225	107
5	PCA	SVM, radial	Boosting, 3	97	87	92	98	93	95	149	62
6	PCA	k-NN, 9	Bagging, 5	97	86	91	99	93	96	8536	8741
7	PCA	ANN, 20-20-20	Bagging, 5	94	70	80	97	78	86	10257	94
8	PCA	ANN, 10-10-10	Bagging, 5	93	69	79	96	76	85	2031	87
9	PCA	ANN, 10-10	Bagging, 5	93	67	78	97	76	85	10965	114
10	PCA	ANN, 10-10	Bagging, 3	90	65	76	94	74	83	6675	91

Table 6.8: Segmentation results using generalized templates, where the input vector consists of q and $\dot{q}$, $n_{exp} = 25$ and $n_{stack} = 15$. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), $Acc_{F_{Seg}}$ (F_S), and $Acc_{F_{Class}}$ (F_C) accuracies [%] for training and classifier testing results are reported. For brevity, only the top 10 results are reported.

	Classifier Parameter			**Training**				**Classifier Testing**			
	DR	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_C	F_P	F_R	F_S	F_C
1	PCA	SVM, radial	Boosting, 3	91	97	94	97	75	90	82	95
2	PCA	ANN, 10-10	Bagging, 5	93	95	94	97	76	89	82	95
3	PCA	SVM, radial	None	92	97	94	97	76	89	82	95
4	PCA	SVM, radial	Bagging, 5	91	96	94	97	76	89	82	95
5	PCA	SVM, radial	Boosting, 5	91	97	94	97	74	90	81	95
6	PCA	SVM, radial	Bagging, 3	91	96	94	97	75	88	81	95
7	PCA	ANN, 10-10-10	Bagging, 5	94	96	95	97	76	88	81	95
8	PCA	ANN, 10-10	Bagging, 3	92	94	93	96	75	88	81	95
9	PCA	*k*-NN 9	Bagging, 5	93	95	94	97	73	90	81	94
10	PCA	ANN, 20-20-20	Bagging, 5	95	97	96	98	75	85	80	94

Table 6.9: Segmentation results using generalized templates, where the input vector consists of q and $\dot{q}$, $n_{exp} = 25$ and $n_{stack} = 15$. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), and $Acc_{F_{Seg}}$ (F_S) accuracies [%] for temporal testing results at different t_{err} are reported. For brevity, only the top 10 results are reported. The training time accounts for classifier training time, while the testing time accounts for classification time.

	Classifier parameter			**Temporal, t_{err} = 0.2s**			**Temporal, t_{err} = 0.3s**			**Time Taken [s]**	
	DR	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_P	F_R	F_S	**Training**	**Testing**
1	PCA	SVM, radial	Boosting, 3	83	74	78	89	84	87	764	2259
2	PCA	ANN, 10-10	Bagging, 5	86	75	80	92	85	88	1653	173
3	PCA	SVM, radial	None	85	75	80	90	85	87	98	406
4	PCA	SVM, radial	Bagging, 5	84	75	79	90	84	87	445	1559
5	PCA	SVM, radial	Boosting, 5	82	73	78	89	83	86	421	1521
6	PCA	SVM, radial	Bagging, 3	84	74	79	90	84	87	274	972
7	PCA	ANN, 10-10-10	Bagging, 5	86	74	80	92	84	88	1564	185
8	PCA	ANN, 10-10	Bagging, 3	85	74	79	91	85	87	958	149
9	PCA	*k*-NN 9	Bagging, 5	83	73	78	88	83	85	3175	23974
10	PCA	ANN, 20-20-20	Bagging, 5	84	70	77	91	82	86	5452	202

Table 6.10: Segmentation results using generalized templates, where the input vector consists of q and $\dot{q}$, $n_{exp} = 25$ and $n_{stack} = 15$. This table is similar to Table 6.8, but without any aggregation algorithms. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), and $Acc_{F_{Seg}}$ (F_S) [%] accuracies for training and testing results are reported.

	Classifier Parameter		**Training**				**Classifier Testing**			
	DR	**Classifier**	F_P	F_R	F_S	F_C	F_P	F_R	F_S	F_C
1	PCA	SVM, radial	92	97	94	97	76	89	82	95
2	None	kNN, 9	94	96	95	97	73	89	81	94
3	PCA	kNN, 9	94	96	95	97	73	89	81	94
4	PCA	ANN, 10-10-10	93	95	94	97	74	87	80	94
5	None	kNN, 3	97	98	98	99	74	87	80	94
6	PCA	kNN, 3	97	98	98	99	74	87	80	94
7	PCA	ANN, 10-10	92	94	93	96	72	88	79	94
8	PCA	ANN, 20-20-20	94	96	95	97	73	85	78	94
9	None	SVM, radial	84	95	89	94	68	92	78	93
10	None	ANN, 10-10	93	94	93	97	71	84	77	93

The proposed approach outperforms prior work [134], particularly for individualized templates. At $t_{err} = 0.2s$, for individualized templates, the proposed algorithm reports a top temporal Acc_{F_1} score of 92%, compared to a Acc_{F_1} score of 85% in [134]. For generalized templates, the proposed algorithm reports a top temporal Acc_{F_1} score of 80%, compared to a Acc_{F_1} score of 84% from the prior work [134]. The two approaches achieve similar performance at $t_{err} = 0.3s$, 93% and 95%, respectively. The proposed algorithm requires less parameter tuning and does not require velocity crossings [134] to occur at all segment edge points, making it easier to deploy, and apply to a wider number of exercises.

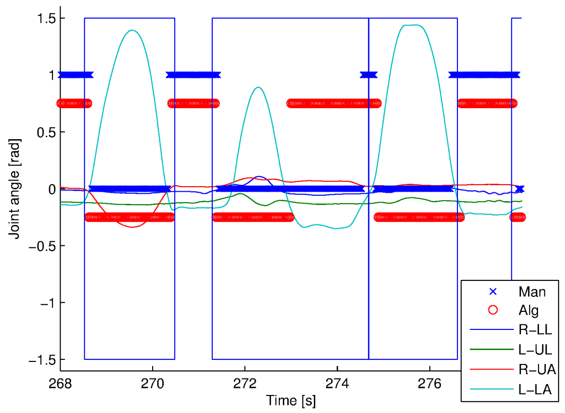
UT Dataset

Three sets of data were generated:

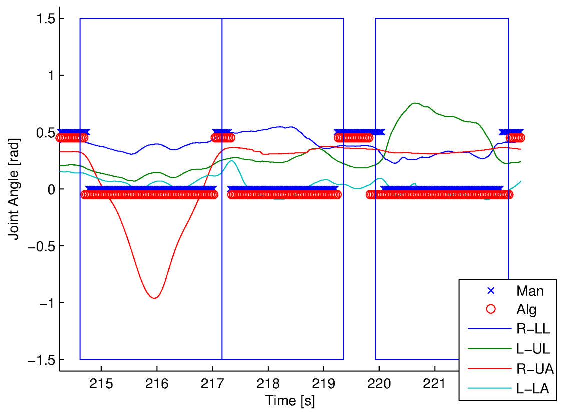
1. Training results, where the Acc_{F_1} score for the data used for the classifier training was assessed.

2. Classifier results, where the Acc_{F_1} score for the full observation data for primitives of the same type as the training data were assessed. The data examined here have not been used in the classifier training.

3. Generalization results, where the Acc_{F_1} score for the full observation data for primitives that have not been seen by the classifier training were assessed.



(a) Exercises depicted here are the left arm raise (286.5-270.5s), left arm punch (271.5-274.5s), and another left arm raise (274.5-276.5s). Manual segmentation markers, as can be seen in the left arm punch, can sometimes be inaccurate, leading to difficulties in training and classifying. Although the classifier was declared incorrect in this instance, the classifier was more accurate than the manual segments.



(b) Exercises depicted here are the right arm raise (214.5-217s), bowing (217-219.5s), and the squat (220-222s). The classifier must be able to handle a variety of situations. Here, the participant moves from right arm raise to the bowing motion immediately, but takes a break between bowing and squatting. The classifier was able to handle both situations.

Figure 6.7: Segmentation results of a participant performing full body exercises [114]. The blue rectangles denote the full-motion manual segmentation boundaries. The blue and red lines at $y = 0$ and $y = 0.5$ denote the classifier p_0 and p_1 for ground truth (blue x) and algorithmic (red circle) segmentation, respectively. The DOFs shown are right lower leg (R-LL), left upper leg (L-UL), right upper arm (R-UA) and left lower arm (L-LA).

Results for UT Full and Half For the classifier results, Tables 6.11, 6.12, 6.13, and 6.14 show that all the classifiers considered perform well, and that dimensionality reduction generally improves performance while decreasing training time. The high F_1 scores suggest that the classifier approach is capable of accurate segmentation under different configurations, including either full or half segments. These tables, sorted by the performance in $Acc_{F_{Seg}}$, show that the best classifiers for classification do not necessarily also generalize well. For brevity, only the results for classifiers ran at $n_{stack} = 15$ was shown, but classifiers ran at $n_{stack} = 0$ also show comparable performance.

PCA is included in the a majority of the top scoring configurations. A closer examination of the generated reduced-dimensionality subspace for the $n_{stack} = 0$ case shows that 80% of the variance is accounted for by the first 13 PCs, of the 40 total DOFs available from the dataset. The first PC accounts for 16% of the total variance, and has its top weights on the joint velocities of the right upper leg, right lower leg, left upper leg, left lower leg, and left upper arm. This shows that PCA selects the DOFs from all parts of the body. For $n_{stack} = 15$, the important body segments correspond to the right upper arm, right upper leg, left upper arm, and left upper leg, suggesting that the joint velocities have much higher variance than other DOFs.

Tables 6.11 and 6.13 show that aggregated techniques do not improve segmentation accuracy, as the top 10 performing configurations include both non-aggregated and different aggregated variants. It is important to note that some aggregation is already performed by the sampling technique, which preferably selects p_0 points closer to the segmentation boundary for training. These results suggest that the training data sampling scheme is sufficient, since the sampling scheme emphasizes the p_0 points close to the p_1, which are likely to have a higher chance of misclassification.

The runtime for the classifiers renders it impractical to utilize certain combinations. In general, the runtime for the k-NN is higher than the 17 minute dataset, thus k-NN cannot be used online, while the fastest NN and SVM configurations completed both the training and testing task in under 1 minute.

Several algorithms do not make the top 10 list in any of the configurations considered: the FDA, the QDA, and the RBF, caused by the specific features of this dataset. For the FDA, it can be shown that there will be at most $n - 1$ positive eigenvalues where n is the number of class labels. Because of this, FDA is not well suited for this two class problem since it will reduce the data to a one dimensional space and too much information is lost. Although the data appears to be separable, it does not seem to be linearly separable, thus the QDA did not perform well. For the RBF, the chosen parameters did not perform well.

We next examine if the model learned from a subset of motions can be extended to unseen motions. We hypothesize that segment edge points share similarities across motions (*i.e.*, zero velocity crossings or rest poses) so that the classifier can generalize to motions not included in the training set. In Figures 6.8 and 6.9, we provide the classification results for each motion type for the SVM and k-NN classifiers together with PCA dimensionality reduction, the top performing combinations from Tables 6.11 and 6.13, where the training primitives were randomly selected, and the F_1 score averaged over 100 trials. Half of the total available primitives were selected in both cases, so 9 primitives for the full-segments and 19 primitives for the half-segments. These results show that most motions, such as left arm raise to 180° (LA180), can still perform well even if the motion has not been seen in the training data. However, some movements, such as bowing (BOW) and squatting (SQ) do not generalize well if they are not included in the training set. This suggests some degree of classifier generalizability, which does not extend to all novel motions.

The misclassified data points generally fall into one of several categories: (1) ground truth labelling error from incorrect labels, or overlap in motion segments, leading to misalignment between the ground truth data and the algorithmic segment edge points, (2) classifier only being able to generalize into motions somewhat similar to the trained motions.

Table 6.11: Segmentation results, where the input vector consists of q and $\dot{q}$, $n_{exp} = 4$ and $n_{stack} = 15$, and the exercises examined consist of 751 half-motion repetitions of full body exercises [114]. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), $Acc_{F_{Seg}}$ (F_S), and $Acc_{F_{Class}}$ (F_C) accuracies [%] for training and classifier testing results are reported. For brevity, only the top 10 results are reported.

	Classifier Parameter			**Training**				**Classifier Testing**			
Rank	**DR**	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_C	F_P	F_R	F_S	F_C
1	PCA	SVM, polynomial	Bagging, 5	96	98	97	99	93	89	91	95
2	PCA	SVM, polynomial	Boosting, 5	97	99	98	99	92	89	91	95
3	PCA	SVM, polynomial	Boosting, 3	98	99	98	99	93	89	91	95
4	PCA	k-NN, 9	Boosting, 5	94	96	95	98	93	88	91	95
5	PCA	SVM, radial	Boosting, 3	96	97	96	98	94	87	91	95
6	None	k-NN, 9	Boosting, 5	93	96	95	98	93	88	91	95
7	None	k-NN, 9	Boosting, 3	94	96	95	98	93	88	91	95
8	PCA	SVM, radial	Boosting, 5	95	97	96	98	94	87	91	95
9	PCA	SVM, radial	None	97	97	97	99	95	87	91	95
10	PCA	k-NN, 9	Boosting, 3	94	96	95	98	94	88	91	95

Table 6.12: Segmentation results, where the input vector consists of q and $\dot{q}$, $n_{exp} = 4$ and $n_{stack} = 15$, and the exercises examined consist of 751 half-motion repetitions of full body exercises [114]. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), $Acc_{F_{Seg}}$ (F_S), and $Acc_{F_{Class}}$ (F_C) accuracies [%] for generalization testing results are reported. For brevity, only the top 10 results are reported. Training timing consists only of the classifier training time, while testing timing denotes the time taken to classify the whole dataset.

	Classifier Parameter			**Generalization**				**Time Taken [s]**	
Rank	**DR**	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_C	**Training**	**Testing**
1	PCA	SVM, polynomial	Bagging, 5	79	87	83	88	114	39
2	PCA	SVM, polynomial	Boosting, 5	77	88	82	88	112	43
3	PCA	SVM, polynomial	Boosting, 3	78	88	82	88	79	26
4	PCA	*k*-NN, 9	Boosting, 5	82	88	85	90	1547	3202
5	PCA	SVM, radial	Boosting, 3	85	86	86	91	185	98
6	None	*k*-NN, 9	Boosting, 5	83	88	85	90	12907	38764
7	None	*k*-NN, 9	Boosting, 3	83	88	85	90	20033	40819
8	PCA	SVM, radial	Boosting, 5	85	86	86	91	273	164
9	PCA	SVM, radial	None	86	85	86	91	89	52
10	PCA	*k*-NN, 9	Boosting, 3	82	87	85	90	1086	2145

Table 6.13: Segmentation results, where the input vector consists of q and $\dot{q}$, $n_{exp} = 4$ and $n_{stack} = 15$, and the exercises examined consist of 349 full-motion repetitions of full body exercises [114]. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), $Acc_{F_{Seg}}$ (F_S), and $Acc_{F_{Class}}$ (F_C) accuracies [%] for training and classifier testing results are reported. For brevity, only the top 10 results are reported.

	Classifier Parameter			**Training**				**Classifier Testing**			
Rank	**DR**	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_C	F_P	F_R	F_S	F_C
1	PCA	*k*-NN, 3	Boosting, 3	100	100	100	100	95	100	98	100
2	None	*k*-NN, 3	Boosting, 5	100	100	100	100	95	100	98	100
3	PCA	*k*-NN, 9	Boosting, 3	95	97	96	99	94	90	92	97
4	None	*k*-NN, 9	Boosting, 3	95	96	96	99	94	90	92	97
5	None	*k*-NN, 9	Boosting, 5	95	96	95	99	94	90	92	97
6	None	SVM, radial	None	88	95	91	97	94	90	92	97
7	None	SVM, radial	Boosting, 3	87	95	91	97	93	91	92	97
8	PCA	*k*-NN, 9	Boosting, 5	95	96	96	99	94	90	92	97
9	None	SVM, radial	Boosting, 5	86	96	91	97	93	91	92	97
10	None	*k*-NN, 3	Bagging, 5	98	97	98	99	95	89	92	97

Table 6.14: Segmentation results, where the input vector consists of q and $\dot{q}$, $n_{exp} = 4$ and $n_{stack} = 15$, and the exercises examined consist of 349 full-motion repetitions of full body exercises [114]. The $Acc_{precision}$ (F_P), Acc_{recall} (F_R), $Acc_{F_{Seg}}$ (F_S), and $Acc_{F_{Class}}$ (F_C) accuracies [%] for generalization testing results are reported. For brevity, only the top 10 results are reported. Training timing consists only of the classifier training time, while testing timing denotes the time taken to classify the whole dataset.

	Classifier Parameter			**Generalization**				**Time Taken [s]**	
Rank	**DR**	**Classifier**	**Aggregator**	F_P	F_R	F_S	F_C	**Training**	**Testing**
1	PCA	k-NN, 3	Boosting, 3	75	100	86	97	627	546
2	None	k-NN, 3	Boosting, 5	75	100	85	97	13866	16631
3	PCA	k-NN, 9	Boosting, 3	73	93	82	91	241	566
4	None	k-NN, 9	Boosting, 3	74	93	83	92	9057	20969
5	None	k-NN, 9	Boosting, 5	74	93	82	92	16462	38956
6	None	SVM, radial	None	78	91	84	93	521	489
7	None	SVM, radial	Boosting, 3	78	92	84	93	1652	1271
8	PCA	k-NN, 9	Boosting, 5	74	93	82	92	393	924
9	None	SVM, radial	Boosting, 5	77	92	84	93	2137	1653
10	None	k-NN, 3	Bagging, 5	75	93	83	92	6859	15072

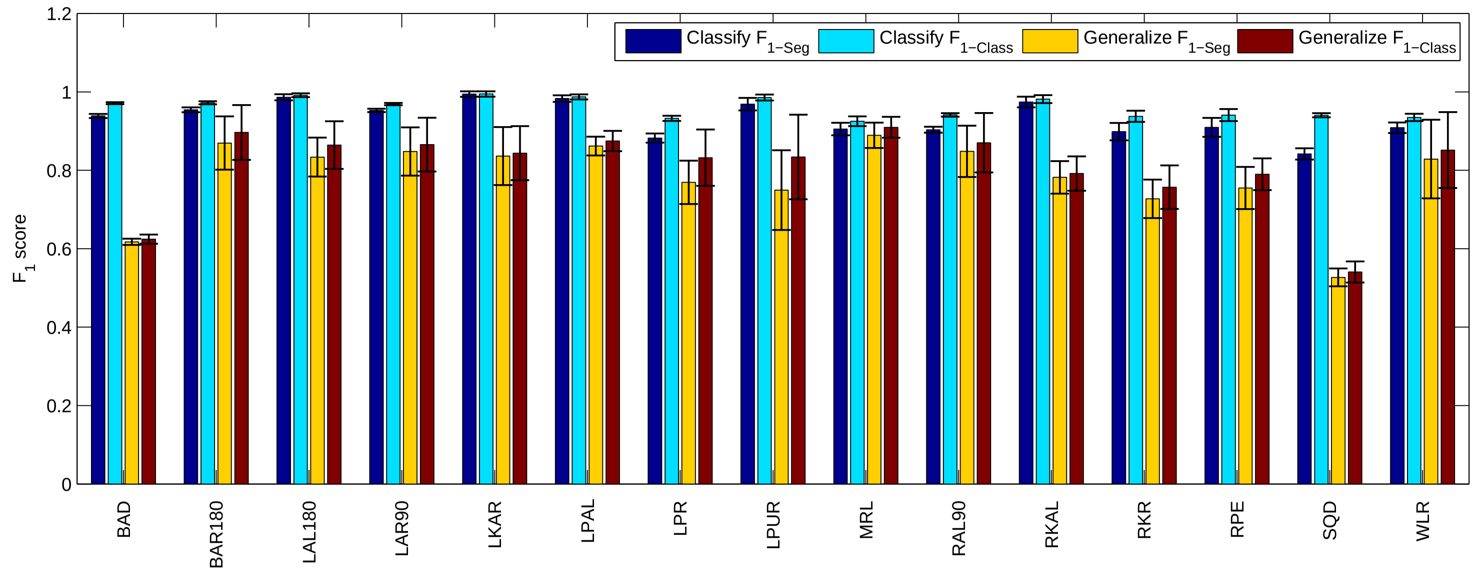


Figure 6.8: Segmentation results for training and testing on the top performing classifier from Table 6.11, the bagged PCA SVM classifier. 23 half-segment primitives are randomly selected for training, and tested on the other 22 primitives. For brevity, a representative subset of all the primitives are displayed here. The F_1 scores were generated by randomizing the primitives chosen for the training set and averaging over 100 iterations. The exercises shown here are bowing down (BAD), both arm raise to 180° (BAR180), left arm lower from 180° (LAL180), left arm raise to 90° (LAR90), left kick arm raise (LKAR), left punch arm lower (LPAL), left punch retract (LPR), left punch upper retract (LPUR), marching right leg lowering (MRL), right arm lower from 90° (RAL90), right kick arm lower (RKAL), right kick raise (RKR), right punch extending (RPE), squat down (SQD) and walking left leg raise (WLR).

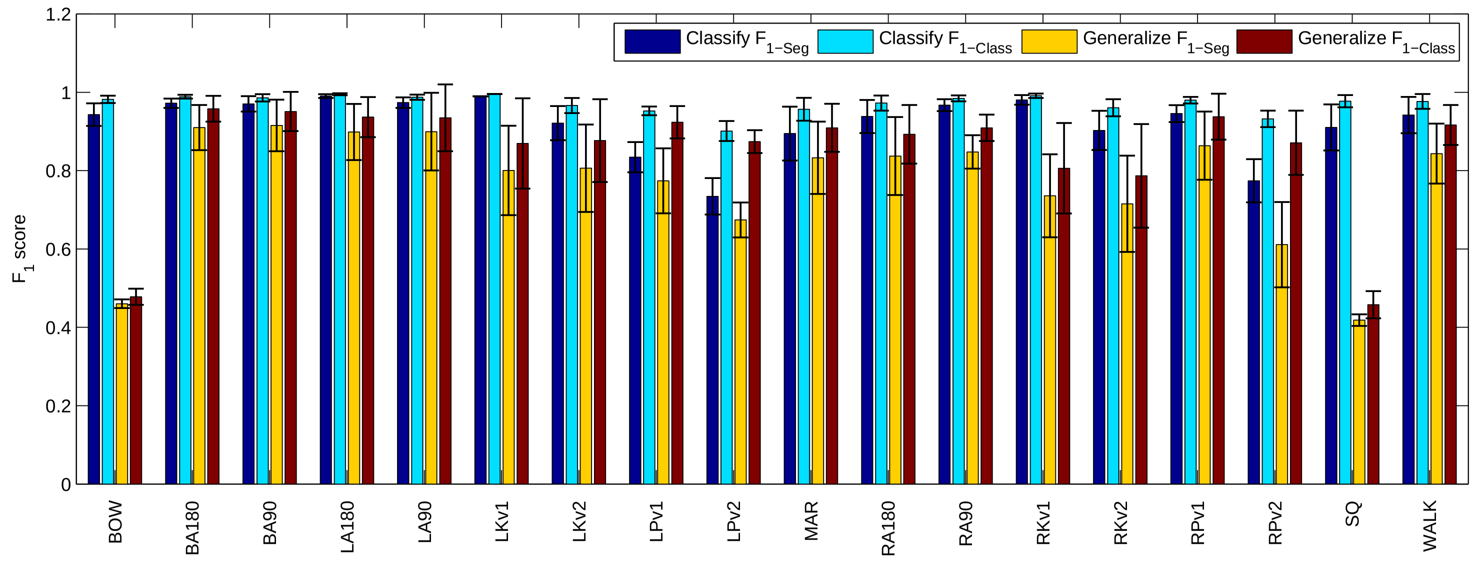


Figure 6.9: Segmentation results for training and testing on the top performing classifier from Table 6.13, the boosted PCA k-NN classifier. 9 full-segment primitives are used for training, and tested on 19 primitives. The F_1 scores were generated by randomizing the primitives chosen for the training set and averaging over 100 iterations. The exercises consist of bowing (BOW), both arms raise to 180° (BA180), both arms raise to 90° (BA90), left arm raise to 180° (LA180), left arm raise to 90° (LA90), left kick with arm raises (LKv1), left kick only (LKv2), left punch after arm guard (LPv1), left punch only (LPv2), marching (MAR), right arm raise to 180° (RA180), right arm to 90° (RA90), right kick with arm raises (RKv1), right kick only (RKv2), right punch after arm guard (RPv1), right punch only (RPv2), squat (SQ) and walking (WALK).

The proposed work outperforms more computationally complex time series methods [114], with a $Acc_{F_{Seg}}$ of 91%, compared to 75% achieved in [114], showing that classification techniques are a promising approach for accurate online time-series segmentation.

6.4.2 Normalization

Four sets of data were generated:

1. Results for known exercises, where the Acc_{bal} score for primitives that were included in the training set for UW, SJHCG, and TRI datasets were assessed.
2. Results for novel exercises, where the Acc_{bal} score for primitives that were not included in the training set for UW, SJHCG, and TRI datasets were assessed.
3. PCA mapping analysis, where the impact of normalization and n_{stack} on the resultant PCA projection was examined to obtain insight on these variables.
4. Comparison to existing work, where the proposed algorithm was compared to ZVC and HMM-based approaches.

Results for Known Exercises A 4-fold cross-validation was utilized to test inter-participant generalization with exercises that appear in the training set (Table 6.15). The same classifiers, trained only on healthy data, were used to test both the healthy and rehabilitation datasets, using different pre-processing methods.

For healthy participants, removing the offset significantly improves the balanced accuracy, moving from 74% ± 26% to 91% ± 6%, an increase of 15% in accuracy and a drop by a factor of 4 in the standard deviation. The offset removal allows motions that start at different postures to appear more similar, thus improving accuracy. Normalizing without using the scaling factor provided the best performance. The results also show that poor normalization has a heavier impact on the precision scores when compared to recall scores, suggesting a larger number of false segment edge point declarations.

The preprocessing techniques also resulted in a similar outcome for the SJHCG dataset, where the offset normalization significantly improves the segmentation accuracy. Applying the rectification alone led to performance degradation, but when combined with the other normalization methods, up to 17% improvement can be observed.

The preprocessing techniques have the strongest impact on the TRI dataset. Rectifying the data improved performance, but not as significantly as magnitude scaling the data.

Magnitude scaling allows the patient datasets to look more similar to the training set, allowing for improvements of over 17% in accuracy for some configurations. Combining all three preprocessing steps improves performance on the TRI dataset to 85%. Offset removal alone improves accuracy from 60% to 65%. Without normalization, precision scores are very poor and achieve below 50%, while all 3 normalization methods increase precision scores by 40% and recall scores by 35%.

Results for Novel Exercises The 4-fold training cross-validation was also applied to novel exercises, where the test exercises were not included in the training set to test inter-exercise generalization (Table 6.15). This test is important for clinical application because if the algorithm can successfully generalize to exercises that were not explicitly included in the training dataset, it reduces the amount of training data that must be provided. Table 6.15 shows that, without pre-processing, the novel exercises do not perform as well as the known ones.

For healthy participants, the offset value removal is the most influential factor, improving the accuracy by 13%. Scaling and rectification alone did not seem to impact the accuracy significantly. However, with all three normalizations active, the accuracy is improved by 18%. Similar to known exercises, the lack of proper normalization impacts the precision scores more heavily than the recall scores, suggesting false positive segment edge declarations.

For patient data, without normalization, the segmentation performance declines significantly, but with offset value removal, scaling and rectification, the TRI novel exercise approaches performance with the known exercises, at 85% to 83% for the known and novel datasets, respectively. The most influential normalization is the offset and the scaling, which allowed the TRI data to appear more similar to the healthy training data. Similar results can be observed for the SJHCG novel exercises, where the reported accuracy was 89% and 88% for the known and novel motions, respectively. TRI precision score is very low, 22%, when the normalization methods are not sufficient, but improves by 51% after offset removal and magnitude scaling normalization.

PCA Mapping The PCA mapping of the data shows that the top PCs are composed of the knee sagittal joint angle, the knee sagittal joint velocity, the hip sagittal joint angle, the hip sagittal joint velocity, the hip abduction joint angle, and finally the hip abduction joint velocity, which are the main features that change over time over all the datasets. These PCs, under different configurations, can be seen in Figure 6.10.

Table 6.15: 4-fold cross-validated classifier $Acc_{precision}$, Acc_{recall}, and Acc_{bal} performance for known and novel exercises (Table 6.4). Three types of normalizations were examined: offset value removal (off), magnitude scaling (sca), and rectification of the joint angle (abs).

$Acc_{precision}$			**UW**		**SJHCG**		**TRI**	
Off	**Sca**	**Abs**	**Known**	**Novel**	**Known**	**Novel**	**Known**	**Novel**
0	0	0	57 ± 42	41 ± 31	69 ± 30	67 ± 20	43 ± 36	26 ± 31
0	0	1	58 ± 42	34 ± 26	58 ± 27	38 ± 20	42 ± 34	22 ± 24
0	1	0	63 ± 39	48 ± 33	71 ± 28	38 ± 19	47 ± 37	35 ± 36
0	1	1	64 ± 39	50 ± 30	68 ± 26	36 ± 17	44 ± 37	29 ± 33
1	0	0	84 ± 16	71 ± 24	76 ± 20	61 ± 16	51 ± 22	48 ± 20
1	0	1	84 ± 16	73 ± 22	73 ± 22	**63 ± 16**	52 ± 21	46 ± 18
1	1	0	78 ± 19	**78 ± 18**	82 ± 19	58 ± 15	69 ± 23	**73 ± 22**
1	1	1	**86 ± 12**	77 ± 17	**85 ± 15**	62 ± 14	**73 ± 20**	71 ± 21

Acc_{recall}			**UW**		**SJHCG**		**TRI**	
Off	**Sca**	**Abs**	**Known**	**Novel**	**Known**	**Novel**	**Known**	**Novel**
0	0	0	62 ± 44	66 ± 41	71 ± 31	80 ± 26	56 ± 46	47 ± 48
0	0	1	63 ± 44	67 ± 41	87 ± 23	95 ± 11	59 ± 46	50 ± 49
0	1	0	69 ± 42	53 ± 38	74 ± 32	93 ± 11	47 ± 45	39 ± 45
0	1	1	70 ± 41	52 ± 35	80 ± 27	93 ± 09	48 ± 46	38 ± 45
1	0	0	91 ± 12	88 ± 16	91 ± 14	**95 ± 09**	98 ± 09	99 ± 08
1	0	1	**92 ± 10**	**89 ± 15**	**91 ± 12**	95 ± 10	**99 ± 07**	**99 ± 05**
1	1	0	90 ± 15	85 ± 15	84 ± 24	94 ± 10	90 ± 20	88 ± 21
1	1	1	90 ± 16	85 ± 15	86 ± 22	94 ± 11	91 ± 16	90 ± 18

Acc_{bal}			**UW**		**SJHCG**		**TRI**	
Off	**Sca**	**Abs**	**Known**	**Novel**	**Known**	**Novel**	**Known**	**Novel**
0	0	0	74 ± 26	69 ± 16	76 ± 16	84 ± 12	61 ± 22	51 ± 21
0	0	1	75 ± 26	64 ± 16	72 ± 17	70 ± 12	58 ± 21	46 ± 16
0	1	0	77 ± 25	70 ± 15	78 ± 17	71 ± 09	62 ± 21	57 ± 11
0	1	1	78 ± 25	67 ± 14	78 ± 16	69 ± 09	61 ± 20	54 ± 08
1	0	0	91 ± 06	84 ± 12	86 ± 12	88 ± 06	65 ± 16	60 ± 15
1	0	1	**92 ± 05**	86 ± 10	83 ± 13	**89 ± 06**	67 ± 16	58 ± 14
1	1	0	89 ± 10	**87 ± 08**	87 ± 12	86 ± 07	81 ± 15	83 ± 15
1	1	1	91 ± 07	**87 ± 08**	**89 ± 11**	88 ± 05	**85 ± 12**	**83 ± 14**

For $n_{stack} = 0$, the first PC contains the knee sagittal and hip sagittal joint velocity. The second PC contains the knee sagittal and hip sagittal joint angles. These two PCs effectively form a phase plot of these two different joints, emphasizing the cyclic characteristics of the exercise movements.

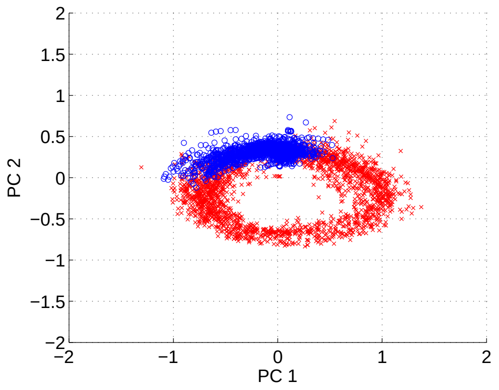
For $n_{stack} = 15$, PCA generates a similar looking projection, even though the input feature has been embedded with temporally-offset data. This indicates that PCA is constructing linear combinations of the different joint angles and velocities, generating a space analogous to the phase plot, but automatically incorporating multiple joints and short-term temporal information.

Table 6.16 shows the classification accuracy of different n_{stack} values. Smaller n_{stack} values, corresponding to less temporal information embedded at each timestep, led to a decrease in accuracy. The accuracy difference between $n_{stack} = 15$ used in Sections 6.4.2 and 6.4.2 and larger n_{stack} values does not vary significantly or degraded slightly in accuracy, due to the wider windows covering components of the trajectory that are both moving and not moving. These results suggest that for the dataset examined, the window of 0.6 seconds provided by $n_{stack} = 15$ is a suitable choice.

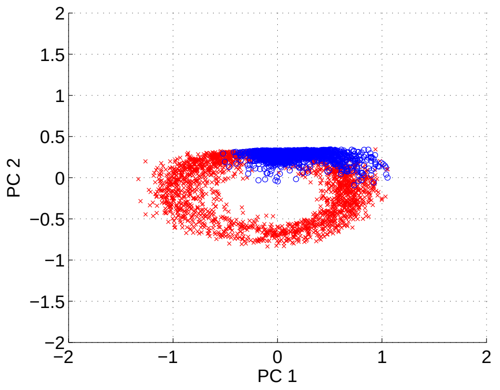
Comparison to Existing Work The velocity feature HMM (vfHMM) [134] was compared to the proposed algorithm. It uses velocity crossings and peaks to identify potential segments, then selects the best one based on HMM identification. In addition, a ZVC algorithm [66] was also implemented, where velocity crossings indicated direction change and thus segment points.

Joint angle data that had discontinuities, integration drift, or noisy recovery were removed, as they caused the vfHMM training algorithm [134] to diverge. For the ZVC algorithm [66], every second identified ZVC was used to create the segments. The single point segments declared by the comparison algorithms were converted into p_0 and p_1 points such that the balanced accuracy metric can be used for all algorithms. For this section, the training and testing data of all three algorithms were normalized by offset removal and magnitude normalization.

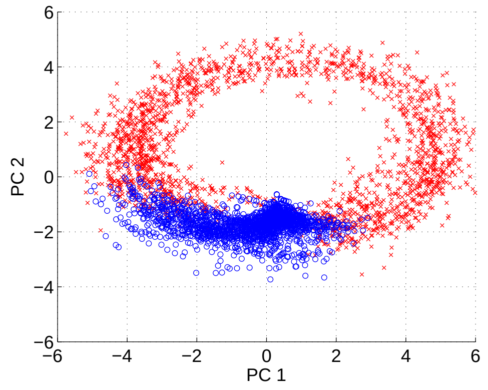
Table 6.17 indicates that the proposed algorithm outperforms both of the comparison algorithms, especially in the recall scores. Both the vfHMM and ZVC methods rely on velocity crossings to indicate crossing points, rather than learning the segment point locations from expert-labelled data. Human observers may define segments to lead or lag the velocity crossing. These variations in the manual segments highlight the importance of segment definition flexibility, where the proposed algorithm outperforms the two compar-



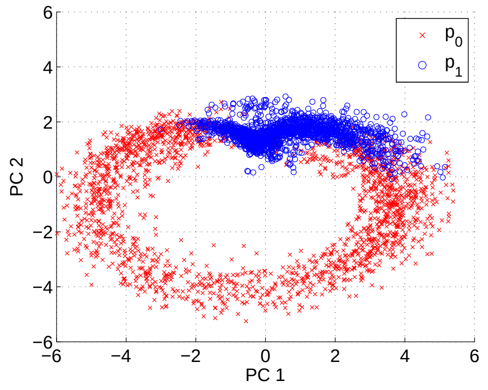
(a) Unrectified data, $n_{stack} = 0$.



(b) Rectified data, $n_{stack} = 0$.



(c) Unrectified data, $n_{stack} = 15$.



(d) Rectified data, $n_{stack} = 15$.

Figure 6.10: 4 plots of the training data, projected onto a 2D plane via PCA using the projection matrix generated by the offset value removal and scaled training data. These plots show the different PCs at different n_{stack} values and rectification state. Note that the scale between the left two figures and the right two figures are different. In all configurations, the first 4 PCs heavily weigh the knee sagittal and hip sagittal joint angles and velocities, due to their high variance in the trajectories examined. Although not shown here, the subsequent PCs weigh the hip abduction joint angle and velocity more heavily.

Table 6.16: Classification $Acc_{precision}$, Acc_{recall}, and Acc_{bal} accuracy of testing data using classifiers that have been normalized by using offset value removal, magnitude scaling and rectification. The baseline is bolded and denoted as $n_{stack} = 15$. Different n_{stack} parameters are examined. The results are separated into known and novel exercises, as denoted by Table 6.4.

$Acc_{precision}$	**UW**		**SJHCG**		**TRI**	
n_{stack}	**Known**	**Novel**	**Known**	**Novel**	**Known**	**Novel**
0	74 ± 16	68 ± 17	77 ± 18	57 ± 13	71 ± 19	71 ± 21
5	81 ± 15	72 ± 17	81 ± 16	59 ± 14	72 ± 20	71 ± 21
10	84 ± 13	76 ± 17	82 ± 16	59 ± 13	72 ± 20	71 ± 21
15	**86 ± 12**	**77 ± 17**	**85 ± 15**	**62 ± 14**	**73 ± 20**	**71 ± 21**
30	84 ± 15	78 ± 19	81 ± 19	60 ± 14	73 ± 20	73 ± 21
Acc_{recall}	**UW**		**SJHCG**		**TRI**	
n_{stack}	**Known**	**Novel**	**Known**	**Novel**	**Known**	**Novel**
0	91 ± 16	91 ± 13	86 ± 23	93 ± 11	86 ± 24	85 ± 24
5	90 ± 15	88 ± 14	84 ± 24	92 ± 11	89 ± 18	88 ± 20
10	90 ± 14	85 ± 15	85 ± 23	94 ± 10	91 ± 17	89 ± 19
15	**90 ± 16**	**85 ± 15**	**86 ± 22**	**94 ± 11**	**91 ± 16**	**90 ± 18**
30	86 ± 23	86 ± 15	83 ± 24	90 ± 16	91 ± 17	89 ± 18
Acc_{bal}	**UW**		**SJHCG**		**TRI**	
n_{stack}	**Known**	**Novel**	**Known**	**Novel**	**Known**	**Novel**
0	86 ± 08	86 ± 8	86 ± 11	86 ± 05	83 ± 13	82 ± 16
5	89 ± 07	87 ± 8	87 ± 11	87 ± 05	84 ± 12	82 ± 15
10	90 ± 07	87 ± 8	88 ± 11	87 ± 05	85 ± 12	83 ± 14
15	**91 ± 07**	**87 ± 08**	**89 ± 11**	**88 ± 05**	**85 ± 12**	**83 ± 14**
30	89 ± 11	87 ± 8	86 ± 13	86 ± 08	85 ± 12	83 ± 14

ison works. In addition, the proposed approach can generalize to unseen exercises, while vfHMM [134] requires a template for each motion.

Table 6.17: Classification $Acc_{precision}$, Acc_{recall}, and Acc_{bal} accuracy of the proposed algorithm, compared against two other methods, the velocity feature HMM (vfHMM) and the zero-velocity crossing (ZVC), using known exercises. The proposed method is bolded.

$Acc_{precision}$	**UW**	**SJHCG**	**TRI**
ZVC [66]	75 ± 21	70 ± 29	59 ± 30
vfHMM [134]	72 ± 29	67 ± 33	59 ± 35
PCA SVM	**86 ± 12**	**85 ± 15**	**73 ± 20**
Acc_{recall}	**UW**	**SJHCG**	**TRI**
ZVC [66]	48 ± 19	49 ± 25	50 ± 28
vfHMM [134]	45 ± 23	47 ± 26	59 ± 30
PCA SVM	**90 ± 16**	**86 ± 22**	**92 ± 16**
Acc_{bal}	**UW**	**SJHCG**	**TRI**
ZVC [66]	71 ± 11	70 ± 15	67 ± 18
vfHMM [134]	69 ± 13	68 ± 16	70 ± 18
PCA SVM	**91 ± 7**	**89 ± 11**	**85 ± 12**

With suitable pre-processing, the classifier can generalize both to patient data and to novel exercises, based on only a healthy participant training dataset.

6.4.3 Features

UWT Dataset

Four sets of data were generated:

1. Results for normalization, where the Acc_{bal} scores for different features under different EMG-based normalization were examined.

2. Results for classification, where the Acc_{bal} scores of different classifiers were examined.

3. Results for LOMO, where one motion type was left out of the training set and cross-validation was conducted.

4. Results for LOPO, where one participant was left out of the training set and cross-validation was conducted.

Results for UWT Normalization In the first experiment involving the UWT Dataset (Table 6.1, UWT IG Normalize), two factors were examined: (1) EMG features, and (2) normalization type: The EMG features specified in Section 5.4 were considered. The calculations were performed over a sliding window of length 20 samples (0.067 sec) with a window overlap of 10 samples (0.033 sec).

The influence of these factors on segmentation accuracy is reported in Table 6.18. These results were generated by dividing the IG dataset into two folds of 5 participants each and using each fold to train a separate classifier. Both classifiers were tested against the CR dataset, and the averaged accuracy between the two classifiers was calculated. The table reports the top 5 performing features according to segmentation Acc_{bal}. The top performing features while using the threshold approach were the Teager energy (TE), waveform length (WFL), mean absolute value (MAV), root mean square (RMS) + PIP (RP), and raw EMG + RMS + PIP (RRP). Other features examined generally require larger windows of the EMG data to be available, and are not suitable for the temporal resolution required for segmentation purposes. These features are denoted in Appendix B.

Table 6.18 shows that channel-wise normalization did not perform as well as the other normalizations, while the other three normalization methods showed comparable results. This is likely due to channel-wise normalization de-emphasizing individual channel differences in the EMG data between the different gestures. Similar results between normalized and non-normalized data could indicate that these features are not sensitive to inter-participant differences in the signal, leading to comparable results between the no normalization case and the normalization cases. Although participant-based normalization did not perform the best in Table 6.18, its performance was comparable to the other normalization types, it requires the least amount of input from the participant while improving the ability to handle large inter-personal variability, and will be used as the normalization scheme in the rest of the EMG-based experiments.

Results for UWT Classifier In the second set of experiments with the UWT dataset (Table 6.1, UWT Classifier), the best features from the UWT Normalization experiments (TE, WFL, MAV, RP, and RPP) were used to evaluate the impact of classifier choice.

The results are summarized in Table 6.19. An illustration of the algorithm segmenting the EMG features can be found in Figure 6.11. Similar to Table 6.18, Table 6.19 was

Table 6.18: Acc_{bal} scores [%], reported for varying features, after channel, motion, participant or no normalization. The features reported are the top performing features, and are as follows: the Teager energy (TE), waveform-length (WFL), mean absolute value (MAV), the combined features of RMS EMG + inner-product (RP), and the combined features of raw EMG + RMS EMG + inner-product (RRP). Highest Acc_{bal} in each normalization type is bolded. The threshold classifier was used to generate this table, with $n_{exp} = 5$.

	Normalization			
Features	**Channel**	**Motion**	**Participant**	**None**
TE	72 ± 04	74 ± 01	74 ± 01	74 ± 01
WFL	72 ± 07	76 ± 03	75 ± 02	76 ± 03
MAV	65 ± 12	71 ± 08	69 ± 09	69 ± 09
RP	**73 ± 05**	**79 ± 00**	**78 ± 01**	**78 ± 00**
RRP	72 ± 01	72 ± 01	72 ± 01	78 ± 00

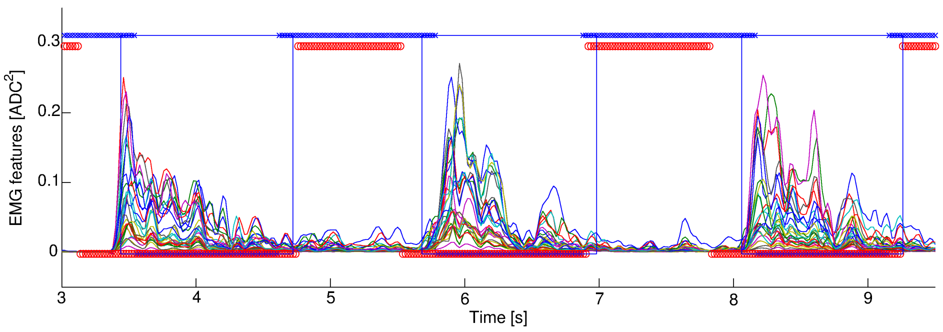


Figure 6.11: Segmentation results of a participant performing a sequence of random hand gestures, using EMG pairwise inner-product features. The blue rectangles denote the full-motion manual segmentation boundaries. The blue and red lines at the top and bottom denote the classifier p_0 and p_1 for ground truth (blue x) and algorithmic (red circle) segmentation, respectively.

generated by dividing the IG set into two subsets, training a classifier on each subset and averaging the accuracy obtained by each classifier on the testing set. Table 6.19 shows that ANN and QDA both performed comparatively, showing that both simple and complex classifiers can perform well. In Table 6.18, the RP was shown to be the best performing feature for participant normalized data, but was outperformed by other features in Table 6.19.

Table 6.19: Acc_{bal} scores [%], reported for varying classifiers and features. The top scoring classifier from each classifier type is reported. The best performing features are mean absolute value (MAV), and the RMS + inner-product (RP).

Rank	**Classifier**	**Feature**	**Accuracy**
1	ANN, 10^2	MAV	83 ± 6
2	QDA	MAV	81 ± 7
3	SVM, linear	WFL	81 ± 5
4	LDA	RP	80 ± 6
5	k-NN, 3	MAV	78 ± 7
6	Threshold	RP	78 ± 9

Results for UWT LOMO Table 6.20 reports the inter-gesture generalization results, where the gesture under inspection was left out of the IG dataset using leave-one-motion-out (LOMO) cross validation. The training data was generated from the same participant as the observation data, and the Acc_{bal} scores were averaged across all participants. The best performing classifier from Table 6.19, the ANN, was used for this test.

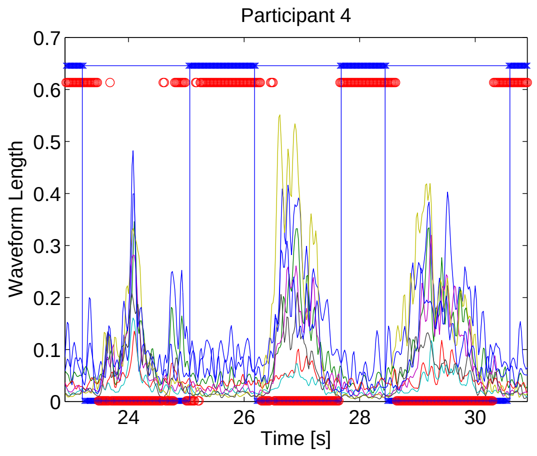
From Table 6.20, it can be observed that all features tend to perform similarly, with the exception of RRP. Of all the assessed features, RRP contains 44 (8 from raw EMG, 8 from RMS EMG and 28 from the PIP) elements, which is the largest number of elements. The PCA reduction of RRP features might have resulted in overfitting to the training data.

Table 6.20 also provides insight into the generalizability of the proposed approach to unseen motions. Table 6.20 suggests that the classifier is able to generalize to most new motions. This is a major potential advantage of the proposed approach, eliminating the need for having a fixed set of *a priori* specified motions. However, some motions, such as the fist motion, do not generalize well, and would need to be explicitly included in the training set to improve the segmentation performance.

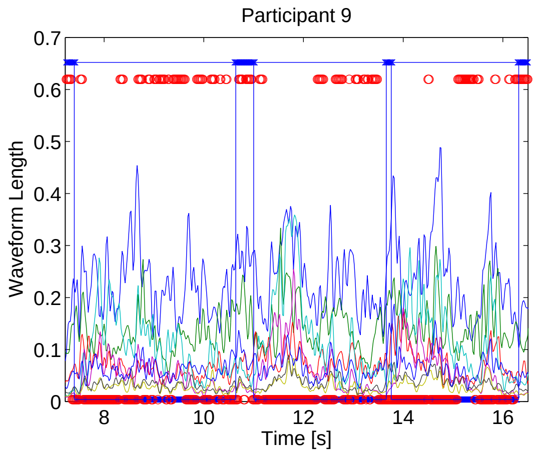
Table 6.20: Acc_{bal} scores [%], reported for LOMO cross-validation. The gestures tested were finger spread (FISP), fist (FIST), gun (GUNM), paddle in (PDIN), paddle out (PDOU), pointing with index finger (POIN), pointing with index and middle (POIM), finger snapping (SNAP) and thumb-pinky touch (THPK). The features listed are the Teager energy (TE), waveform-length (WFL), mean absolute value (MAV), the RMS + inner-product (RP), and the raw EMG + RMS + inner-product (RRP). The ANN classifier was used. The best performing feature for each gesture is bolded.

Gesture	**FISP**	**FIST**	**GUNM**	**PDIN**	**PDOU**	**POIN**	**POIM**	**SNAP**	**THPK**
TE	85 ± 07	77 ± 12	**85 ± 07**	82 ± 11	**87 ± 07**	82 ± 07	82 ± 13	78 ± 05	82 ± 11
WFL	86 ± 06	**76 ± 12**	85 ± 08	**83** ± 11	85 ± 11	**85** ± 08	**86 ± 08**	81 ± 06	**83 ± 10**
MAV	**87** ± 06	76 ± 13	85 ± 08	82 ± 13	83 ± 13	84 ± 10	85 ± 10	**83 ± 06**	82 ± 09
RP	82 ± 09	75 ± 15	85 ± 08	76 ± 12	84 ± 12	85 ± 09	85 ± 09	82 ± 09	81 ± 10
RRP	77 ± 07	69 ± 9	75 ± 08	71 ± 14	78 ± 11	71 ± 09	80 ± 09	70 ± 12	75 ± 09

Results for UWT LOPO To evaluate the generalizability of the learned classifier to participants unseen during training, Table 6.21 was generated by leaving one-participant-out (LOPO) of the training IG dataset, and testing against the data of that individual from the CR dataset. This cross-validation shows that, for most participants, excellent generalization performance is achieved, with classification results comparable to the case when their data is included in the training set. However, for some participants, *e.g.*, participant 9, lower accuracy scores are observed when they are left out, suggesting that their EMG data differs from the other participants. See Figure 6.12 for a comparison between participant 4 and 9. An examination into the movements of participant 9 revealed higher variance in both movement and resting postures, as well as significant differences in the EMG data between these two participants and the other participants, thus causing a degradation in the segmentation performance. Even when participant 9 is included in the training set, the classifier accuracy does not increase, which suggests differences in the way the gestures were performed between the IG and the CR data for this participant.



(a) Participant 4, performing pointing with index (23 - 25 sec), pointing with index and middle (26 - 27.5 sec) and paddle out (28.5 - 30.5 sec). This participant obtained a high LOPO score.



(b) Participant 9, performing 3 instances of pointing with index and middle (7 - 10.5 sec, 11 - 13.5 sec, 14 - 16 sec). Note the dissimilarity between the two participant's motions.

Figure 6.12: Segmentation results of two participants performing a random sequence of hand gestures, with the WFL feature. The blue rectangles denote the manual segmentation boundaries. The blue and red lines at the top and bottom denote the classifier p_0 and p_1 for ground truth (blue x) and algorithmic (red o) segmentation, respectively. The coloured waveforms correspond to different EMG channels and feature elements.

Table 6.21: Acc_{bal} scores [%], reported for LOPO cross-validation. The features reported are the Teager energy (TE), waveform-length (WFL), mean absolute value (MAV), the RMS + inner-product (RP), and the raw EMG + RMS + inner-product (RRP). The ANN classifier was used. The best performing result for each gesture is bolded.

Participant	**4**	**5**	**6**	**7**	**8**	**9**	**10**
TE	88 ± 0	87 ± 0	86 ± 1	82 ± 1	83 ± 2	71 ± 4	76 ± 3
WFL	**89 ± 1**	91 ± 1	84 ± 1	82 ± 2	83 ± 1	67 ± 4	**81 ± 1**
MAV	89 ± 1	**91 ± 1**	**86 ± 1**	**83 ± 1**	**87 ± 1**	**72 ± 6**	80 ± 2
RP	88 ± 1	90 ± 0	82 ± 1	82 ± 2	85 ± 2	63 ± 5	78 ± 2
RRP	82 ± 1	83 ± 0	81 ± 1	75 ± 2	71 ± 1	59 ± 3	72 ± 3

From Tables 6.19, 6.20 and 6.21, it can be observed that different features achieved the highest balanced accuracy score, suggesting that different combinations of features and classifiers can perform well in different situations. The best feature across all three tables is the MAV, achieving 82.9% accuracy with the ANN classifier, 86.5% with finger spread in the LOMO test, as well as achieving 91.4% with participant 5 in the LOPO test. This could be due to the fact that the MAV removes the high frequency variations and abrupt changes in the signals, whereas features such as RMS may amplify these fluctuations

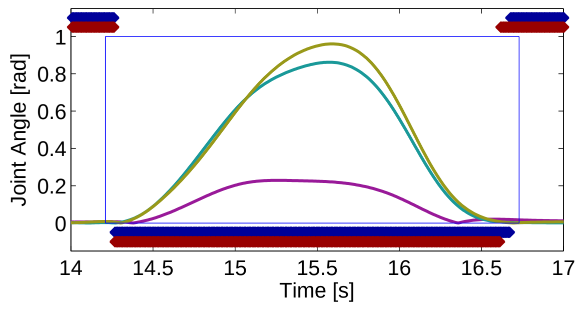
URFI Dataset

One set of data was generated:

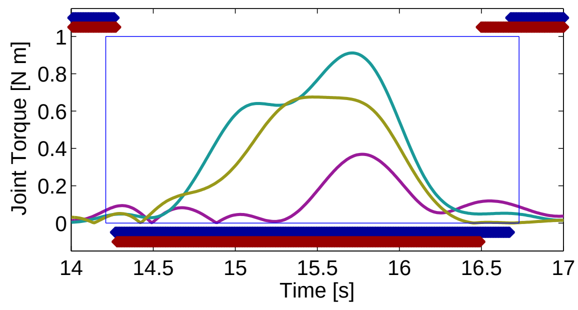
1. Results for different features, where the Acc_{bal} scores for features generated from different sensor modalities were examined.

Figure 6.13 illustrates the segmentation results using different features. As can be observed from the figure, different features generate differing signal profiles with respect to the manually defined segments. Data obtained from the kinematic measurements (joint position and angle) is much less noisy compared to data obtained from the force plates (GRF and COP), with joint torque lying in between. Kinematic data and GRF data align temporally with the manually defined segments, while joint torque activations trail the start of a manually defined segment, and precede the end of a manually defined segment. This is due to the fact that the manual segments were generated from the joint angles. As seen in Figures 6.13a-6.13b, while the joint angles of the participant are still moving to complete the squat, the torques approach zero, likely due to the use of passive dynamics to complete the motion. For COP, slight movements or shifts in posture are more pronounced when compared to joint angles, so minor deviations from the rest posture or small movements between repetitions in joint angle space resulted in large magnitude differences in COP space, and led to accuracy degradation. These differences in feature behaviour led to an accuracy penalty for the dynamic features when compared to the kinematic ones.

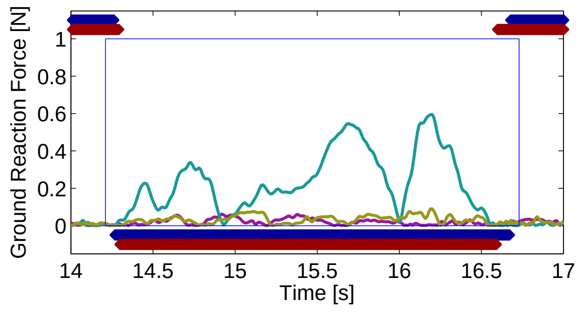
Figure 6.14 shows how segmentation results are impacted by magnitude normalization, initial value normalization, and the choice of input features. As expected, features that are impacted by inter-participant effects, such as height and weight, can improve significantly after normalization. For joint position and GRF data, initial value normalization increases the accuracy mean and decreases the standard deviation, resulting in improvement. For these features, the feature values at rest are heavily influenced by participant



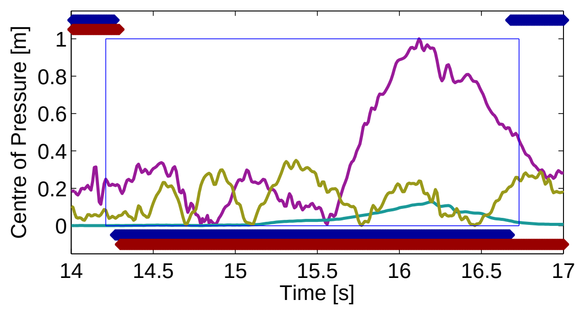
(a) Normalized joint angle data.



(b) Normalized joint torque data.



(c) Normalized GRF data.



(d) Normalized COP data.

Figure 6.13: Segmented squat motion using joint angles, torques, GRF, and COP, as input features. Blue boxes denote manual segment boundaries. Points at the bottom are ground truth (red) and algorithmic (blue) points denoted as p_0, while points at the top are p_1 points.

kinematic/dynamic parameters, and the normalization process reduces this effect. Initial value normalization also improved joint angle accuracy as the initial body joint angles are not consistent between each participant. Some participants chose to start in a fully straight standing posture, while others were hunched over with bent knees in preparation to start the squat. For joint position, angle, and GRF data, magnitude normalization does not provide an additional boost in performance, likely due to the fact that there was not much motion performance variability within this healthy participant dataset. The opposite is observed in the torque data, where magnitude normalization significantly improved the segmentation accuracy, likely due to the fact that torque was always small or zero at the start of the segment, therefore not providing a suitable normalization constant, as can be observed in Figure 6.13b. With the proper normalization, all features achieve over 80% accuracy.

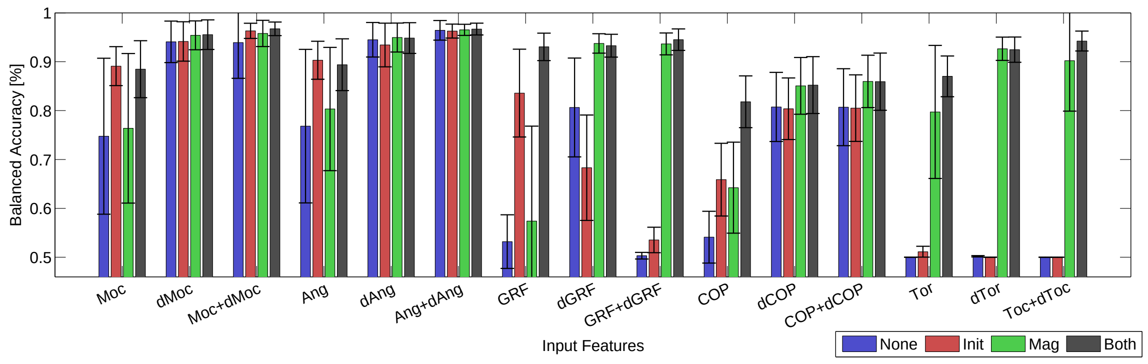


Figure 6.14: Balanced accuracy results for varying initial value (init) and magnitude (mag) normalization, and input features, with $n_{stack} = 10$, $n_{exp} = 5$, and full-segment definition. The five features considered were motion capture joint position (moc), COP, GRF, torque (tor), and angles (ang), combined with their respective derivatives.

Features computed by differentiation of the measured data showed less benefit from normalization, as the differentiation process leaves only relative data, instead of absolute data. In general, the differentiated signals outperform the base features. However, segment point characteristics can be a function of both the absolute features and the relative features, which is reflected in the combined base and differentiated features outperforming individual features.

With the appropriate normalization, joint position, angles, and GRF perform similarly. These results indicate that features that require a significant amount of processing, such as joint angle and torque, do not necessarily outperform raw sensor data such as motion capture joint position and GRF. Employing inverse kinematics and dynamics to calculate joint angles and torques is computationally expensive and requires the specification of a kinematic/dynamic model, as well as the identification of participant-specific kinematic and dynamic parameters, which may be difficult to obtain [14]. The ability to segment accurately without the need for detailed kinematic and dynamic models, as implied by these results, is promising for enabling applications where participants have significant variations in body morphology, such as rehabilitation.

The remaining parameter variations tested did not result in major accuracy differences. Half-segments resulted in lower accuracy for non-kinematic data, but not significantly. A half segment point, which corresponds to a pause in the movement in the kinematic data, does not correspond to a meaningful point for the dynamic features. Variations in n_{stack} generally improved accuracy by a small factor, as more temporal data was included into the classifier's consideration. Increases in n_{exp} had no significant influence on kinematic feature accuracy, but adversely impacted the dynamic feature accuracy. Kinematic features generally changed gradually around the segment boundaries, while the dynamic features changed more rapidly, thus the classifier with high n_{exp} had more difficulty extracting characteristics unique to the segment points. Rectification also had minimal effect, likely because only a single motion type was examined.

6.5 Discussion

6.5.1 Impact of Classifiers

From Tables 6.10, 6.13, 6.14, 6.11, and 6.12, it can be seen that PCA plays a strong role in extracting the key features to obtain a strong segmentation accuracy. In the classifier combinations examined, PCA SVM and PCA ANN typically perform equally or outperform SVM and ANN without PCA. It is worth noting that k-NN performance appears uninfluenced by PCA, but k-NN also requires much more testing time (Tables 6.6 and 6.8) when compared to SVM and ANN, making it difficult to justify the use of k-NN if online operations is required. The PCA analysis in Figure 6.10 suggests that PCA is consistent in selecting combinations of the important moving joints, which means PCA is not only reducing the dimensionality for faster computation, it is also facilitating high accuracy when the feature set is altered by factors such as n_{stack}.

In a majority of the joint angle-based studies, SVM consistently ranks top as a base classifier (Tables 6.10 and 6.11), while ANN outperforms with EMG data (Table 6.19). Since SVM with radial kernel outperforms SVM with linear kernel, it suggests that a linear decision hyperplane is not suitable, and while ANN outperforms with EMG data, SVM seems to be a good default algorithm for segmentation.

Lastly, aggregators did not seem to influence the segmentation accuracy and served only to increase the computation time required (Table 6.9). This is likely due to Gaussian sampling (Section 6.2.1) selecting non-segment points closer to the segmentation boundary for training. These results suggest that the training data sampling scheme is sufficient, since the sampling scheme emphasizes the p_0 points close to the p_1, which are likely to have a higher chance of misclassification.

6.5.2 Impact of Normalization

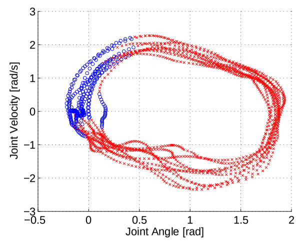
While the proposed method is designed to automatically extract segment edge features, machine learning approaches can be sensitive to input feature scaling. The results show that the normalization allows the healthy and the rehabilitation data to appear more similar (Section 6.4.2), and the temporal stacking and PCA selected a suitable linear combination of features that provide good separability for high segmentation accuracy (Section 6.4.2). Most importantly, the classifier trained on healthy data generalizes both to patient data and unseen exercises, demonstrating the utility and flexibility of the proposed approach for clinical use.

6.5.3 Impact of Feature Sets

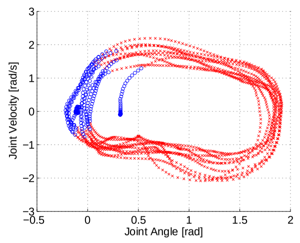
For both the EMG-based studies (Figure 6.12) and the force plate-based studies (Figure 6.13, the manual segments were creating using kinematic data. An examination into the figures produced by these two sets of experiments suggest that, although the segmentation accuracy scored above 80%, the manual segments does not line up with the motion from the EMG or the force plate as well as the joint angles. This suggests that kinematic and dynamic trajectories differ, likely because small motions in kinematic space are difficult to notice until the motion is substantial, or due to the use of passive dynamics and gravity to assist the motion. These findings suggests that each modality may potentially perform with higher accuracy with manual segments tailored to the modality itself.

6.5.4 Healthy and Patient Data Comparison

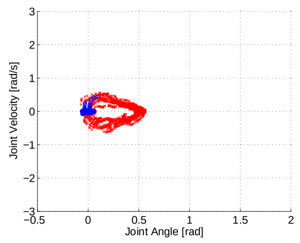
The results show that the pre-processing plays a key role in enabling generalization from healthy to patient data, due to a number of differences between the two. Magnitude differences can be observed between healthy data and patient data. Patients tend to move more slowly with smaller amplitude, when compared to healthy participants. This is illustrated in Figure 6.15 as phase plots, showing both healthy and patient data. In this figure, intra-participant variance is shown to be significant, even within the same healthy participant when the data is collected within a short time period. This difference is magnified between healthy and patient data as the patient data is both smaller in magnitude and noisier. However, as the patient's condition improves over time, the phase plot also begins to look more similar to the healthy data.



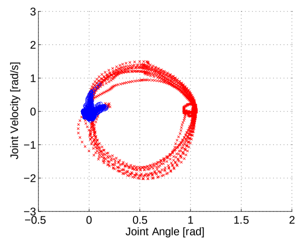
(a) UW 5, session 1, no scaling.



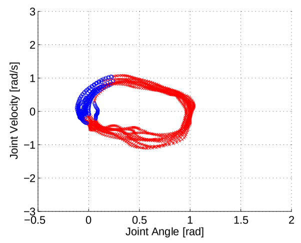
(b) UW 5, session 2, no scaling.



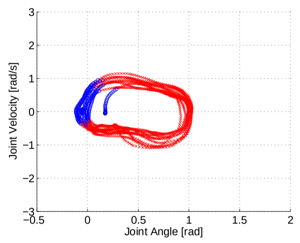
(c) TRI 2, session 1, no scaling.



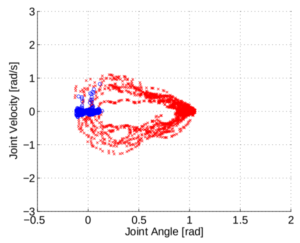
(d) TRI 2, session 10, no scaling.



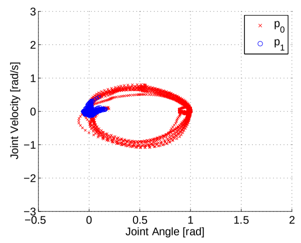
(e) UW 5, session 1, with scaling.



(f) UW 5, session 2, with scaling.



(g) TRI 2, session 1, with scaling.



(h) TRI 2, session 10, with scaling.

Figure 6.15: 8 phase plots of seated knee extension (KEFO-SIT), where the x-axis denotes knee sagittal joint angle and the y-axis denotes knee sagittal joint velocity. The left figures denote the same UW participant performing the exercise at different times within the same hour, illustrating intra-participant variation. The right figures show the same TRI patient performing the exercise at different times over a 2 week span, illustrating significant magnitude differences as the patient improves over time. The top figures show the data without magnitude scaling, while the bottom figures show the data after scaling.

Beyond movement differences, data collection differences also exist between healthy and rehabilitation participants. In general, healthy data was collected in a controlled environment without any obstacles. In contrast, the rehabilitation patient is in an environment filled with training and support equipment. Healthy participants are more likely to understand the exercise better, and can reproduce the exercise more accurately. For accelerometers, joint angle recovery with the Kalman filter [133] assumes that the sensors are rigidly attached to the limb. While this is not a problem for healthy participants, the sensors cannot be attached as tightly to joint replacement patients due to concerns about pain. Sensor placements are also less exact due to the need to avoid surgical sites, bandages, weights, and slings, and may be displaced during movement. Rehabilitation patients are also typically in a more frail state, and may suffer from osteoarthritis, which impacts movement and stability, or obesity and post-surgical swelling and bleeding, which can cause an increase in soft tissue movement [59]. Joint compensation [59], where the patient moves joints other than the prescribed ones to meet the same end-effector goals, due to pain or soreness, is another major problem. Patient data also varies widely due to prescribed patient-specific exercise modifications such as support slings or weights. These differences emphasize the need to perform data validation against both healthy and rehabilitation data.

Reformulating the segmentation problem as a classification problem simplifies the task and allows for machine learning techniques to be used to automatically extract segmentation boundaries without hand crafted features. The various normalization techniques investigated allowed for the healthy template to generalize to both participants and exercises unseen during training, even though the healthy and rehabilitation participants were shown to have very different movement patterns.

6.6 Summary

This chapter proposed a classifier-based segmentation method that recasts the time-series segmentation problem into a 2-class classification problem. The proposed approach allows for segment edge characteristics to be automatically extracted by the machine learning approach instead of relying on hand crafted features. A series of experiments was performed to determine the best configuration for the algorithm and to validate it with several different datasets. From these experiments, the proposed approach was shown to generalize from healthy participant data to rehabilitation participant data, from known primitives to novel primitives, and demonstrated high accuracy over different sensor modalities and feature sets.

To determine the optimal combination of dimensionality reduction, classifier, and aggregator components, different combinations were systematically tested using 3 different datasets. With the UW dataset, consisting of 30 healthy participants performing 5 primitives, PCA SVM resulted in the best performance at $F_1 = 97\%$. For the UT dataset, consisting of 1 healthy participant performing 45 primitives, PCA k-NN performed best at $F_1 = 97\%$. For the UWT dataset, consisting of EMG data of 8 participants performing 9 primitives, PCA ANN performed best at $Acc_{bal} = 83\%$. These results show that while PCA is important to obtain high accuracy, different base classifiers perform optimally while testing against different datasets.

For a given algorithm configuration, classification accuracy is also dependent on the input feature set. Feature normalization to allow datasets to appear more similar can greatly improve generalization from healthy data to rehabilitation patient data. 3 different datasets were examined. The UW dataset was once again used, in order to provide the training data and baseline healthy data segmentation accuracy. SJHCG and TRI datasets, consisting of a wide range of total joint replacement patients, were used to validate generalization. Using initial offset removal, magnitude normalization, and rectification, the proposed method achieved Acc_{bal} = 85-92% on known primitives and Acc_{bal} = 83-87% on novel primitives. The proposed approach compares positively to previous work.

Lastly, to demonstrate algorithm generalizability, different sensor modalities and features were examined. A majority of segmentation algorithms rely on joint position and angles, but few have examined dynamics features such as ground reaction force. With proper normalization, the segmentation algorithm achieves 80+% on all features examined. It was found that joint angle, GRF, and motion capture joint position obtained the best results. This finding indicates that, with appropriate normalization, accurate segmentation can be achieved without the need for accurate and time-consuming individual participant kinematic and dynamic modelling.

Chapter 7

Segmentation based on Inverse Optimal Control

7.1 Introduction

The central nervous system, as the controller of the body, can choose from an unlimited number of joint trajectories in order to carry out an action. However, literature in human motor control over the last three decades has shown that the joint trajectory variance is limited to a much smaller subset, likely due to the fact that the motor system utilizes an optimization strategy to generate the movements [6, 65]. Studies in biomechanics and human motion analysis have proposed many possible cost functions that may be used by the motor system during optimization, such as minimizing time, jerk, or energy, and have shown that the cost function used may differ from task to task [213].

Existing works in IOC typically segment continuous movement into discrete motion primitives before the IOC analysis is applied, and assume that the cost function does not change over the duration of a single primitive. However, a continuous movement sequence may consist of multiple motion primitives, and each primitive may not necessarily share a common cost function. This thesis proposes that if the cost function can be estimated as a function of the motion data, then a change in the cost function may be used as an indication that the motion primitive being performed has changed, and be used to segment the motion. To achieve this, a sliding window over the trajectory data is used to determine the basis weights of the cost function using IOC. The basis weights are temporally filtered to form a time varying feature of the motion trajectory. A threshold can be applied to this feature to perform motion segmentation.

This thesis hypothesizes that the human motion trajectory cost function can be represented as a weighted sum of basis functions and applies IOC to a sliding window over observation data to recover the weights of the windowed trajectory. The weight vector at a given timestep is calculated as the average of all the windows that include that timestep and also have sufficiently low KKT error residual. Motion segmentation is performed on the recovered weights to segment the continuous time series trajectory into discrete actions[1].

7.2 Algorithm

The cost function $J(\mathbf{x})$ that is minimized to generate a given motion is modelled as a weighted sum of basis cost functions $J_{bf}(\mathbf{x})$:

$$J(\mathbf{x}) = \sum_{i=0}^{n_{bf}} c_i J_{bf,i}(\mathbf{x}) \tag{7.1}$$

where $\mathbf{x}$ is the variable that is manipulated to minimize the cost. The KKT approach [169, 63] is used to determine the basis weights $\mathbf{c}$ of the observed trajectory, velocity, and acceleration, which are collectively denoted as $\mathbf{Q_{obs}} = [\mathbf{q_{obs}}; \dot{\mathbf{q}}_{\mathbf{obs}}; \ddot{\mathbf{q}}_{\mathbf{obs}}]$. Previous studies assume that the motion trajectory is segmented [22, 209] or that the motion consists of a single motion primitive or cost function [154, 179]. To the authors' knowledge, this thesis is the first to remove the above assumptions. Instead, IOC is performed sequentially over sliding windows over the time series data. A change in the basis function weights is used to determine when the motion objective has changed, indicating a segment point. The approximate IOC method proposed in [169] is adapted to allow for changing basis weights by handling windows of arbitrary length and rejecting degenerate estimates.

This section describes two components of the trajectory optimization framework: TO, where $\mathbf{c}$ is known and $\mathbf{Q_{obs}}$ is to be generated, and IOC, where $\mathbf{Q_{obs}}$ is known, and $\mathbf{c}$ is to be estimated. The IOC estimates the basis weights $\hat{\mathbf{c}}$ based on $\mathbf{Q_{obs}}$, or parts of $\mathbf{Q_{obs}}$, while the TO is used to generate simulation data and determine goodness-of-fit by comparing $\mathbf{Q_{obs}}$ to its estimate $\hat{\mathbf{q}}_{\mathbf{obs}}$ as generated by $\hat{\mathbf{c}}$.

[1]Portions of this work has been published in the IEEE-RAS International Conference on Humanoid Robots [128].

7.2.1 Trajectory Representation

The trajectories in this chapter are represented as piecewise 5^{th} order polynomials, where each individual polynomial is a spline of the form $q_p = p_5t^5 + p_4t^4 + p_3t^3 + p_2t^2 + p_1t + p_0$. The spline is used to reduce the problem dimensionality by allowing modelling to occur on the spline control knots instead of the full trajectory, and to avoid bias in estimation [169]. It also allows for the trajectory derivatives to be estimated analytically.

Given the set of splining control knot locations $\mathbf{t_{ck}}$, joint angles $\mathbf{q_{ck}} = \mathbf{q}(\mathbf{t_{ck}})$, velocities $\dot{\mathbf{q}}_{\mathbf{ck}} = \dot{\mathbf{q}}(\mathbf{t_{ck}})$, and accelerations $\ddot{\mathbf{q}}_{\mathbf{ck}} = \ddot{\mathbf{q}}(\mathbf{t_{ck}})$, a polynomial is constructed between each pair of knots. The coefficients for the 5^{th} order polynomial between control knot $t_{ck,k}$ and $t_{ck,k+1}$ are found by constructing a variation of the Vandermonde matrix $\mathbf{V}$ [204]:

$$
\begin{aligned}
\mathbf{V} &= \begin{bmatrix} t_k^5 & t_k^4 & t_k^3 & t_k^2 & t_k^1 & 1 \\ 5t_k^4 & 4t_k^3 & 3t_k^2 & 2t_k^1 & 1 & 0 \\ 20t_k^3 & 12t_k^2 & 6t_k^1 & 2 & 0 & 0 \\ t_{k+1}^5 & t_{k+1}^4 & t_{k+1}^3 & t_{k+1}^2 & t_{k+1}^1 & 1 \\ 5t_{k+1}^4 & 4t_{k+1}^3 & 3t_{k+1}^2 & 2t_{k+1}^1 & 1 & 0 \\ 20t_{k+1}^3 & 12t_{k+1}^2 & 6t_{k+1}^1 & 2 & 0 & 0 \end{bmatrix} \\
\mathbf{q_c} &= \begin{bmatrix} q_k & \dot{q}_k & \ddot{q}_k & q_{k+1} & \dot{q}_{k+1} & \ddot{q}_{k+1} \end{bmatrix} \\
\mathbf{p} &= \mathbf{V}\mathbf{q_c}^{-1}
\end{aligned}
\tag{7.2}
$$

where $\mathbf{p}$ are the coefficients of the polynomial and the control knot subscript was removed for brevity (i.e. $t_k = t_{ck,k}$). The number of control knots is a fixed value, and the control knots are evenly distributed in the window.

In this application, the piecewise polynomial is preferred over the B-spline method since the piecewise polynomial is a local splining technique and allows for arbitrary addition and removal of control knot points without changing the trajectory in other parts of the spline. B-spline is a global splining method, meaning that the generated trajectory is affected by the presence of all control knot points, which is not suitable for an application where arbitrary window length and location are needed.

7.2.2 Trajectory Optimization

Given $\mathbf{c}$ and $\mathbf{t_{ck}}$, the goal of TO is to generate the $\mathbf{q_{ck}}$, $\dot{\mathbf{q}}_{\mathbf{ck}}$, and $\ddot{\mathbf{q}}_{\mathbf{ck}}$ that minimize $J(\mathbf{x})$. These control knots, when optimized (denoted as $\mathbf{q}^*_{\mathbf{ck}}$), are used to generate a spline that approximates the optimal trajectory $\mathbf{Q_{obs}} = sp(\mathbf{q}^*_{\mathbf{ck}}, \dot{\mathbf{q}}^*_{\mathbf{ck}}, \ddot{\mathbf{q}}^*_{\mathbf{ck}})$. The general form of the constrained optimization problem is as follows:

$$\min_{\mathbf{x}} J(\mathbf{x}) \in h(\mathbf{x}) = 0, g(\mathbf{x}) \leq 0 \tag{7.3}$$

where $h(\mathbf{x})$ are the equality constraints, and $g(\mathbf{x})$ are the inequality constraints. The TO problem is obtained by modifying Equation 7.3 into:

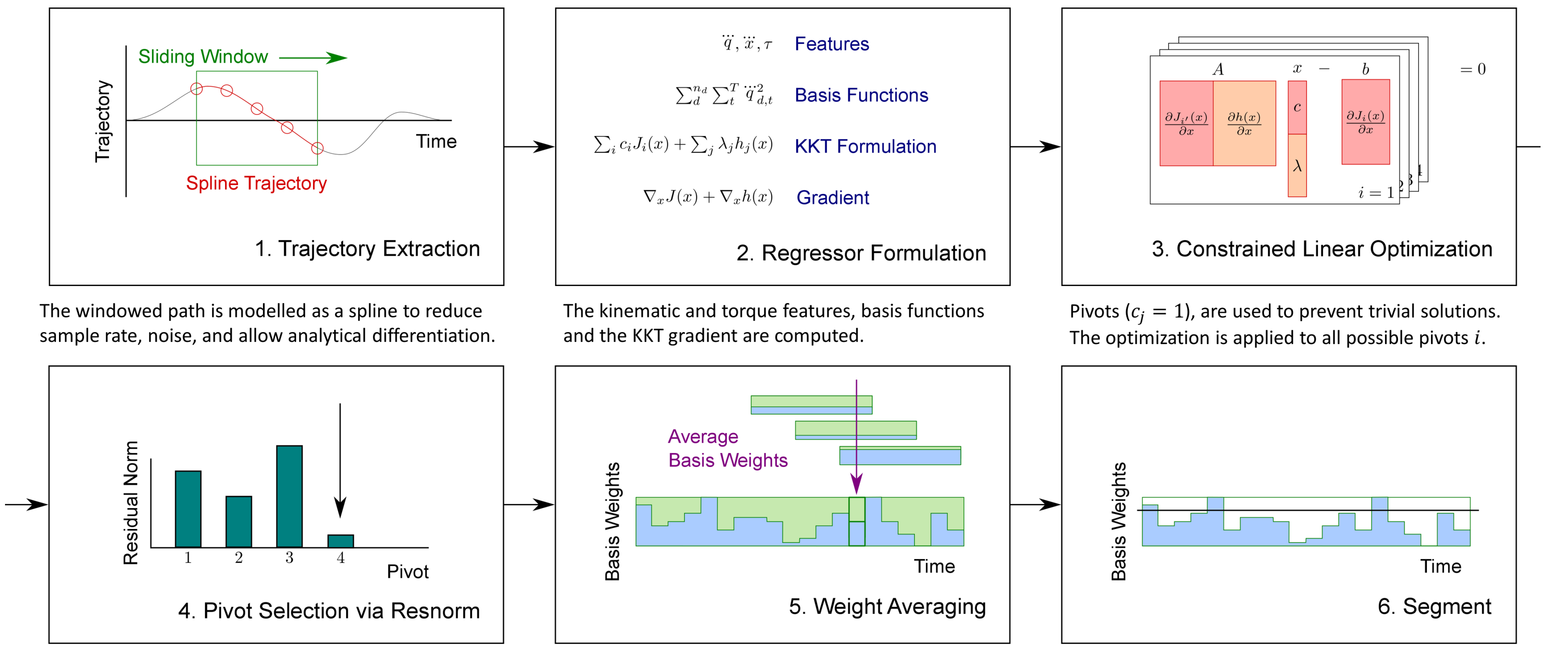
$$\begin{aligned} \min_{\mathbf{x}=\mathbf{q_{ck}},\dot{\mathbf{q}}_{\mathbf{ck}},\ddot{\mathbf{q}}_{\mathbf{ck}}} J(\mathbf{x}) &= \sum_{i=0}^{n_{bf}} c_i J_{bf,i}(\mathbf{Q_{obs}}) \\ \in h(\mathbf{x}) &= \begin{cases} \mathbf{q}(\mathbf{t_{const,q}}) = \mathbf{q_{const}} \\ \dot{\mathbf{q}}(\mathbf{t_{const,dq}}) = \dot{\mathbf{q}}_{\mathbf{const}} \\ \ddot{\mathbf{q}}(\mathbf{t_{const,ddq}}) = \ddot{\mathbf{q}}_{\mathbf{const}} \end{cases} \end{aligned} \tag{7.4}$$

where $\mathbf{q_{const}}$, $\dot{\mathbf{q}}_{\mathbf{const}}$, and $\ddot{\mathbf{q}}_{\mathbf{const}}$ denote the joint position, velocity, and acceleration constraints, respectively, and $\mathbf{t_{const,q}}$, $\mathbf{t_{const,dq}}$, $\mathbf{t_{const,ddq}}$ refer to their corresponding time points. These constraints form the equality constraints $h(\mathbf{x})$ of the system. This thesis does not include any inequality constraints $g(\mathbf{x})$, but potential inequality constraints can include joint and torque limits. The basis functions are then normalized by taking the average basis function value over all the subjects in a given dataset.

To solve the TO problem, the interior point optimization method is used [40]. An initial trajectory is created by a 5^{th} order polynomial, constrained for starting and ending $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$. The joint angles at $\mathbf{t_{ck}}$ are extracted from this trajectory and used to initialize $\mathbf{q}(\mathbf{t_{ck}})$. At each optimization step, $\mathbf{q_{ck}}$, $\dot{\mathbf{q}}_{\mathbf{ck}}$, and $\ddot{\mathbf{q}}_{\mathbf{ck}}$ are used to create the spline, then all the features needed to calculate J_{bf} are determined. After the J_{bf} calculations, J_{bf} is normalized by the mean value over all J_{bf} for a given dataset.

7.2.3 Inverse Optimal Control

In the IOC problem, $\mathbf{q_{ck}}$ is known, and $\mathbf{c}$ must be estimated. To achieve this, the IOC is formulated as an inverse KKT problem [98, 179, 169, 63]. By minimizing the residuals of the KKT equations, a near-optimal solution can be found (Figure 7.1).


Sliding Window
Trajectory
Time
Spline Trajectory
1. Trajectory Extraction
$\dddot{q}, \dddot{x}, \tau$ Features
$\sum_d^{n_d} \sum_t^T \dddot{q}_{d,t}^2$ Basis Functions
$\sum_i c_i J_i(x) + \sum_j \lambda_j h_j(x)$ KKT Formulation
$\nabla_x J(x) + \nabla_x h(x)$ Gradient
2. Regressor Formulation
A
x
$-$
b
$= 0$
$\frac{\partial J_{i'}(x)}{\partial x}$
$\frac{\partial h(x)}{\partial x}$
c
λ
$\frac{\partial J_i(x)}{\partial x}$
$i = 1$
2
3
4
3. Constrained Linear Optimization
Residual Norm
1
2
3
4
Pivot
4. Pivot Selection via Resnorm
Average
Basis Weights
Basis Weights
Time
5. Weight Averaging
Basis Weights
Time
6. Segment

The windowed path is modelled as a spline to reduce sample rate, noise, and allow analytical differentiation.

The kinematic and torque features, basis functions and the KKT gradient are computed.

Pivots ($c_j = 1$), are used to prevent trivial solutions. The optimization is applied to all possible pivots i.

The final pivot selected is determined by the smallest residual norm.

The weight at t (arrow) is determined by averaging all windows that contain that t.

Segmentation is performed by thresholding on a basis weight.

Figure 7.1: Flow diagram for the IOC basis weight recovery.

Given the problem formulation in Equation 7.4, the KKT Lagrangian $L(\mathbf{x} = \mathbf{q_{ck}}, \dot{\mathbf{q}}_{\mathbf{ck}}, \ddot{\mathbf{q}}_{\mathbf{ck}})$ and its gradient $\nabla_{\mathbf{x}} L(\mathbf{x})$ are defined as:

$$L(\mathbf{x}) = \sum_{i=0}^{n_{bf}} \hat{c}_i J_{bf,i}(\mathbf{Q_{obs}}) + \sum_{j=0}^{n_h} \lambda_j h_j(\mathbf{Q_{obs}})$$
$$\nabla_{\mathbf{x}} L(\mathbf{x}) = \sum_{i=0}^{n_{bf}} \hat{c}_i \nabla_{\mathbf{x}} J_{bf,i}(\mathbf{Q_{obs}}) + \sum_{j=0}^{n_h} \lambda_j \nabla_{\mathbf{x}} h_j(\mathbf{Q_{obs}})$$

where the partial differential of the gradient $\nabla_{\mathbf{x}}$ is calculated with respect to the state variables $\mathbf{q_{ck}}$, $\dot{\mathbf{q}}_{\mathbf{ck}}$, and $\ddot{\mathbf{q}}_{\mathbf{ck}}$, λ are the Lagrangian multipliers on $h(\mathbf{x})$, and $\mathbf{Q_{obs}}$ is constructed from the spline representation of the trajectory. The condition that must be met to ensure optimality is:

$$\nabla_{\mathbf{x}} L(\mathbf{Q_{obs}}) = 0 \tag{7.5}$$

If it is assumed that the system is not strictly optimal, but rather only approximately optimal [98], then Equation 7.6 is minimized but is not strictly zero:

$$\min_{\hat{\mathbf{c}},\lambda} \nabla_{\mathbf{x}} L(\mathbf{Q_{obs}}) \tag{7.6}$$
$$\in \hat{\mathbf{c}} \geq 0$$

Since the KKT equations are linear with respect to the unknown variables $\hat{\mathbf{c}}$ and λ, Equation 7.6 can be written as a least square problem in the form of $\mathbf{Az}$, as shown in Figure 7.2, and solved computationally efficiently. To solve this constrained linear least squares problem, the active set method is used [73]. The gradient is calculated numerically.

In order to prevent trivial solutions, one of the values of $\hat{\mathbf{c}}$ must be set to a non-zero value. This term, denoted as the *pivot*, may be selected with some prior knowledge of the nature of the cost functions [169]. In this thesis, no prior knowledge is assumed, so all basis functions are used as the pivot, and the best fit is selected by selecting the entry with the smallest KKT error residual. To construct the pivot $\mathbf{b_i}$, the i^{th} column of $\mathbf{A_0}$, c_i is constrained to be 1.

To allow for proper comparison between the different pivots, the IOC basis weights are normalized by dividing all the recovered weights $\hat{\mathbf{c}}$ by $\sum \hat{c}$. The normalized weights are then used to calculate the residual norm again, which has the effect of scaling the residual norm by $\sum \hat{c}$.

The IOC process is applied on a sliding window of arbitrary length over the Q_{obs} to recover the $\hat{\mathbf{c}}$ of the trajectory over that window. Depending on the size and location of the

$$
\mathbf{A_0} = \begin{bmatrix} \nabla_{\mathbf{x}} J_{bf,1} & \nabla_{\mathbf{x}} J_{bf,2} & \cdots & \nabla_{\mathbf{x}} h_1 & \nabla_{\mathbf{x}} h_2 & \cdots \end{bmatrix}
$$

$$
= \begin{bmatrix}
\frac{\partial(J_{bf,1})}{\partial(q_{d=1,ck=1})} & \frac{\partial(J_{bf,2})}{\partial(q_{d=1,ck=1})} & \cdots & \frac{\partial(h_1)}{\partial(q_{d=1,ck=1})} & \frac{\partial(h_2)}{\partial(q_{d=1,ck=1})} & \cdots \\
\frac{\partial(J_{bf,1})}{\partial(q_{d=2,ck=1})} & \frac{\partial(J_{bf,2})}{\partial(q_{d=2,ck=1})} & \cdots & \frac{\partial(h_1)}{\partial(q_{d=2,ck=1})} & \frac{\partial(h_2)}{\partial(q_{d=2,ck=1})} & \cdots \\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots \\
\frac{\partial(J_{bf,1})}{\partial(q_{d=n_d,ck=n_{ck}})} & \frac{\partial(J_{bf,2})}{\partial(q_{d=n_d,ck=n_{ck}})} & \cdots & \frac{\partial(h_1)}{\partial(q_{d=n_d,ck=n_{ck}})} & \frac{\partial(h_2)}{\partial(q_{d=n_d,ck=n_{ck}})} & \cdots \\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots \\
\frac{\partial(J_{bf,1})}{\partial(\dot{q}_{d=n_d,ck=n_{ck}})} & \frac{\partial(J_{bf,2})}{\partial(\dot{q}_{d=n_d,ck=n_{ck}})} & \cdots & \frac{\partial(h_1)}{\partial(\dot{q}_{d=n_d,ck=n_{ck}})} & \frac{\partial(h_2)}{\partial(\dot{q}_{d=n_d,ck=n_{ck}})} & \cdots \\
\vdots & \vdots & \cdots & \vdots & \vdots & \cdots \\
\frac{\partial(J_{bf,1})}{\partial(\ddot{q}_{d=n_d,ck=n_{ck}})} & \frac{\partial(J_{bf,2})}{\partial(\ddot{q}_{d=n_d,ck=n_{ck}})} & \cdots & \frac{\partial(h_1)}{\partial(\ddot{q}_{d=n_d,ck=n_{ck}})} & \frac{\partial(h_2)}{\partial(\ddot{q}_{d=n_d,ck=n_{ck}})} & \cdots
\end{bmatrix}
$$

$$
\mathbf{z_0} = \begin{bmatrix} \hat{c}_1 & \hat{c}_2 & \cdots & \lambda_1 & \lambda_2 & \cdots \end{bmatrix}^T
$$

Figure 7.2: Formulation of the least squares problem for the IOC. $\mathbf{A_0}$ denotes the gradient, differentiated against the k^{th} control knot point and d^{th} DOF, before the array is split and the pivot basis function is extracted. $\mathbf{z_0}$ denotes the variables to recover.

window in the time series data stream, it may not be possible to recover the weights, leading to a degenerate solution. In these cases, the error residual from Equation 7.6 is very high, and typically corresponds to a negative $\hat{\mathbf{c}}$ if the $\mathbf{c} \geq 0$ optimization constraint in Equation 7.6 is relaxed. To detect these degenerate solutions, the residual norm $||\mathbf{A_i z_i} + \mathbf{b_i}||_2^2$ can be checked as an indicator of the quality of the $\hat{\mathbf{c}}$ estimates. Once the pivot has been selected, the trajectory $\hat{\mathbf{q}}_{\mathbf{obs}}$ corresponding to $\hat{\mathbf{c}}$ can be generated via TO if RMSE is required.

7.2.4 Threshold Segmentation

When the observed data moves from one task to another, the hypothesis is that the controller, and thus the basis function weights, would change to reflect the change in the task. Segmentation is performed by examining each basis weight profile and varying a threshold. If the weight at time t is above the threshold, then it is assigned a label of p_1. Otherwise, it is assigned a label of p_0. The threshold that obtained the highest match between this method and the the ground truth data is retained. This procedure is repeated with p_1 label occurring if the weight is below the threshold, and over all the basis weight types.

7.2.5 Analysis via the Segmentation Framework

This algorithm can be analyzed via the segmentation framework proposed in Section 3.

Segment definition.

The segment definition is specified by domain experts in physiotherapy. The definition specifies that an extension and a flexion movement makes one segment.

Data collection.

The primary sensor modality used is motion capture, with the derived feature set of joint angles.

The source and target populations for the training and testing are healthy participants performing exercise data. The manual segment data are sourced from human observers from video playback.

Application specific requirements.

To meet the target application of physiotherapy, the algorithm must be able to handle inter-participant and inter-primitive variability. While typical clinical applications require online testing performance, the proposed algorithm is currently in an early prototyping stage and thus the training or testing time requirements are relaxed.

Algorithm design.

The sensor data is converted to joint angles using inverse kinematics. Other necessary features are then calculated using forward dynamics.

This algorithm is categorized as a sliding window offline unsupervised segmentation technique. There is no training component, while testing is offline. This algorithm has no segment identification component. The algorithm classifies between motion and non-motion time points.

Verification.

This algorithm is validated with healthy datasets. $Acc_{precision}$, Acc_{recall}, and Acc_{bal} accuracy measures are used with classification by data point labels to produce the segmentation accuracy.

7.3 Experiments

Two sets of experiments were ran to verify the algorithm, in order to verify that a trajectory generated by TO can successfully have its basis weights recovered by IOC using simulation data, and to examine the segmentation accuracy of the IOC weights with real data.

All processing and algorithm implementation were done in MATLAB 8.0 [212].

7.3.1 Simulation

The first set of experiments utilize TO to generate repetitions of a given exercise using TO and a given set of basis function weights, then recover the weights using IOC. However, with real life data, the length of trajectory that shares a common cost function is uncertain, so simulation testing must be done where TO window length is different than IOC window length to verify that the weights can be properly recovered.

In simulation, degenerate cases can be detected by comparing the estimated $\hat{\mathbf{c}}$ against the ground truth $\mathbf{c}$ used to generate the test trajectory. For human data, there is no way to determine the ground truth $\mathbf{c}$ to verify $\hat{\mathbf{c}}$. Therefore, the residual norm thresholds, which will change according to the J_{bf} normalization values, are tuned using the simulation data.

To generate the trajectory, TO was used to generate a set of $\mathbf{Q_{obs}}$ with a known set of $\mathbf{c}$ values. Multiple repetitions of a squat or hip extension task were simulated by minimizing J_{ddq}, J_{ddx}, and J_{tau} (Table 7.1), or a weighted sum of all three criteria. Each repetition had 9 $h(\mathbf{x})$ constraints, corresponding to the position, velocity, and acceleration constraints for 3 key poses during the task: standing, squatting or hip extension, then standing again, placed at the start, middle, and end of the TO trajectory (Figure 7.3). Each repetition had a duration of 2 s. $\mathbf{Q_{obs}}$ was modelled as a 3 DOF system, corresponding to the ankle $\mathbf{q_{ankle}}$, knee $\mathbf{q_{knee}}$, and hip $\mathbf{q_{hip}}$.

Algorithm Settings

For IOC reconstruction, n_{ck} was set to 5 points every 1 s, evenly distributed over $\mathbf{Q_{obs}}$. A sliding window of width 2 s, incrementing by 0.2 s, was passed over the trajectory. $h(\mathbf{x})$ constraints were set so that the joint position, velocity, and acceleration constraints were placed at the start, middle, and end of the IOC window such that they coincided with n_{ck} points. The IOC pivot that resulted in the smallest residual was selected as the most suitable pivot.

All features were calculated from the joint angle measurements. Angular acceleration $\ddot{\mathbf{q}}$ and jerk $\dddot{\mathbf{q}}$ values were calculated from the derivatives of the joint angle spline. The Cartesian acceleration $\ddot{\mathbf{x}}$ values were calculated via forward kinematics, while the Cartesian jerk $\dddot{\mathbf{x}}$ values were calculated from numerical differentiation of the Cartesian acceleration. Torque τ values were calculated using anthropometric table [60] data for the dynamic

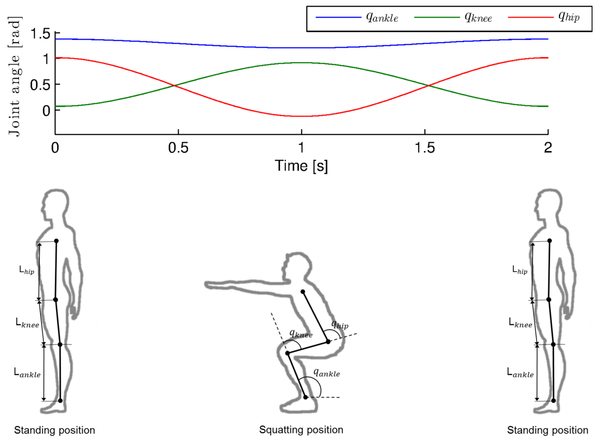


Figure 7.3: Generation of the simulation squat TO. The first and third set of $h(\mathbf{x})$ constraints denote the standing position, where the $\mathbf{q}$ are set to simulate a standing person, and the second set of $h(\mathbf{x})$ constraints denote the squatting position. All $h(\mathbf{x})$ constraints corresponding to the $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are zero, denoting a stationary person at the key poses.

Table 7.1: Basis functions used for simulation.

Basis function	**Definition**
Angular acceleration (ddq)	$J_{ddq} = \sum_{d}^{n_d} \sum_{t}^{T} \ddot{q}_{d,t}^2$
Cartesian acceleration (ddx)	$J_{ddx} = \sum_{t}^{T} \ddot{x}_{n_d,t}^2$
Torque (tau)	$J_{tau} = \sum_{d}^{n_d} \sum_{t}^{T} \tau_{d,t}^2$

parameters and inverse dynamics through Symoro [99]. Torque change $\dot{\tau}$ and effort $\ddot{\tau}$ values were obtained from numerical differentiation.

Visual inspection was used to confirm a match between the original TO weights and the IOC reconstructed weights. Windows of inaccurate weight reconstructions were removed by adjusting the residual norm threshold to tune the residual norm threshold value.

7.3.2 Human Data

The second set of experiments applied the IOC algorithm to recover weights of human data. 2 datasets were considered, the UW and the URFI datasets. With human motion data, we are uncertain if the basis functions proposed in the literature are comprehensive as missing basis functions will lead to poor reconstruction. Human data is also noisy, and may require filtering, which can change the trajectory and impact the IOC results.

Datasets

The first dataset considered for this experiment set was the URFI dataset, using the joint angles and velocities calculated from the motion capture data. The URFI dataset contains a single exercise performed with instruction, resulting in motion that was very similar between the different participants. This forms a simple baseline for the IOC recovery due to the simplicity of the dataset.

The second dataset examined in this experiment set was the UW dataset, using the joint angles and velocities calculated from the motion capture data. The UW dataset contains the data of 30 participants performing a variety of exercises, and thus serves to test the algorithm's ability to generalize. The UW dataset was split into two sub-datasets, where one set used the ankle-to-hip (denote as UW ankle or ankle-first) kinematic chain, while the other was a hip-to-ankle (denote as UW hip or hip-first) chain, due to the nature of the motions being modelled.

The feature calculation and IOC basis weight recovery are identical to the simulation approach described in Section 7.3.1. The basis functions used for the human testing [22] can be found in Table 7.2. Although the basis functions in this thesis were chosen using the set recommended by Berret *et al.* [22], which is based on upper body tasks, the chosen basis functions have been applied to lower-body motion as well, including joint and end-effector kinematics [103, 53, 172], joint torque [191, 155, 197, 103, 53] and its higher derivatives [197], and power [38, 103].

Table 7.2: Basis functions used for human data testing [22], summed over all n_d DOFs and T time. M denotes the inertial matrix.

Basis function	Definition
Angular acceleration (ddq)	$J_{ddq} = \sum_{d}^{n_d} \sum_{t}^{T} \ddot{q}_{d,t}^2$
Angular jerk (dddq)	$J_{dddq} = \sum_{d}^{n_d} \sum_{t}^{T} \dddot{q}_{d,t}^2$
Cartesian jerk (dddx)	$J_{dddx} = \sum_{t}^{T} \dddot{x}_{n_d,t}^2$
Torque (tau)	$J_{tau} = \sum_{d}^{n_d} \sum_{t}^{T} \tau_{d,t}^2$
Torque change (dtau)	$J_{dtau} = \sum_{d}^{n_d} \sum_{t}^{T} \dot{\tau}_{d,t}^2$
Torque effort (ddtau)	$J_{ddtau} = \sum_{d}^{n_d} \sum_{t}^{T} \ddot{\tau}_{d,t}^2$
Kinetic energy (en)	$J_{en} = \sum_{d}^{n_d} \sum_{t}^{T} \dot{q}_{d,t} \mathbf{M}(\mathbf{q}) \dot{q}_{d,t}$
Power	$J_{power} = \sum_{d}^{n_d} \sum_{t}^{T} (\dot{q}_{d,t} \tau_{d,t})^2$

The ground truth of the segmentation is generated as follows: each data point is assigned a label. If it is within a manual segment, then the label is p_1. If it is not within a manual segment, then the label is p_0. These results were then compared to the algorithmic segment labels.

The validation metric used was the Acc_{bal}.

7.4 Results

7.4.1 Simulation

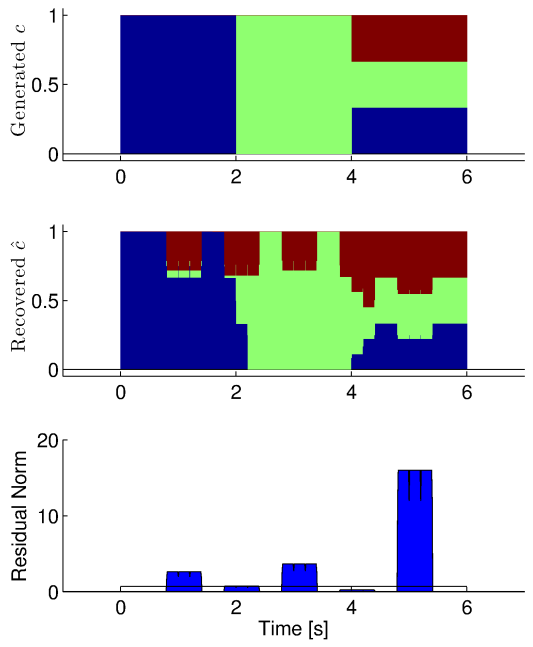
Two sets of results were generated:

1. Residual norm thresholds obtained by selecting a threshold that removed all incorrectly reconstructed windows.
2. Recovered basis weights from simulated TO trajectories.

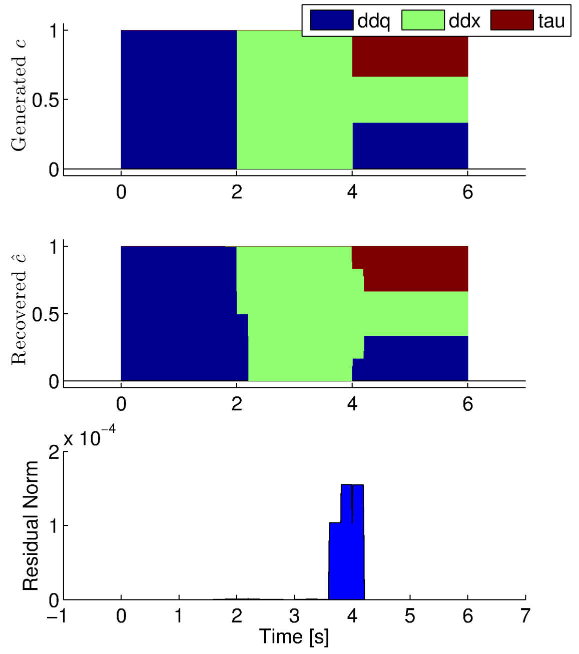
Experiments with the simulation data show that degenerate situations can occur in two different cases. The first case is if the basis functions hypothesized during IOC do not correspond to the cost function used to generate the motion. This leads to an $\mathbf{A_0}$ matrix that does not provide the correct basis functions that can sufficiently minimize Equation 7.6 and leads to a high residual norm value. These cases can be rejected by the residual norm threshold test if properly tuned (Figure 7.4), and be used to produce threshold values for the human data testing (Table 7.3). See Figure 7.5 for examples of the simulation reconstruction.

The second case is if the windowed part of the trajectory does not provide sufficient information for the IOC model. This could happen if the window is insufficiently long, or if there are the same or more $h(\mathbf{x})$ constraints than knot points (i.e. $n_h \geq n_{ck}$) in the IOC window. In this case, the residual norm test will be inaccurate, especially if there are enough $h(\mathbf{x})$ constraints to satisfy the least squares without the basis function columns in the $\mathbf{A_0}$ matrix, which will result in a low residual norm value but a degenerate case. This can be avoided by ensuring that all IOC windows have more control knots than $h(\mathbf{x})$ constraints.

Figure 7.5 shows the that the IOC can successfully recover the simulated TO weights.

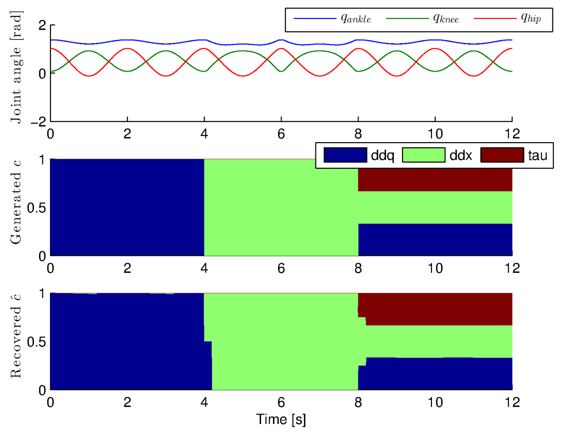


(a) No residual norm threshold enabled. Average residual norm is 2.23×10^{0}

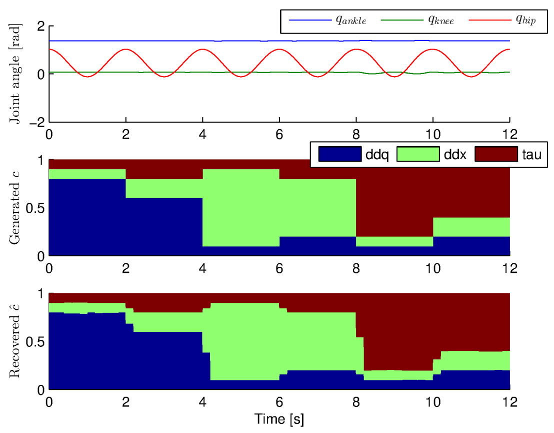


(b) Residual norm threshold properly tuned. Average residual norm is 1.39×10^{-5}

Figure 7.4: Simulation data, illustrating the generated $\mathbf{c}$ (top), recovered blended $\hat{\mathbf{c}}$ (middle), and the residual norm (bottom). Without the residual norm threshold (left), the IOC reconstruction results in blocks of incorrect segment. When the threshold is properly tuned (right), the IOC basis weights produces a good match to the TO basis weights.



(a) Squat data. The simulated cost vectors are [1; 0; 0], [0; 1; 0], and [1; 1; 1].



(b) Hip extension data. The simulated cost vectors are [0.8; 0.1; 0.1], [0.6; 0.2; 0.2], [0.1 0.8 0.1], [0.2 0.6 0.2], [0.1 0.1 0.8], and [0.2 0.2 0.6].

Figure 7.5: Simulation data, illustrating $\mathbf{Q_{obs}}$ (top), the recovered blended $\hat{\mathbf{c}}$ (middle), and the original $\mathbf{c}$ (bottom). The data was created using three basis functions, J_{ddx} (blue), J_{ddq} (green) and J_{tau} (red), and varying the cost weights. The jagged components in the $\hat{\mathbf{c}}$ plot are caused by window averaging when a given window overlaps two different basis functions.

Table 7.3: Residual norm threshold values determined by simulation for the squat motion. The threshold is determined by selecting a residual norm that would remove erroneous weight reconstruction. The actual value used is set to be 95% of the threshold listed here to reduce numerical issues. The residual norm value is a function of the basis function normalization and thus changes per dataset.

Dataset	**Threshold** [unitless]
UW Ankle	3.80
UW Hip	5.75
URFI	2.32

7.4.2 Human Data

Three sets of results were generated:

1. Basis weight distribution, as recovered from IOC.

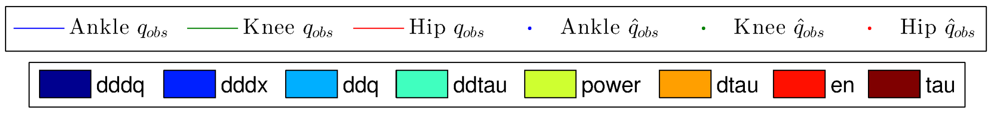
2. Post-threshold residual norm values.

3. Post-threshold RMSE value. The basis weights recovered from the IOC are used with TO to generate the joint angle trajectory, and compared to the original trajectory.

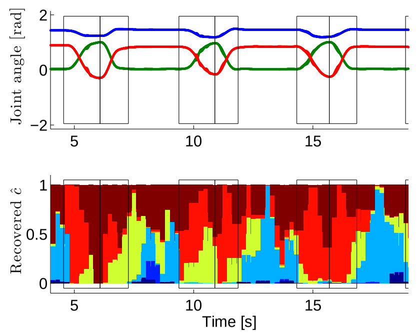
4. Segmentation accuracy. Two different methods of calculating the overall segmentation accuracy are reported. The first calculated the maximum segmentation accuracy for each set of observations individually and averaged them, thus allowing for different selected basis functions and segmentation thresholds. A second set obtained the maximum accuracy if a single basis function and threshold was used across all observations.

Table 7.4: The RMSE and residual norm (resnorm) mean and standard deviation (stddev) for the UW Ankle, UW Hip, and URFI datasets. The RMSE is calculated between the trajectory generated by TO and the observed trajectory. Both RMSE and residual norm values are averaged over all the sliding windows in a given trajectory, and are weighted within each dataset by datapoint count.

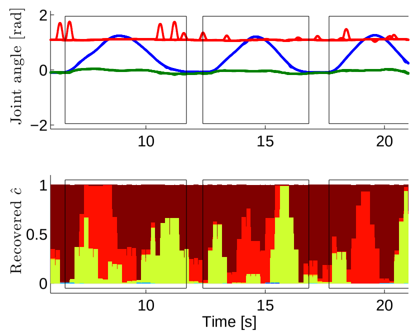
Data		**RMSE** [rad]		**Resnorm [unitless]**	
Dataset	**Primitive**	**Mean**	**StdDev**	**Mean**	**StdDev**
UW Ankle	SQUA-STD	0.006	0.010	0.128	0.233
	STSO-SIT	0.006	0.008	0.276	0.335
	AVG	0.006	0.009	0.212	0.291
UW Hip	HFEO-SUP	0.018	0.039	0.086	0.178
	KEFO-SIT	0.026	0.047	0.100	0.186
	KHEF-SUP	0.013	0.034	0.035	0.085
	AVG	0.019	0.040	0.073	0.149
URFI	SQUA-STD	0.006	0.013	0.079	0.131



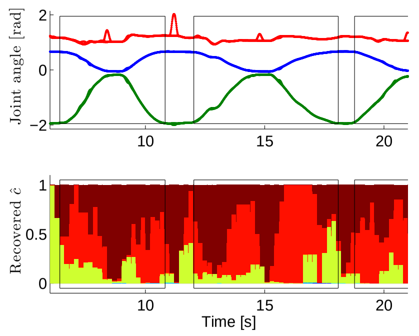
(a)



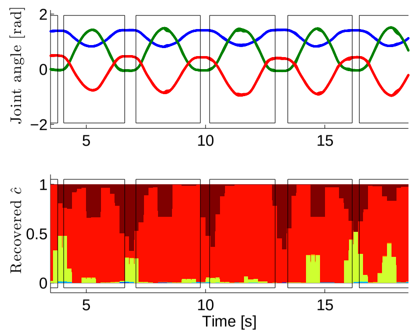
(b) URFI squats.



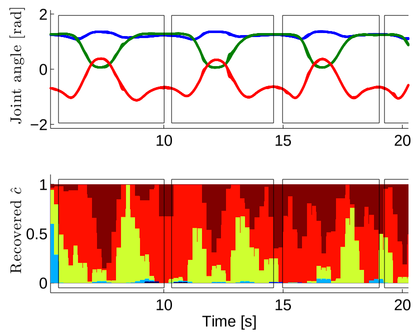
(c) UW hip flexion.



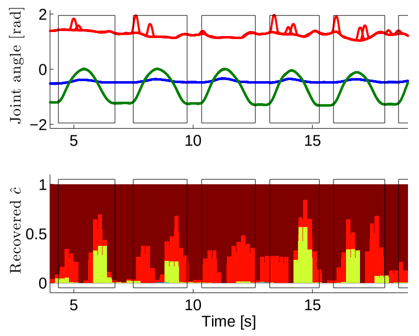
(d) UW knee/hip flexion.



(e) UW squats.



(f) UW sit to stand.



(g) UW knee extension.

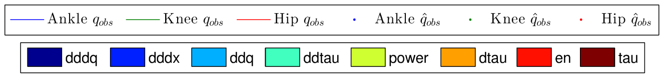
Figure 7.6: Reconstructed trajectory (top) and recovered $\hat{\mathbf{c}}$ (bottom) for the UW and URFI dataset. For all windows, the time axis is kept constant at 15 seconds long, showing a variety in segment length over each primitive. The 3 hip-first motions, hip flexion, knee/hip flexion, and knee extension, have higher reconstruction error in the last joint (red, ankle), leading to high RMSE errors in these 3 motion types (Table 7.6).

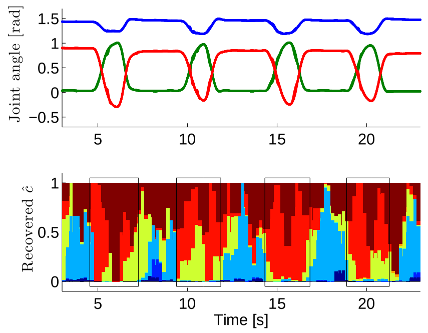
Table 7.5: Segmentation accuracy of UW Ankle, UW Hip, and URFI datasets using a single basis function and threshold for each exercise (when the exercise is denoted as the primitive) and dataset (where ‘All’ is denoted as the exercise). The basis weights columns report the number of times that particular basis weight was selected for the maximum segmentation accuracy.

Data		**Combined Accuracy**			**Basis Function Selected**							
Dataset	**Primitive**	$Acc_{precision}$	Acc_{recall}	Acc_{bal}	J_{dddq}	J_{dddx}	J_{ddtau}	J_{ddq}	J_{power}	J_{dtau}	J_{en}	J_{tau}
UW Ankle	SQUA-STD	91	56	80	0	0	0	0	0	0	1	0
	STSO-SIT	77	88	72	0	0	0	1	0	0	0	0
	All	92	52	67	0	0	0	0	0	0	1	0
UW Hip	HFEO-SUP	83	80	73	0	0	0	0	0	0	0	1
	KEFO-SIT	87	81	67	0	0	0	0	0	0	1	0
	KHEF-SUP	92	55	74	0	0	0	0	0	0	1	0
	All	87	70	70	0	0	0	0	0	0	0	1
URFI	SQUA-STD	86	90	86	0	0	0	0	0	0	1	0

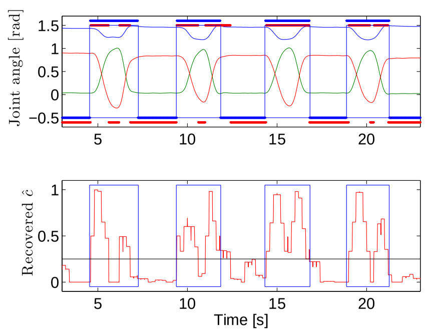
Table 7.6: Segmentation accuracy of UW Ankle, UW Hip, and URFI datasets when the threshold and basis function is selected separately for each set of observation (when the exercise is denoted as the primitive), as well as the average accuracy and sum of the basis functions selected (when 'Sum' is denoted as the primitive). The basis weights columns report the number of times that particular basis weight was selected for the maximum segmentation accuracy.

Data		**Individual Accuracy**			**Basis Function Selected**							
Dataset	**Primitive**	$Acc_{precision}$	Acc_{recall}	Acc_{bal}	J_{dddq}	J_{dddx}	J_{ddtau}	J_{ddq}	J_{power}	J_{dtau}	J_{en}	J_{tau}
UW Ankle	SQUA-STD	98	77	83	0	0	0	0	2	0	30	9
	STSO-SIT	94	87	77	0	0	27	0	11	0	2	1
	Sum	96	83	79	0	0	27	0	13	0	32	10
UW Hip	HFEO-SUP	97	70	77	1	0	0	0	8	0	4	25
	KEFO-SIT	97	65	75	2	0	1	0	5	0	16	17
	KHEF-SUP	97	74	80	0	0	0	0	2	0	15	18
	Sum	97	70	77	3	0	1	0	15	0	35	60
URFI	SQUA-STD	92	87	88	0	0	0	0	0	0	15	0

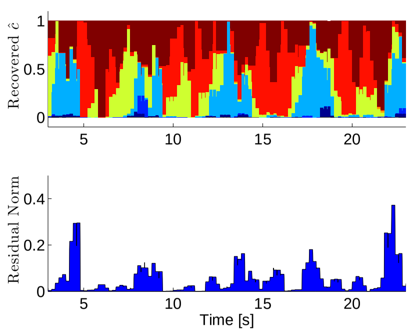




(b) Motion trajectory and recovered weights.

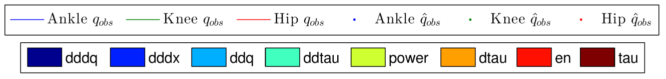


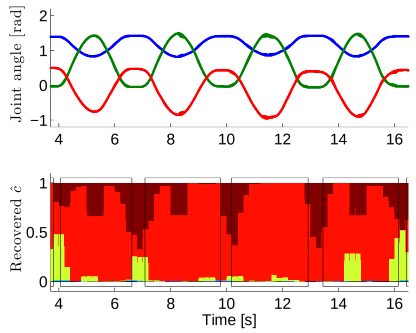
(c) Segmentation results using J_{en} as threshold, set at 0.25. The ground truth (b) denotes motion (y=1.5) and resting (y=-0.5) with algorithmic (r) segment points.



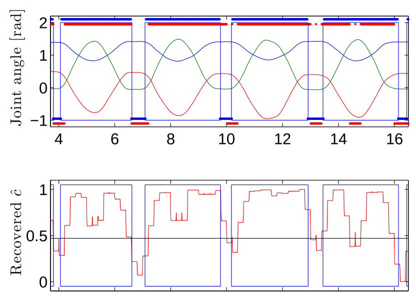
(d) Residual norm.

Figure 7.7: Human squat data showing the recovered weights (left), segmentation results (middle) and residual norm (right) from the URFI dataset.

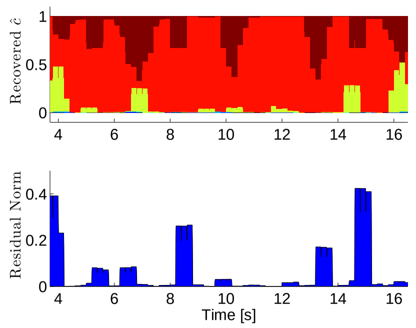




(b) Motion trajectory and recovered weights.

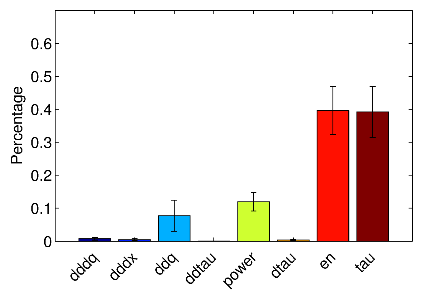


(c) Segmentation results using J_{en} as threshold, set at 0.47. The ground truth (b) denotes motion (y=1.5) and resting (y=-0.5) with algorithmic (r) segment points.

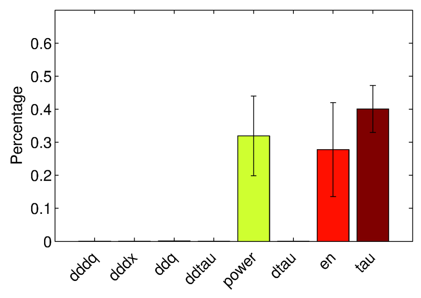


(d) Residual norm.

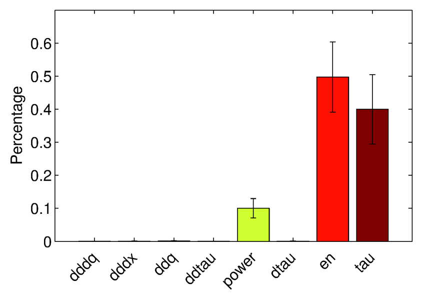
Figure 7.8: Human squat data showing the recovered weights (left), segmentation results (middle) and residual norm (right) from the UW dataset.



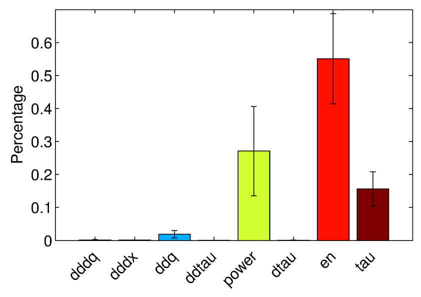
(a) URFI squats.



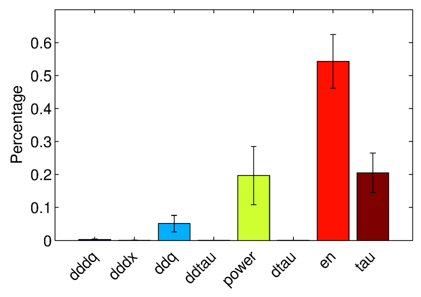
(b) UW hip flexion.



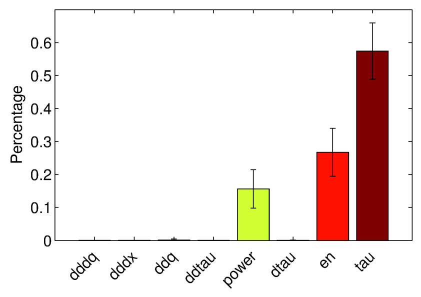
(c) UW knee/hip flexion.



(d) UW squats.



(e) UW sit to stand.



(f) UW knee extension.

Figure 7.9: Percentage of $\hat{\mathbf{c}}$ over the all subjects in the datasets examined. These plots were generated by calculating the mean $\hat{\mathbf{c}}$ over the full trajectory of each subject, resulting in 8 sets of $\hat{\mathbf{c}}$. The mean and standard deviation in this graph shows the mean and variation between participants.

The residual norm threshold testing (Table 7.4) shows that the URFI squats obtain low RMSE and residual norm. However, low residual norm does not necessarily imply that the recovered trajectory would result in low RMSE. It is difficult to compare two different datasets in residual norm as they are a function of the normalization specific to that dataset, but the RMSE values are in the same units of radians. For the two subsets of the UW dataset, the hip-first motions perform much worse in the RMSE, and result in paths that were not similar to the original. This emphasizes that the $\hat{\mathbf{c}}$ extracted by IOC does not necessarily result in a good fit in all DOFs, because we have assumed that the same cost function is being used for each joint. From the results, it can be noted that even though the ankle trajectory for the hip-first motion (Figure 7.6, red) fits poorly with the original, the other 2 DOFs fit fairly well. This suggests that the ankle joint may not be using the same cost function as the hip and knee.

In many instances, a high standard deviation in the residual norm compared to the mean is observed, denoting that large regions have low residual norm, but spikes of high residual norm are occurring (Figures 7.7 and 7.8). This suggests that there are regions that are not properly modelled by the basis functions available, leading to poor performance.

A closer examination into the reconstructed weights reveal regular patterns (Figure 7.6) between movement and rest. During movement, kinetic energy (red), torque (dark red), and power (yellow) are highly weighted, while resting periods between repetitions are dominated by ddq (blue). This shift suggests a change in the cost function during different parts of the motion.

The segmentation results show high accuracy for the individual primitives (Table 7.6), suggesting that basis weights can be used as a segmentation feature. The lowest score in Table 7.6 is Acc_{recall} of 65%, suggesting that false negative non-movements are a larger problem than the false positives. However, the segmentation results were generated by individually determining the basis weight and threshold that resulted in the highest segmentation accuracy for each observed trajectory, and averaging them together. The basis weights selected vary between individuals, suggesting that a single basis weight may not serve as the most optimal for the whole dataset. This observation is emphasized by examining Table 7.5, where a single threshold was selected for the denoted exercise type. This table shows a lower Acc_{bal} score when compared to individual results, and also shows that Acc_{recall} suffers lower accuracy when compared to $Acc_{precision}$. The average $\hat{\mathbf{c}}$ for each primitive (Figure 7.9) show that ddq, power, kinetic energy, and tau are significant basis functions, but the distribution can vary from primitive to primitive.

A large difference is observed between the UW dataset and the URFI dataset as well. This may be due to the fact that the UW dataset contains motions where the participant did not pause in between repetitions, while the URFI dataset contained long pauses. The UW dataset also had larger variety in joint angle range and velocity, while URFI motions were more controlled. These major differences led to a wider variety of basis functions and thresholds selected (Table 7.6).

The runtime for this experiment approximated 5.7 seconds for every second of data used in the IOC, suggesting that further computational optimization is required for online operation.

7.5 Conclusion

This chapter proposes a method for human motion segmentation based on inverse optimal control. The approach accepts arbitrary length trajectories and estimates the underlying basis function weights for successive windows of that trajectory using inverse optimal control. A method to reject low-quality weight estimates by examining the residual norm is proposed, and the algorithm is demonstrated in both simulation and with real data, achieving a segmentation accuracy of 77-88%.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

Motion segmentation, the process of identifying the starting and ending locations of movements of interest, is a key enabling technology in a number of different fields, including automated rehabilitation, imitation learning, and gesture recognition. The focus of this thesis is to develop accurate time-series segmentation for physiotherapy and rehabilitation to reduce dependency on subjective measures and visual observation for diagnosis, assessment, and progress monitoring.

For a segmentation algorithm to be suitable for clinical applications, it should be able to utilize a wide range of sensor modalities, provide consistent and accurate segments, operate online, and exhibit inter-primitive and inter-participant generalizability.

While many prior works have proposed methods to solve the problem of motion segmentation, key problems such as the lack of development and validation against inter-participant and inter-primitive generalizability, difficulty comparing existing segmentation techniques, and dependency on single sensor modalities has motivated the research detailed in this thesis.

The contributions of this thesis are as follows:

A framework for segmentation that can be used for algorithm development and comparison.

A comprehensive literature review on time-series segmentation was carried out, the first in the literature focusing specifically on segmentation. This review was used to build a segmentation framework that enables systematic comparisons of different segmentation algorithms and serves as a guide for the development of new segmentation algorithms. This review showed that a majority of the segmentation algorithms did not report segmentation accuracy or were tested against small datasets, and that the lack of large publicly available datasets make the comparison of segmentation algorithms difficult. Inter-participant and inter-primitive generalizability were frequently cited as a difficult for segmentation, but rarely tested.

The framework illustrates that, to develop a new segmentation algorithm, the user must first determine the segment definition, which can vary from application to application. Then, the data source, such as sensor modality to be used, as well as the source and target populations, including any public datasets, should be considered. Next to consider are any algorithmic constraints, such as online or semi-online time requirements, or any inter-participant and inter-primitive generalizability. Once these considerations have been made, the segmentation algorithm can be designed and verified against the target population. The framework provides a survey of relevant methods, organized by online, semi-online, offline supervised, and offline unsupervised approaches, allowing the user to draw motivation from the techniques that suit their requirements.

The framework also illustrates that, to compare existing techniques, the user must be aware of the validation dataset used, which may be too small or does not properly address the target population. The validation metric and cross-validation technique employed may also emphasize or obscure the strengths and weaknesses of the segmentation algorithm. Greater availability of public datasets is needed to enable direct comparison between algorithms.

A motion segmentation algorithm that meets the requirements of a clinically-focused method.

A classifier-based approach was developed to address the time-series segmentation problem. The proposed approach classifies all data points as either a segment edge point or a non-segment point, using machine learning to automatically extract common segment edge characteristics, which leads to better inter-primitive generalizability. This is critical for rehabilitation applications as prescribed motions tend to be personalized and differ slightly from patient to patient, so by reducing the need for individualized templates, this segmentation algorithm can be deployed with the type of performance that is required for clinical applications. The proposed approach was tested with both healthy and rehabilitation data, and demonstrated generalization to novel motions and novel participants. In particular, templates generated from healthy data successfully segmented rehabilitation data.

Systematic validation with multiple sensor modalities.

A majority of the developed segmentation algorithms are only tested with a single sensor modality. The ability to use a variety of sensor and feature modalities is necessary for a clinical-based algorithm as the sensors available may change with the deployment environment. The proposed algorithm does not require specific hand crafted features, improving its usefulness in the clinic. The proposed method was successfully validated with IMU, EMG, motion capture, and force plate data, using joint angle, joint position, EMG signal, ground reaction force, centre of pressure, and joint torque features. Segmentation success using joint position and ground reaction force features is particularly promising, as these features can be extracted directly from the sensor, and do not require expensive algorithms such as inverse kinematics or dynamics, thus improving the speed that a segmentation algorithm can be deployed.

Inverse optimal control for motion segmentation.

A novel feature set motivated by inverse optimal control was created and tested for segmentation. Hypothesizing that human motion is generated by an optimal controller, the cost function of this controller is modelled as a sum of weighted basis functions representing features such as joint acceleration, linear acceleration, or torque, and the weights were recovered using KKT. Poorly recovered weights were removed based on the residual norm, and rejected based on a simulation-derived threshold. The recovery of basis weights was validated in simulation and was shown to be time-varying in real data. This signal was then segmented by a simple threshold and showed promising performance.

The proposed segmentation framework identifies a number of open research questions:

Segment definition.

Numerous algorithms have been proposed for all three segment types, consisting of physical boundaries, derived metrics boundaries, and template boundaries. Beyond this thesis, no other techniques have examined the impact on segmentation accuracy of varying the segment definition, such as testing both the half segment and the full segment. The segment definition can significantly affect the segmentation accuracy but has not been studied extensively.

Data collection.

The lack of public datasets with manual segments also hamper the ability to compare motion segmentation algorithms directly. In particular, the publication of datasets with varying physical conditions can facilitate segmentation algorithm development and benchmarking significantly. Most public datasets have no manual labels or are no longer accessible. Some datasets have verified activity-level labels [158, 210] that are useful for activity recognition. However, the only dataset that has verified manual labels at the motion repetition-level labels is the University of Tokyo dataset [114].

Application specific requirements.
Inter-participant and inter-primitive generalizability are not commonly considered in current research. This restricts many algorithms from being readily deployed in clinical settings. Inter-participant variability can be assessed by validating in a LOPO cross-validation scheme, which is rarely done. Although algorithms that utilize domain knowledge features are likely capable of some degree of inter-primitive generalizability, this is rarely assessed in published work.

Verification.
Many segmentation algorithms are developed for a specific application and are not explicitly validated. Of the validated techniques, reported segmentation accuracy and experimental details are often sparse. This makes it difficult to compare reported techniques and to recommend suitable methods, despite the numerous prior works in the literature.

In the thesis, two different motion segmentation algorithms are proposed, representing two different approaches to the rehabilitation-based motion segmentation problem. The classifier-based approach is a semi-online supervised technique that requires manually segmented exercise data and normalization methods in order to achieve high segmentation accuracy when applied to the target population. The optimal control-based approach is an offline unsupervised technique that does not require any training data, thus reducing the overhead time required to set up the algorithm. However, the optimal control based approach requires domain knowledge, specifically the hypothesized optimality criteria used during movement. In most applications, segmentation accuracy and online response is important for clinical deployment, and thus the classifier-based approach is the most suitable approach to utilize. However, there may be settings where the target primitives to be recognized are very different compared to the source primitives and manual segment data are not readily available. In these cases, the optimal control-based may be appropriate. The optimal control method can also be helpful for understanding the causes of motion variability observed.

8.2 Future Work

The open research questions that have been identified by the proposed segmentation framework suggest numerous potential directions for future work. The impact of varying segment definitions should be examined in depth.

Public datasets with manual segments at the repetition-level enable algorithms to be directly compared and should be made available. Datasets with varying physical conditions should also be created so segmentation techniques can be tested on populations other than nominally healthy participants. Algorithm validation on inter-participant and inter-primitive variability is of critical significance and should be a component of all segmentation methods.

For the classifier-based approach, more complex full-body exercises and functional tasks should be examined.

Further exploration into additional sensor modalities and clinical applications is recommended. The results in this thesis (Section 6) show that dynamic data may require different sets of manual segment data and treatment from the kinematic data, but little research have been done in this space. Investigation into the types of modalities that produce segments that best suit clinical applications is needed, as well as efforts to understand the difference between the segments created by kinematic features compared to dynamic features.

Additional feature sets such as the basis weights of the IOC approach are also of interest, as the classifier-based approach may be able to outperform the simple threshold-based approach proposed in Section 7.

Segmentation based on IOC, while promising, requires further investigation. The algorithm does not currently operate in real-time, and thus is not yet feasible for online applications. Converting from numerically calculated gradient to an analytical gradient may reduce computation time. Reducing the tolerance for the optimization may also speed up convergence.

The proposed IOC method does not incorporate dynamic constraints or joint limit constraints, potentially leading to unrealistic estimates of human capabilities. The incorporation of these constrains would allow the IOC method to more realistically model the underlying human movement.

The IOC method is also dependent on the basis functions modelled, so if a critical function is not modelled, the residual norm will be high. A possible complication is that the basis functions examined in this thesis were originally motivated for arm and upper body movements [22], as a majority of the prior work in IOC has been applied to upper body tasks [213]. Conversely, basis functions can also suffer from dependency issues, leading to non-unique solutions. The residual norm threshold is constructed using simulation data, where the basis functions used are perfectly known. However, the basis functions used in human motion data are generally not known *a priori*. Therefore, additional basis functions, such as basis functions that are dependent on individual DOFs instead of all available DOFs, should be incorporated and tested, to see the impact of differing sets of basis functions on the residual norm. Developing methods to reject dependent basis functions, such as using SVD to examine the rank of the $\mathbf{A_0}$ matrix, will also improve the quality of the IOC method.

The appropriate selection of the window length also requires further investigation. Windows that are too short may not contain enough trajectory information to result in unique basis weight recovery. However, windows that are too long may contain multiple motion primitives and lead to poor basis weight recovery. Dynamic window length selection should be investigated.

Insight into how the recovered basis weights are connected to human control theory should also be examined. In particular, how do effects such as fatigue, injury, or recovery change the human controller? Collecting and examining datasets that capture fatigue or rehabilitation recovery may improve our understanding of human neuromuscular control in these processes.

References

[1] N. Aghasadeghi and T. Bretl. Inverse optimal control for differentially flat systems with application to locomotion modeling. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 6018–6025, 2014.

[2] V. Agostini, G. Balestra, and M. Knaflitz. Segmentation and classification of gait cycles. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(5):946–952, 2014.

[3] M. A. R. Ahad, J. K. Tan, H. S. Kim, and S. Ishikawa. Human activity recognition: Various paradigms. In *Proceedings of the International Conference on Control, Automation and Systems*, pages 1896–1901, 2008.

[4] E. E. Aksoy, Y. Zhou, Mi. Wächter, and T. Asfour. Enriched manipulation action semantics for robot execution of time constrained tasks. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, 2016. In press.

[5] S. Albrecht, K. Ramírez-Amaro, F. Ruiz-Ugalde, D. Weikersdorfer, M. Leibold, M. Ulbrich, and M. Beetz. Imitating human reaching motions using physically inspired optimization principles. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 602–607, 2011.

[6] R. M. Alexander. The gaits of bipedal and quadrupedal animals. *The International Journal of Robotics Research*, 3(2):49–59, 1984.

[7] O. Amft, H. Junker, and G. Tröster. Detection of eating and drinking arm gestures using inertial body-worn sensors. In *Proceedings of the IEEE International Symposium on Wearable Computers*, pages 160–163, 2005.

[8] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*, volume 2. Wiley New York, 1958.

[9] T. Aoki, G. Venture, and D. Kulić. Segmentation of human body movement using inertial measurement unit. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pages 1181–1186, 2013.

[10] A. Aristidou, J. Cameron, and J. Lasenby. Real-time estimation of missing markers in human motion capture. In *Proceedings of the International Conference on Bioinformatics and Biomedical Engineering*, pages 1343–1346, 2008.

[11] T. Asfour and R. Dillmann. Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1407–1412, 2003.

[12] A. Ataya, P. Jallon, P. Bianchi, and M. Doron. Improving activity recognition using temporal coherence. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4215–4218, 2013.

[13] K. Ayusawa, G. Venture, and Y. Nakamura. Identification of humanoid robots dynamics using floating-base motion dynamics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2854–2859, 2008.

[14] K. Ayusawa, G. Venture, and Y. Nakamura. Real-time implementation of physically consistent identification of human body segments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 6282–6287, 2011.

[15] S. Baby and V. Krüger. Primitive based action representation and recognition. In *Image Analysis*, volume 5575 of *Lecture Notes in Computer Science*, pages 31–40. Springer Berlin/Heidelberg, 2009.

[16] S. Baby, V. Krüger, and D. Kragic. Unsupervised learning of action primitives. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 554–559, 2010.

[17] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[18] J. Barbič, A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard. Segmenting motion capture data into distinct behaviors. In *Proceedings of Graphics Interface*, pages 185–194, 2004.

[19] F. Bashir, W. Qu, A. Khokhar, and D. Schonfeld. HMM-based motion recognition system using segmented PCA. In *Proceedings of the IEEE International Conference on Image Processings*, pages 1288–1291, 2005.

[20] E. Berlin and K. Van Laerhoven. Detecting leisure activities with dense motif discovery. In *Proceedings of the ACM Conference on Ubiquitous Computing*, pages 250–259, 2012.

[21] K. Bernardin, K. Ogawara, K. Ikeuchi, and R. Dillmann. A sensor fusion approach for recognizing continuous human grasping sequences using hidden markov models. *IEEE Transactions on Robotics*, 21:47–57, 2005.

[22] B. Berret, E. Chiovetto, F. Nori, and T. Pozzo. Evidence for composite cost functions in arm movement planning: An inverse optimal control approach. *PLoS Computational Biology*, 2011.

[23] Bertec Corporation. Bertec force plates. www.bertec.com, 1987.

[24] J. T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998.

[25] C. M. Bishop. *Pattern Recognition and Machine Learning, 2nd Ed.* Springer, 2010.

[26] V. Bonnet, C. Mazza, P. Fraisse, and A. Cappozzo. A least-squares identification algorithm for estimating squat exercise mechanics using a single inertial measurement unit. *Journal of Biomechanics*, 45(8):1472–1477, 2012.

[27] M. C. Boonstra, R. M. A. van der Slikke, N. L. W. Keijsers, R. C. van Lummel, M. C. de Waal Malefijt, and N. Verdonschot. The accuracy of measuring the kinematics of rising from a chair with accelerometers and gyroscopes. *Journal of Biomechanics*, 39:354–358, 2006.

[28] J. Borrás and T. Asfour. A whole-body pose taxonomy for loco-manipulation tasks. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1578–1585, 2015.

[29] J. Boyd and H. Sundaram. A framework to detect and classify activity transitions in low-power applications. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, pages 1712–1715, 2009.

[30] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[31] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[32] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[33] P. Brossier, J. P. Bello, and M. D. Plumbley. Real-time temporal segmentation of note objects in music signals. In *Proceedings of the International Computer Music Association*, 2004.

[34] I. M. Bullock, T. Feix, and A. M. Dollar. The Yale human grasping dataset: Grasp, object, and task data in household and machine shop environments. *International Journal of Robotics Research*, 2014.

[35] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[36] A. Burns, B. R. Greene, M. J. McGrath, T. J. O'Shea, B. Kuris, S. M. Ayer, F. Stroiescu, and V. Cionca. SHIMMER: A wireless sensor platform for noninvasive biomedical research. *IEEE Sensors Journal*, 10:1527–1534, 2010.

[37] T. Buschmann, S. Lohmeier, M. Bachmayer, H. Ulbrich, and F. Pfeiffer. A collocation method for real-time walking pattern generation. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 1–6, 2007.

[38] M. Buss, M. Hardt, J. Kiener, M. Sobotka, M. Stelzer, O. von Stryk, and D. Wollherr. Towards an autonomous, humanoid, and dynamically walking robot: Modeling, optimal trajectory planning, hardware architecture, and experiments. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 2491–2496, 2003.

[39] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan. IEMOCAP: interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42(4):335, 2008.

[40] R. H. Byrd, J. C. Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89(1):149–185, 2000.

[41] S. Calinon and A. Billard. Stochastic gesture production and recognition model for a humanoid robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2769–2774, 2004.

[42] S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. G. Billard. Learning and reproduction of gestures by imitation. *IEEE Robotics and Automation Magazine*, 17:44–54, 2010.

[43] Carnegie Mellon University. CMU graphics lab motion capture database. mocap.cs.cmu.edu, 2010.

[44] F. Carrino, A. Ridi, E. Mugellini, O. Khaled, and R. Ingold. Gesture segmentation and recognition with an EMG-based intimate approach - an accuracy and usability study. In *Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems*, pages 544–551, 2012.

[45] A. Castellani, D. Botturi, M. Bicego, and P. Fiorini. Hybrid HMM/SVM model for the analysis and segmentation of teleoperation tasks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 2918–2923, 2004.

[46] J. Cham. Piled higher and deeper (PHD). www.phdcomics.com, 1997.

[47] F. Chamroukhi, S. Mohammed, D. Trabelsi, L. Oukhellou, and Y. Amirat. Joint segmentation of multivariate time series with hidden process regression for human activity recognition. *Neurocomputing*, 120:633–644, 2013.

[48] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[49] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero. A survey of video datasets for human action and activity recognition. *Computer Vision and Image Understanding*, 117:633–659, 2013.

[50] Z. Chen, X. Wang. Pattern recognition of number gestures based on a wireless surface EMG system. *Biomedical Signal Processing and Control*, 8:184–92, 2013.

[51] S. Chiappa and J. Peters. Movement extraction by detecting dynamics switches and repetitions. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 388–396, 2010.

[52] CIR Systems Inc. Gaitrite. www.gaitrite.com, 2017.

[53] D. Clever, R. M. Schemschat, M. L. Felis, and K. Mombaur. Inverse optimal control based identification of optimality criteria in whole-body human walking on level ground. In *Proceedings of IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 1192–1199, 2016.

[54] E. Costanza, S. Inverso, and R. Allen. Toward subtle intimate interfaces for mobile devices using an EMG controller. In *SIGCHI*, pages 481–9, 2005.

[55] M. Davidson and J. L. Keating. A comparison of five low back disability questionnaires: Reliability and responsiveness. *Physical Therapy*, 82(1):8–24, 2002.

[56] D. T. Davy and M. L. Audu. A dynamic optimization technique for predicting muscle forces in the swing phase of gait. *Journal of Biomechanics*, 20(2):187–201, 1987.

[57] F. De la Torre, J. Hodgins, A. Bargteil, X. Martin, J. Macey, A. Collado, and P. Beltran. Guide to the carnegie mellon university multimodal activity (CMU-MMAC) database. Technical report, Carnegie Mellon University Robotics Institute, 2009.

[58] P. Delacourt and C. J. Wellekens. DISTBIC: A speaker-based segmentation for audio data indexing. *Speech Communication*, 32:111–126, 2000. Accessing Information in Spoken Audio.

[59] J. A. DeLisa, B. M. Gans, N. E. Walsh, and W. L. Bockenek. *Physical Medicine and Rehabilitation: Principles and Practice, 4th edition*. Lippincott Williams and Wilkins, 2005.

[60] R. Dumas, L. Chéze, and J.-P. Verriest. Adjustments to McConville et al. and Young et al. body segment inertial parameters. *Journal of Biomechanics*, 40:543–553, 2007.

[61] S. Ekvall, D. Aarno, and D. Kragic. Online task recognition and real-time adaptive assistance for computer-aided machine control. *IEEE Transactions on Robotics*, 22:1029–1033, 2006.

[62] W. El Falou, J. Duchêne, D. Hewson, M. Khalil, M. Grabisch, and F. Lino. A segmentation approach to long duration surface EMG recordings. *Journal of Electromyography and Kinesiology*, 15:111–9, 2005.

[63] P. Englert and M. Toussaint. Inverse KKT – Learning Cost Functions of Manipulation Tasks from Demonstrations. In *Proceedings of the International Symposium of Robotics Research*, 2015.

[64] J. R. Flanagan and A. K. Rao. Trajectory adaptation to a nonlinear visuomotor transformation: Evidence of motion planning in visually perceived space. *Journal of Neurophysiology*, 74(5):2174–2178, 1995.

[65] T. Flash and N. Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of Neuroscience*, 5(7):1688–1703, 1985.

[66] A. Fod, M. J. Matarić, and O. C. Jenkins. Automated derivation of primitives for movement classification. *Autonomous Robots*, 12:39–54, 2002.

[67] I. K. Fodor. A survey of dimension reduction techniques. Technical report, Lawrence Livermore National Laboratory, 2002.

[68] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin. Instructing people for training gestural interactive systems. In *ACM Conference on Human Factors in Computing Systems*, pages 1737–1746, 2012.

[69] E. B. Fox, M. C. Hughes, E. B. Sudderth, and M. I. Jordan. Joint modeling of multiple time series via the beta process with application to motion capture segmentation. *The Annals of Applied Statistics*, 8:1281–1313, 2014.

[70] Y. Freund and R. E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14:771–780, 1999.

[71] A. Ganapathiraju, J. E. Hamaker, and J. Picone. Applications of support vector machines to speech recognition. *IEEE Transactions on Signal Processing*, 52:2348–2355, 2004.

[72] W. J. Gibbons, N. Fruchter, S. Sloan, and R. D. Levy. Reference values for a multiple repetition 6-minute walk test in healthy adults older than 20 years. *Journal of Cardiopulmonary Rehabilitation*, 21:87–93, 2001.

[73] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization*. Academic Press, Inc, 1981.

[74] G. Guerra-Filho and Y. Aloimonos. A language for human action. *Computer*, 40:42–51, 2007.

[75] Y. Hao, Y. Chen, J. Zakaria, B. Hu, T. Rakthanmanon, and E. Keogh. Towards never-ending learning from time series streams. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 874–882, 2013.

[76] Z.-Y. He and L.-W. Jin. Activity recognition from acceleration data using ar model representation and svm. In *International Conference on Machine Learning and Cybernetics*, volume 4, pages 2245–2250, 2008.

[77] L. Herda, P. Fua, R. Plänkers, R. Boulic, and D. Thalmann. Using skeleton-based tracking to increase the reliability of optical motion capture. *Human Movement Science*, 20:313–341, 2001.

[78] P. Hong, M. Turk, and T. S. Huang. Gesture modeling and recognition using finite state machines. In *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 410–415, 2000.

[79] R. Houmanfar, M. E. Karg, and D. Kulić. Movement analysis of rehabilitation exercises: Distance metrics for measuring patient progress. *IEEE Systems Journal*, 10(3):1014–1025, 2016.

[80] J. A. Howe, E. L. Inness, A. Venturini, J. I. Williams, and M. C. Verrier. The community balance and mobility scale – a balance measure for individuals with traumatic brain injury. *Clinical Rehabilitation*, 20:885–895, 2006.

[81] A. J. Ijspeert, J. Nakanishi, and S. Schaal. Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1398–1403, 2002.

[82] W. Ilg, G. H. Bakir, J. Mezger, and M. A. Giese. On the representation, learning and transfer of spatio-temporal movement characteristics. *International Journal of Humanoid Robotics*, 1:613–636, 2004.

[83] M. Ito and J. Tani. On-line imitative interaction with a humanoid robot using a dynamic neural network model of a mirror system. *Adaptive Behavior*, 12(2):93–115, 2004.

[84] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:852–872, 2000.

[85] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:4–37, 2000.

[86] A. K. Jain, J. Mao, and K. M. Mohiuddin. Artificial neural networks: A tutorial. *Computer*, 29(3):31–44, 1996.

[87] B. Janus and Y. Nakamura. Unsupervised probabilistic segmentation of motion data for mimesis modeling. In *Proceedings of the IEEE International Conference on Advanced Robotics*, pages 411–417, 2005.

[88] O. C. Jenkins and M. Mataric. Deriving action and behavior primitives from human motion data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2551–2556, 2002.

[89] O. C. Jenkins and M. J. Mataric. Automated derivation of behavior vocabularies for autonomous humanoid motion. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 225–232, 2003.

[90] V. Joukov, M. E. Karg, and D. Kulić. Online tracking of the lower body joint angles using IMUs for gait rehabilitation. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 2310–2313, 2014.

[91] M. P. Kadaba, H. K. Ramakrishnan, and M. E. Wootten. Measurement of lower extremity kinematics during level walking. *Journal of Orthopaedic Research*, 8(3):383–392, 1990.

[92] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4569–4574, 2011.

[93] S. B. Kang and K. Ikeuchi. Toward automatic robot instruction from perception-temporal segmentation of tasks from human hand motion. *IEEE Transactions on Robotics and Automation*, 11:670–681, 1995.

[94] G. Kaur, A. Arora, and V. Jain. Comparison of the techniques used for segmentation of EMG signals. In *Proceedings of the WSEAS International Conference on Mathematical and Computational Methods in Science and Engineering*, pages 124–129, 2009.

[95] M. P. Kelly. Transcription methods for trajectory optimization. Technical report, Cornell University, 2015.

[96] E. J. Keogh, S. Chu, D. Hart, and M. Pazzani. *Data Mining in Time Series Databases*, volume 57, chapter Segmenting Time Series: A Survey and Novel Approach, pages 1–22. World Scientific Publishing, 2004.

[97] E. J. Keogh and M. J. Pazzani. Derivative dynamic time warping. In *Proceedings of the SIAM International Conference on Data Mining*, pages 1–11, 2001.

[98] A. Keshavarz, Y. Wang, and S. Boyd. Imputing a convex objective function. In *Proceedings of the IEEE International Symposium on Intelligent Control*, pages 613–619, 2011.

[99] W. Khalil and D. Creusot. Symoro+: A system for the symbolic modelling of robots. *Robotica*, 15(2):153–161, 1997.

[100] J. Kim, S. Mastnik, and E. André. Emg-based hand gesture recognition for realtime biosignal interfacing. In *Proceedings of the International Conference on Intelligent User Interfaces*, pages 30–39, 2008.

[101] D. E. Kirk. *Optimal Control Theory: An Introduction*. Dover Publications, 2004.

[102] C. Kisner and L. A. Colby. *Therapeutic Exercise: Foundations and Techniques*. F. A. Davis Company, 2007.

[103] K. H. Koch, K. Mombaur, and P. Soueres. Optimization-based walking generation for humanoid robot. *IFAC Proceedings Volumes*, 45(22):498–504, 2012.

[104] N. Koenig and M. J. Matarić. Behaviour-based segmentation of demonstrated tasks. In *Proceedings of the International Conference on Development and Learning*, 2006.

[105] J. Kohlmorgen and S. Lemm. A dynamic hmm for on-line segmentation of sequential data. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 14, pages 793–800, 2002.

[106] P. Konrad. The ABC of EMG. Technical report, Noraxon USA, 2005.

[107] R. J. Kosinski. A literature review on reaction time. Technical report, Clemson University, 2008.

[108] S. B. Kotsiantis. Supervised machine learning: A review of classification techniques. In *Proceedings of the Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pages 3–24, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press.

[109] D. Kragic, P. Marayong, M. Li, A. M. Okamura, and G. D. Hager. Human-machine collaborative systems for microsurgical applications. *International Journal of Robotics Research*, 24:731–741, 2005.

[110] N. C. Krishnan, D. Colbry, C. Juillard, and S. Panchanathan. Realtime human activity recognition using tri-axial accelerometers. In *Sensors, Signals and Information Processing Workshop*, 2008.

[111] S. Krishnan, A. Garg, S. Patil, C. Lea, G. Hager, P. Abbeel, and K. Goldberg. Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning. In *Proceedings of the International Symposium of Robotics Research*, 2015.

[112] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A large video database for human motion recognition. In *International Conference on Computer Vision*, pages 2556–2563, 2011.

[113] D. Kulić and Y. Nakamura. Incremental learning of human behaviors using hierarchical hidden Markov models. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4649–4655, 2010.

[114] D. Kulić, W. Takano, and Y. Nakamura. Online segmentation and clustering from continuous observation of whole body motions. *IEEE Transactions on Robotics*, 25:1158–1166, 2009.

[115] D. Kulić, G. Venture, K. Yamane, E. Demircan, I. Mizuuchi, and K. Mombaur. Anthropomorphic movement analysis and synthesis: A survey of methods and applications. *IEEE Transactions on Robotics*, 32(4):776–795, 2016.

[116] A. W. K. Lam, D. Verona-Marin, Y. Li, M. Fergenbaum, and D. Kulić. Automated rehabilitation system: Movement measurement and feedback for patients and physiotherapists in the rehabilitation clinic. *Human Computer Interaction*, 31:294 – 334, 2016.

[117] R. Lan and H. Sun. Automated human motion segmentation via motion regularities. *The Visual Computer*, 31:35–53, 2015.

[118] O. D. Lara and M. A. Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys Tutorials*, 15:1192–1209, 2013.

[119] O. D. Lara, A. J. Pérez, M. A. Labrador, and J. D. Posada. Centinela: A human activity recognition system based on acceleration and vital sign data. *Pervasive and Mobile Computing*, 8:717–729, 2012.

[120] O. Lartillot and P. Toiviainen. A matlab toolbox for musical feature extraction from audio. In *International Conference on Digital Audio Effects*, pages 237–244, 2007.

[121] A. Laudanski, S. Yang, and Q. Li. A concurrent comparison of inertia sensor-based walking speed estimation methods. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3484–3487, Aug 2011.

[122] H.-K. Lee and J. H. Kim. An hmm-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:961–973, 1999.

[123] S. H. Lee, I. H. Suh, S. Calinon, and R. Johansson. Learning basis skills by autonomous segmentation of humanoid motion trajectories. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, 2012.

[124] M. Li and A. M. Okamura. Recognition of operator motions for real-time assistance using virtual fixtures. In *Proceedings of the Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 125–131, 2003.

[125] Z. Li, Z. Wei, W. Jai, and M. Sun. Daily life event segmentation for lifestyle evaluation based on multi-sensor data recorded by a wearable device. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 2858–2861, 2013.

[126] J. Lieberman and C. Breazeal. Improvements on action parsing and action interpolation for learning through demonstration. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 342–365, 2004.

[127] J. F.-S. Lin, V. Bonnet, V. Joukov, G. Venture, and D. Kulić. Comparison of kinematic and dynamic sensor modalities and derived features for human motion segmentation. In *Proceedings of the IEEE/NIH Strategic Conference on Healthcare Innovations and Point of Care Technologies*, pages 109–112, 2016.

[128] J. F.-S. Lin, V. Bonnet, A. M. Panchea, N. Ramdani, G. Venture, and D. Kulić. Human motion segmentation using cost weights recovered from inverse optimal control. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 1107–1113, 2016.

[129] J. F.-S. Lin, V. Joukov, and D. Kulić. Full-body multi-primitive segmentation using classifiers. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 874–880, 2014.

[130] J. F.-S. Lin, V. Joukov, and D. Kulić. Human motion segmentation by data point classification. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 9–13, 2014.

[131] J. F.-S. Lin, V. Joukov, and D. Kulić. Classification-based segmentation for rehabilitation exercise monitoring. *Journal of Rehabilitation and Assistive Technologies Engineering*, 2017. in review.

[132] J. F.-S. Lin, M. E. Karg, and D. Kulić. Movement primitive segmentation for human motion modeling: A framework for analysis. *IEEE Transactions on Human-Machine Systems*, 46:325–339, 2016.

[133] J. F.-S. Lin and D. Kulić. Human pose recovery using wireless inertial measurement units. *Physiological Measurement*, 33:2099–2115, 2012.

[134] J. F.-S. Lin and D. Kulić. On-line segmentation of human motion for automated rehabilitation exercise analysis. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22:168–180, 2014.

[135] J. F.-S. Lin, A. Samadani, and D. Kulić. Segmentation by data point classification applied to forearm surface emg. In *Proceedings of the EAI International Summit of Smart City 360*, volume 166, pages 153–165. Springer International Publishing, 2016.

[136] Z. Liu, Y. Wang, and T. Chen. Audio feature extraction and analysis for scene segmentation and classification. *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, 20:61–79, 1998.

[137] C. Lu and N. J. Ferrier. Repetitive motion analysis: Segmentation and event classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:258–263, 2004.

[138] L. Lu, H. J. Zhang, and H. Jiang. Content analysis for audio classification and segmentation. *IEEE Transactions on Speech and Audio Processing*, 10:504–516, 2002.

[139] H. Luinge and P. Veltink. Measuring orientation of human body segments using miniature gyroscopes and accelerometers. *Medical and Biological Engineering and Computing*, 43:273–282, 2005.

[140] F. Lv and R. Nevatia. Recognition and segmentation of 3-D human action using HMM and multi-class AdaBoost. In *Proceedings of the European Conference on Computer Vision*, pages 359–372, 2006.

[141] Y. Ma, H. M. Paterson, and F. E. Pollick. A motion capture library for the study of identity, gender, and emotion perception from biological motion. *Behavior Research Methods*, 38(1):134–141, 2006.

[142] C. Mandery, J. Borrás, M. Jöchner, and T. Asfour. Analyzing whole-body pose transitions in multi-contact motions. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 1020–1027, 2015.

[143] C. Mandery, J. Borrás, M. Jöchner, and T. Asfour. Using language models to generate whole-body multi-contact motions. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5411–5418, Oct 2016.

[144] C. Mandery, O. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour. The kit whole-body human motion database. In *Proceedings of the IEEE International Conference on Advanced Robotics*, pages 329–336, 2015.

[145] C. Mandery, Ö. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour. Unifying representations and large-scale whole-body motion databases for studying human motion. *IEEE Transactions on Robotics*, 32(4):796–809, 2016.

[146] P. Manns, M. Sreenivasa, M. Millard, and K. Mombaur. Motion optimization and parameter identification for a human and lower back exoskeleton model. *IEEE Robotics and Automation Letters*, 2(3):1564–1570, 2017.

[147] U. Maurer, A. Smailagic, D.P. Siewiorek, and M. Deisher. Activity recognition and monitoring using multiple sensors on different body positions. In *International Workshop on Wearable and Implantable Body Sensor Networks*, pages 113–116, 2006.

[148] Measurand Inc. Shapehand dataglove. www.shapehand.com, 2009.

[149] F. Meier, E. Theodorou, F. Stulp, and S. Schaal. Movement segmentation using a primitive library. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3407–3412, 2011.

[150] F. Meier, E. Theodorou, F. Stulp, and S. Schaal. Movement segmentation and recognition for imitation learning. In *International Conference on Artificial Intelligence and Statistics*, pages 761–769, 2012.

[151] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37:311–324, 2007.

[152] T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104:90–126, 2006.

[153] K. Mombaur, J. P. Laumond, and E. Yoshida. An optimal control model unifying holonomic and nonholonomic walking. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 646–653, 2008.

[154] K. Mombaur, A. Truong, and J.-P. Laumond. From human to humanoid locomotionan inverse optimal control approach. *Autonomous Robots*, 28:369–383, 2010.

[155] K. D. Mombaur, H. G. Bock, J. P. Schloder, and R. W. Longman. Human-like actuated walking that is asymptotically stable without feedback. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 4128–4133, 2001.

[156] Motion Analysis Corporation. Eagle cameras and cortex. www.motionanalysis.com, 1999.

[157] A. Mueen, E. Keogh, Q. Zhu, S. Cash, and B. Westover. Exact discovery of time series motifs. In *SIAM International Conference on Data Mining*, pages 473–484, 2009.

[158] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database HDM05. Technical Report CG-2007-2, Universität Bonn, 2007.

[159] A. Murali, A. Garg, S. Krishnan, F. T. Pokorny, P. Abbeel, T. Darrell, and K. Goldberg. Tsc-dl: Unsupervised trajectory segmentation of multi-modal surgical demonstrations with deep learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4150–4157, 2016.

[160] K. Murphy. Bayes net toolbox for matlab. Software available at `http://code.google.com/p/bnt/`, 1998. Last accessed on 2012/04/27.

[161] K. M. Newell and A. B. Slifkin. *Motor Behavior and Human Skill: A Multidisciplinary Approach*, chapter The Nature of Movement Variability, pages 143–160. Human Kinetics, 1998.

[162] Nintendo Corporation Ltd. Nintendo wii balance board. www.nintendo.com/consumer/downloads/wiiBalanceBoard.pdf, 2007.

[163] C. C. Norkin and D. J. White. *Measurement of Joint Motion: A Guide to Goniometry, 4th Edition.* F. A. Davis Company, 2009.

[164] T. Oberg, L. Sandsjo, and R. Kadefors. EMG mean power frequency: Obtaining a reference value. *Clinical Biomechanics*, 9:253–257, 1994.

[165] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Berkeley MHAD: A comprehensive multimodal human action database. In *IEEE Workshop on Applications of Computer Vision*, pages 53–60, 2013.

[166] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[167] D. Ormoneit, H. Sidenbladh, M. J. Black, and T. Hastie. Learning and tracking cyclic human motion. *Advances in Neural Information Processing Systems*, pages 894–900, 2001.

[168] M. J. L. Orr. Introduction to radial basis function networks. Technical report, Centre for Cognitive Science, University of Edinburgh, 1996.

[169] A. M. Panchea. *Inverse Optimal Control for Redundant Systems of Biological Motion.* PhD thesis, Universite d'Orléans, 2015.

[170] A. M. Panchea and N. Ramdani. Towards solving inverse optimal control in a bounded-error framework. In *American Control Conference*, pages 4910–4915, 2015.

[171] M. G. Pandy. Computer modeling and simulation of human movement. *Annual Review of Biomedical Engineering*, 3:245–273, 2001.

[172] M. G. Pandy, B. A. Garner, and F. C. Anderson. Optimal control of non-ballistic muscular movements: A constraint-based performance criterion for rising from a chair. *Journal of Biomechanical Engineering*, 117(1), 1995.

[173] A. V. Papadopoulos, L. Bascetta, and G. Ferretti. Generation of human walking paths. *Autonomous Robots*, 40(1):59–75, 2016.

[174] N. Peyrard and P. Bouthemy. Content-based video segmentation using statistical motion models. In *British Machine Vision Conference*, pages 1–10, 2002.

[175] A. Phinyomark, F. Quaine, S. Charbonnier, C. Serviere, F. Tarpin-Bernard, and Y. Laurillau. EMG feature evaluation for improving myoelectric pattern recognition robustness. *Expert Systems with Applications*, 40:4832–4840, 2013.

[176] M. Plappert, C. Mandery, and T. Asfour. The KIT motion-language dataset. *Big Data*, 4(4):236–252, dec 2016.

[177] M. Pomplun and M. Matarić. Evaluation metrics and results of human arm movement imitation. In *Proceedings of IEEE/RAS International Conference on Humanoid Robotics*, 2000.

[178] K. L. Priddy and P. E. Keller. *Artificial neural networks: an introduction*, volume 68. SPIE Press, 2005.

[179] A.-S. Puydupin-Jamin, M. Johnson, and T. Bretl. A convex approach to inverse optimal control and its application to modeling human locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 531–536, 2012.

[180] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, 2016(1):67, 2016.

[181] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[182] G. E. Quintana, L. E. Sucar, G. Azcarate, and R. Leder. Qualification of arm gestures using hidden Markov models. In *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 1–6, 2008.

[183] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286, 1989.

[184] C. A. Ratanamahatana and E. Keogh. Making time-series classification more accurate using learned constraints. In *Proceedings of the SIAM International Conference on Data Mining*, pages 11–22, 2004.

[185] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 489–494, 2009.

[186] Z. Razak, K. Zulkiflee, M. Y. I. Idris, E. M. Tamil, M. N. M. Noor, R. Salleh, M. Yaakob, Z. M. Yusof, and M. Yaacob. Off-line handwriting text line segmentation: A review. *International Journal of Computer Science and Network Security*, 8:12–20, 2008.

[187] L. Ricci, D. Formica, E. Tamilia, F. Taffoni, L. Sparaci, O. Capirci, and E. Guglielmelli. An experimental protocol for the definition of upper limb anatomical frames on children using magneto–inertial sensors. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, 2013.

[188] D. G. E. Robertson, G. E. Caldwell, J. Hamill, G. Kamen, and S. N. Whittlesey. *Research Methods in Biomechanics*. Human Kinetics, 2014.

[189] D. Roetenberg, H. Luinge, and P. J. Slycke. Xsens MVN: Full 6dof human motion tracking using miniature inertial sensors. Technical report, Xsens Technologies, 2009.

[190] S. M. M. De Rossi, S. Crea, M. Donati, P. Rebersek, D. Novak, N. Vitiello, T. Lenzi, J. Podobnik, M. Munih, and M. C. Carrozza. Gait segmentation using bipedal foot pressure patterns. In *Proceedings of IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 361–366, 2012.

[191] L. Roussel, C. Canudas-De-Wit, and A. Goswami. Generation of energy optimal complete gait cycles for biped robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 2036–2041, 1998.

[192] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Speech and Signal Processing*, 26:43–49, 1978.

[193] A. Samadani and D. Kulić. Hand gesture recognition based on surface electromyography. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4196–9, 2014.

[194] T. D. Sanger. Human arm movements described by a low-dimensional superposition of principal components. *Journal of Neuroscience*, 20:1066–1072, 2000.

[195] N. I. Sapankevych and R. Sankar. Time series prediction using support vector machines: A survey. *IEEE Computational Intelligence Magazine*, 4(2):24–38, 2009.

[196] R. M. Schemschat, D. Clever, M. Millard, and K. Mombaur. *Model-Based Optimization for the Design of Exoskeletons that Help Humans to Sustain Large Pushes While Walking*, pages 821–825. Springer International Publishing, 2017.

[197] G. Schultz and K. Mombaur. Modeling and optimal control of human-like running. *IEEE/ASME Transactions on Mechatronics*, 15(5):783–792, 2010.

[198] M. A. Schwartz. The importance of stupidity in scientific research. *Journal of Cell Science*, 121(11):1771–1771, 2008.

[199] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, and A. Napolitano. Resampling or reweighting: A comparison of boosting implementations. In *Proceedings of the IEEE*

International Conference on Tools with Artificial Intelligence, volume 1, pages 445–451, 2008.

[200] W. Sheng and C. Zhu. Realtime recognition of complex human daily activities using human motion and location data. *IEEE Transactions on Biomedical Engineering*, 59(9):2422–2430, 2012.

[201] J. Shlens. A tutorial on principal component analysis. Technical report, Google Research, 2014.

[202] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[203] M. A. Siegler, U. Jain, B. Raj, and R. M. Stern. Automatic segmentation, classification and clustering of broadcast news audio. In *Proceedings of the DARPA Broadcast News Workshop*, page 11, 1997.

[204] M. W. Spong, S. Hutchinson, and M Vidyasagar. *Robot Modeling and Control*. John Wiley and Sons, Inc., 2006.

[205] M. Sreenivasa, M. Millard, M. Felis, K. Mombaur, and S. I. Wolf. Optimal control based stiffness identification of an ankle-foot orthosis using a predictive walking model. *Frontiers in Computational Neuroscience*, 11:23:1–13, 2017.

[206] M. Sreenivasa, M. Millard, P. Manns, and K. Mombaur. *Optimizing Wearable Assistive Devices with Neuromuscular Models and Optimal Control*, pages 627–632. Springer International Publishing, 2017.

[207] M. Stelzer, M. Hardt, and O. Von Stryk. Efficient dynamic modeling, numerical optimal control and experimental results for various gaits of a quadruped robot. In *Proceedings of the International Conference on Climbing and Walking Robots*, 2007.

[208] A. Stolcke and E. Shriberg. Automatic linguistic segmentation of conversational speech. In *Proceedings of the International Conference on Spoken Language*, volume 2, pages 1005–1008, 1996.

[209] N. Sylla, V. Bonnet, G. Venture, N. Armande, and P. Fraisse. Human arm optimal motion analysis in industrial screwing task. In *Proceedings of IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 964–969, 2014.

[210] M. Tenorth, J. Bandouch, and M. Beetz. The TUM kitchen data set of everyday manipulation activities for motion tracking and action recognition. In *IEEE International Conference on Computer Vision Workshops*, pages 1089–1096, 2009.

[211] Thalmic Labs Inc. Myo. www.thalmic.com, 2013.

[212] The MathWorks Inc. Matlab, 2012.

[213] E. Todorov. Optimality principles in sensorimotor control. *Nature Neuroscience*, 7:907–915, 2004.

[214] A. Valtazanos, D. K. Arvind, and S. Ramamoorthy. Comparative study of segmentation of periodic motion data for mobile gait analysis. In *Wireless Health*, pages 145–154, 2010.

[215] L. J. P. Van der Maaten, E. O. Postma, and H. J. Van Den Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10:1–41, 2009.

[216] B. Varadarajan, C. Reiley, H. Lin, S. Khudanpur, and G. Hager. Data-derived models for segmentation with application to surgical assessment and training. In Guang-Zhong Yang, David Hawkes, Daniel Rueckert, Alison Noble, and Chris Taylor, editors, *Medical Image Computing and Computer-Assisted Intervention*, Lecture Notes in Computer Science, pages 426–434. Springer Berlin / Heidelberg, Berlin, Germany, 2009.

[217] A. Veeraraghavan, R. Chellappa, and A.K. Roy-Chowdhury. The function space of an activity. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 959–968, 2006.

[218] I. S. Vicente, V. Kyrki, D. Kragic, and M. Larsson. Action recognition and understanding through motor primitives. *Advanced Robotics*, 21:1687–1707, 2007.

[219] Vicon Motion Systems Ltd. Vicon cameras. www.vicon.com, 1984.

[220] A. Vögele, B. Krüger, and R. Klein. Efficient unsupervised temporal segmentation of human motion. *ACM SIGGRAPH Symposium on Computer Animation*, 2014.

[221] M. Wächter and T. Asfour. Hierarchical segmentation of manipulation actions based on object relations and motion characteristics. In *Proceedings of the IEEE International Conference on Advanced Robotics*, pages 549–556, July 2015.

[222] M. Wächter, S. Schulz, T. Asfour, E. E. Aksoy, F. Wörgötter, and R. Dillmann. Action sequence reproduction based on automatic segmentation and object-action complexes. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 189–195, 2013.

[223] M. A. Watkins, D. L. Riddle, R. L. Lamb, and W. J. Personius. Reliability of goniometric measurements and visual estimates of knee range of motion obtained in a clinical setting. *Physical Therapy*, 71(2):90–96, 1991.

[224] A. D. Wilson and A. F. Bobick. Parametric hidden markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:884–900, 1999.

[225] D. Winter, G. Rau, R. Kadefos, H. Broman, and C. De Luca. Units, terms and standards in reporting of EMG research. Technical report, International Society of Electrophysiogical Kinesiology, 1980.

[226] D. A. Winter. *Biomechanics and Motor Control of Human Movement*. John Wiley and Sons, Inc., 1990.

[227] D. A. Winter. *Biomechanics and Motor Control of Human Gait: Normal, Elderly and Pathological*. University of Waterloo Press, 1991.

[228] J. Wojtusch and O. von Stryk. HuMoD - A Versatile and Open Database for the Investigation, Modeling and Simulation of Human Motion Dynamics on Actuation Level. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robots*, pages 74–79, 2015.

[229] F. Wörgötter, E. E. Aksoy, N. Kruger, J. Piater, A. Ude, and M. Tamosiunaite. A simple ontology of manipulation actions based on hand-object relations. *IEEE Transactions on Autonomous Mental Development*, 5:117–134, 2013.

[230] Y. Wu and T. Huang. Vision-based gesture recognition: A review. In *Gesture-Based Communication in Human-Computer Interaction*, volume 1739 of *Lecture Notes in Computer Science*, pages 103–115. Springer Berlin/Heidelberg, 1999.

[231] T. Yabuki and G. Venture. Motion recognition from contact force measurement. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 7245–7248, 2013.

[232] M. Yamamoto, H. Mitomi, F. Fujiwara, and T. Sato. Bayesian classification of task-oriented actions based on stochastic context-free grammar. In *International Conference on Automatic Face and Gesture Recognition*, pages 317–322, 2006.

[233] K. Yamane and Y. Nakamura. Natural motion animation through constraining and deconstraining at will. *IEEE Transactions on Visualization and Computer Graphics*, 9:352–360, 2003.

[234] Q. Yang and X. Wu. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, 5(04):597–604, 2006.

[235] L. Yardley, N. Beyer, K. Hauer, G. Kempen, C. Piot-Ziegler, and C. Todd. Development and initial validation of the falls efficacy scale-international. *Age and Ageing*, 34(6):614–19, 2005.

[236] M. Yuwono, S. W. Su, B. D. Moulton, and H. T. Nguyen. Unsupervised segmentation of heel-strike imu data using rapid cluster estimation of wavelet features. In *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, 2013.

[237] T. Zhang, M. E. Karg, J. F.-S. Lin, D. Kulić, and G. Venture. Imu based single stride identification of humans. In *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, pages 220–225, 2013.

[238] T. Zhang and C. C. J. Kuo. Audio content analysis for online audiovisual data segmentation and classification. *IEEE Transactions on Speech and Audio Processing*, 9:441–457, 2001.

[239] X. Zhang, X. Chen, Y. Li, V. Lantz, K. Wang, and J. Yang. A framework for hand gesture recognition based on accelerometer and EMG sensors. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41:1064–76, 2011.

[240] Z. Zhang. Microsoft kinect sensor and its effect. *IEEE Multimedia*, 19:4–10, 2012.

[241] X. Zhao, X. Li, C. Pang, X. Zhu, and Quan Z. Sheng. Online human gesture recognition from motion data streams. In *Proceedings of the ACM International Conference on Multimedia*, MM '13, pages 23–32, New York, NY, USA, 2013. ACM.

[242] F. Zhou, F. de la Torre, and J. K. Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:582–596, 2013.

[243] H. Zhou and H. Hu. Human motion tracking for rehabilitation - a survey. *Biomedical Signal Processing and Control*, 3(1):1–18, 2008.

[244] R. Zollner, T. Asfour, and R. Dillmann. Programming by demonstration: dual-arm manipulation tasks for humanoid robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 479–484 vol.1, 2004.

Appendices

Appendix A

Segmentation Related Works Tables

This appendix contains a summary of the reported accuracy of the surveyed papers from Chapter 4.

Table A.1: Online segmentation algorithms with validation results. The specifications (Spec) denotes the algorithm's computational effort (online (O)), template (supervised (S), unsupervised (U) or adaptive (A)), and segment definition (physical (P), derived (D) or template (T)). For the data used, joint angle (q), Cartesian (x), gyroscope (w), accelerometer (a), and force (f) data are used. The number of subjects (n_s) and primitives (n_p) used, as well as reported verification (Ver) and accuracy (Acc) method are also denoted.

	Spec	**Segmentation Algorithm**	n_s	n_p	**Data**	**Ver**	**Acc**	[%]
[114]	O/A/D	Segment on high PDF distances between windows.	1	45	q; exercise [114].	Temporal, $t_{err} = 0.12s$.	F_1	75-76%
[117]	O/A/D	Threshold on neighbouring topical similarity.	6	51	x; exercise [43, 158].	Temporal, gradient.	F_1	87-90%
[7]	O/U/D	Piece-wise regression.	2	4	q; eating.	Temporal.	F_1	48-92%
[18]	O/U/D	Segment on distance threshold between windows.	-	7	q; ADL.	Temporal inv.	F_1	93%
[19]	O/U/D	GLR.	5	3	x; hand [17].	Labels.	$1 - \frac{FP}{TP}$	91-96%
[145]	O/U/D	Distance and velocity threshold	-	19	q, hand	Labels	F_1	97
[220]	O/U/D	Region grow from moving window distance matrix.	6	51	x; exercise [43, 158].	Labels.	Precision.	88-97%
[237]	O/U/P	Gyroscope turning points.	20	5	w; gait.	Labels.	$\frac{TP}{total}$	80-99%

Table A.2: Semi-online segmentation algorithms with validation results. The specifications (Spec) denotes the algorithm's computational effort (online (O)), template (supervised (S), unsupervised (U) or adaptive (A)), and segment definition (physical (P), derived (D) or template (T)). For the data used, joint angle (q), Cartesian (x), gyroscope (w), accelerometer (a), and force (f) data are used. The number of subjects (n_s) and primitives (n_p) used, as well as reported verification (Ver) and accuracy (Acc) method are also denoted.

	Spec	**Segmentation Algorithm**	n_s	n_p	**Data**	**Ver**	**Acc**	[%]
[20]	M/S/T	Piecewise regression slopes discretization. Sequences labelled by bag-of-word.	6	1	a; exercise.	-	F_1	60-95%
[45]	M/S/T	SVM hyperplane used in HMM emissions. Segment by online Viterbi.	-	4	f; teleop.	Labels.	Class.	84%
[124]	M/S/T	Online Viterbi.	4	3	q; tracing.	Labels.	Class.	94%
[134]	M/S/T	Velocity profiles matched to observation, then verified on HMM.	20	5	q; exercise.	Temporal, $t_{err} = 0.12s$.	F_1	85-90%
[149]	M/S/T	DMP in Kalman filter parameter estimation.	1	22	x; writing.	Labels.	Class.	90-100%
[241]	M/S/T	Threshold on similarity between observation and time aligned low-DOF templates.	30	12	x; exercise [68].	Labels.	F_1	55-73%

Table A.3: Offline segmentation algorithms with validation results. The specifications (Spec) denotes the algorithm's computational effort (online (O)), template (supervised (S), unsupervised (U) or adaptive (A)), and segment definition (physical (P), derived (D) or template (T)). For the data used, joint angle (q), Cartesian (x), gyroscope (w), accelerometer (a), and force (f) data are used. The number of subjects (n_s) and primitives (n_p) used, as well as reported verification (Ver) and accuracy (Acc) method are also denoted.

	Spec	**Segmentation Algorithm**	n_s	n_p	**Data**	**Ver**	**Acc**	[%]
[12]	F/S/T	Statistical features used in classifiers, then HMM. Segment by Viterbi.	48	9	a; ADL.	Labels.	Class.	76-99%
[47]	F/S/T	Viterbi fitted data to regression model library.	6	12	a; ADL.	Labels.	Class.	90%
[140]	F/S/T	AdaBoost HMM.	-	22	x; various.	Labels.	Class.	84-100%
[184]	F/S/T	DTW with dynamic band.	2	2	x; hand.	Labels.	Class.	100%
[190]	F/S/T	Viterbi algorithm.	5	6	f; insole.	Temporal, $t_{err} = 0.05s$.	Precision.	73-90%
[216]	F/S/T	Viterbi algorithm.	8	20	q; surgical.	Labels.	Class.	70-87%
[218]	F/S/T	SVM labels. used as HMM feature vectors. Segment by Viterbi.	10	12	x; reaching.	Labels.	Class.	48-95%
[159]	F/U/D	GMM/feature clustering	8	2	q, surgical. Video.	Labels.	NMI.	0.27-0.64
[221]	F/U/D	Normalized jerk subwindow ratio	-	-	q, ADL	Temporal		0.27 s
[242]	F/U/D	Minimize segment-cluster distances.	36	25	q, x; ADL [43]. Video.	Labels.	Class.	77-83%

Appendix B

EMG Features

In the following definitions, $E(t)$ is the raw EMG signal, W_n is the length of the window, t_0, t_1 ... t_n denote entries in the window.

Mean Absolute Value

$MAV = \sum_{t_0}^{t_n} \frac{1}{W_n}|E(t)|$

The MAV is the sum of the absolute values of the EMG signal over a window, which effectively computes a moving average filter of the EMG signal.

Waveform-length

$WFL = \sum_{t_0}^{t_n} |\dot{E(t)}|$

The WFL is the cumulative successive change of the EMG signal over a temporal interval. If a signal fluctuates greatly within a window, WFL is high, while a signal with low WFL does not contain a lot of local variations.

Pairwise inner-product

$PIP = E(t)_x \cdot E(t)_y$

where $E(t)_x$ and $E(t)_y$ refer to specific channels of the EMG. The PIP is calculated by taking the dot product of a moving window between two channels, and captures the interactions between channel-pairs.

Root mean square

$RMS = \sqrt{\sum_{t_0}^{t_n} \frac{1}{W_n} E(t)^2}$

The RMS provides a measure of the signal power.

Teager energy

$TE = \dot{E}^2(t) - E(t) \cdot \ddot{E}(t)$

The TE is a local property of the signal, that varies with the amplitude profiles and instantaneous frequency of the signal. TE captures the energy required to generate the signal with various amplitude and frequency specifications. For two signals with similar amplitude profile and different frequency components, the Teager energy returns different values.

Glossary

aggregators
: a class of techniques that combines several classifiers together to create a more accurate classifier (Section 2.3)

automated rehabilitation
: a system designed to provide supervision and feedback on patient exercises

classifiers
: a class of techniques that learns class boundaries using training data (Section 2.2)

data, known
: data primitives or participants that have been included in the training set

data, labelled
: data that has manual segment data labelled, and is typically used for algorithm training and validation

data, novel
: data primitives or participants that have not been included in the training set

data, unlabelled
: data that does not have manual segment data labelled

dimensionality transformation
: a class of techniques that transforms the input data from one feature space to another (Section 2.1)

features
: a variable that is derived from a sensor modality or through an algorithm

generalizability, inter-participant
  an algorithm that can be trained on data from one set of participants and be tested on another set of participants

generalizability, inter-primitive
  an algorithm that can be trained on data from one set of primitive and be tested on another set of primitive

gesture recognition
  process of identifying the primitive under a sliding window

identification
  labelling segmented data with the appropriate motion type

inertial measurement units
  a sensor package that consists of an accelerometer, gyroscope, and occasionally an magnetometer

joint compensation
  the usage of improper joints to perform prescribed movements

Karush-Kuhuh-Tucker conditions
  a set of necessary conditions for non-linear programming optimality (Section 2.4.3)

machine learning
  a class of techniques that utilizes statistics, optimization, and analytics to perform classification and prediction

motion primitives
  small lengths of motions of interest

optimal control
  a class of techniques that produces a controller that minimizes a given cost function (Section 2.4)

principal component analysis
  an unsupervised dimensionality transformation technique (Section 2.1.1)

real-time
  an algorithm that can process data faster than human reaction speed

rehabilitation
process of restoring movement capabilities through physiotherapy exercise

segmentation
process of extracting interesting time points from observed data

support vector machine
a supervised statistical classifier technique (Section 2.2.4)

trajectory optimization
a class of techniques that produces a trajectory that minimizes a given cost function (Section 2.4)

variability, spatial
variance that occurs in the feature space

variability, temporal
variance that occurs in time