A Framework for Human Motion Strategy Identification and Analysis

by

Muhammad Umar Choudry

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Masters of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2019

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Parts of this thesis have been adapted from previous works I authored/co-authored. The method described in [1] is used to compare differences in observation space variables between Hidden Markov Models (HMMs) models of two different movement trajectories. Additionally, the divisive clustering algorithm described in [2] is used to cluster similar time series datasets to detect movement strategies.

In order to create the software to test the proposed framework, we customized several publicly available software packages. The software package that was modified for training HMM's was the JaHMM (Java HMM) package [3]. Several modifications were made to this package to support more efficient HMM training, HMM based trajectory reconstruction, and to support various HMM based clustering algorithms. We used the Matlab-based Gaussian Process Dynamical Models (GPDM) code provided by J.M. Wang [4] as a starting point for the GPDM training algorithms described within the thesis. Several modifications were made to these scripts to support training acyclic motion trajectories of various lengths and to determine the dimensionality of the latent space based on model reconstruction error.

There are two main comparison algorithms that we compare our approach against. Joint Component Vector [5] and Kinematic Joint Synergies [6]. Both of these approaches were implemented after the original works and coded using Matlab.

To validate the proposed framework in this thesis, two different motion capture datasets were used for analysis. These datasets were collected for prior work by Jack P. Callaghan and Tyson Beach at the University of Waterloo's Kinesiology Department. The first dataset consisted of participants performing a known predetermined lifting strategy - squat or stoop [7]. The second dataset was collected as part of an experiment for examining the influence of unilateral ankle immobilization on lower back loading and injury potential during lifting [8]. Both of these datasets were collected using an optoelectronic motion capture system (Optotrak Certus, NDI, Waterloo, Canada) and consisted of the positions and orientations of the various joints. Standard inverse kinematics computations were used to convert the Cartesian coordinates into Cardan joint angles for the ankles, knees, hips, and lumbar spine (Visual3D™ Software, C-Motion Inc.).

Abstract

The human body has many biomechanical degrees of freedom and thus multiple movement strategies can be employed to execute any given task. Automated identification and classification of these movement strategies have potential applications in various fields including sports performance research, rehabilitation, and injury prevention. For example, in the field of rehabilitation, the choice of movement strategy can impact joint loading patterns and risk of injury. The problem of identifying movement strategies is related to the problem of classifying variations in the observed motions. When differences between two movement trajectories performing the same task are large, they are considered to be different movement strategies. Conversely, when the differences between observed movements are small, they are considered to be variations of the same movement strategy. In the simplest scenario a movement strategy can represent a cluster of similar movement trajectories, but in more complicated scenarios differences in movements could also lie on a continuum. The goal of this thesis is to develop a computational framework to automatically recognize different movement strategies for performing a task and to identify what makes each strategy different.

The proposed framework utilizes Gaussian Process Dynamical Models (GPDM) to convert human motion trajectories from their original high dimensional representation to a trajectory in a lower dimensional space (i.e. the latent space). The dimensionality of the latent space is determined by iteratively increasing the dimensionality until the reduction in reconstruction error between iterations becomes small. Then, the lower dimensional trajectories are clustered using a Hidden Markov Model (HMM) clustering algorithm to identify movement strategies in an unsupervised manner. Next, we introduce an HMM-based technique for detecting differences in signals between two HMM models. This technique is used to compare latent space variables between the low-dimensional trajectory models as well as differences in degrees-of-freedom (DoF) between the corresponding high-dimensional (original) trajectory models. Then, through correlating latent variable and DoF differences movement synergies are discovered.

To validate the proposed framework, it was tested on 3 different datasets – a synthetic dataset, a real labeled motion capture dataset, and an unlabeled motion capture dataset. The proposed framework achieved higher classification accuracy against competing algorithms (Joint Component Vector and Kinematic Synergies) where labels were known apriori. Additionally, the proposed algorithm showed that it was able to discover strategies that were not known apriori and how the strategies differed.

Acknowledgements

I would like to thank my supervisor Dana Kulic for her guidance and support from my first introductory undergraduate robotics class through my graduate studies at the University of Waterloo. It was always great to talk endlessly with someone who has the same fervor and fascination with the future of robotics research as I do. Thanks for being a great mentor and supporting me throughout the years through thick and thin.

I would also like to thank the following people for their assistance during my research:

- Matt Pillar who initially started the motion analysis project using the library JaHMM. His initial work on the project was my first exposure to HMM's and it has been an adventure since.
- Tyson Beach and Jack Callaghan for collaborating with us and providing us with human motion capture data without which the analysis in this thesis would not be possible.

I would like to thank my parents for their endless love, support and encouragement throughout my academic career at the University of Waterloo. Last but not least, I would like to thank my wife Farah for always pushing me to be better and always convincing me that there's a light at the end of the tunnel.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Given the redundancy of the human body at both the joint and muscle levels, multiple movement strategies can be employed to execute a given task. Movement strategy differences can be exhibited both through different muscle recruitment patterns and different kinematic trajectories. This flexibility is advantageous because it allows individuals to adapt to varying environments, execute a wide variety of tasks, and compensate for personal morphological (e.g. body size and shape) or functional (e.g. joint range-of-motion, strength, fatigue, etc.) movement constraints. However, inter- and intra-individual variations in movement strategies pose significant challenges for human motion analysis and make it difficult to objectively identify and classify movement strategies.

The automated identification and classification of movement strategies have many potential applications including sports performance research, rehabilitation, and injury prevention. In sports performance research, there may be interest in applications that attempt to determine the skill level an athlete has attained through performing certain exercises and regimens [9] [10] [11]. In rehabilitation applications, there is interest in determining whether specific intervention strategies are suitable for all types of patients and whether patients with different classes of movement variation are impacted in different ways [12] [13]. Lastly, for injury prevention applications, there may be interest in identifying ergonomically incorrect strategies and taking preventative measures to correct those strategies [5] [14].

A variety of artificial intelligence techniques have been proposed for quantitative analysis of movement variation [15]. At present, there are few quantitative methods capable of objectively identifying and grouping inter- and intra-personal movement strategies, especially for complex whole-body tasks that are performed variably between individuals and over time. Most current algorithms abstract and condition multivariate time-series data and treat movement variability as noise. This represents a significant loss of information, as it has been argued convincingly that movement variability is an important observed feature of human performance that should not be treated as noise [15]. Many existing approaches also result in a loss of temporal information during a feature extraction step and require apriori knowledge of (expected) movement behaviors and underlying data structures. A desirable attribute of a motion analysis framework would be the ability to handle arbitrary movement while accounting for movement progression and motion variability.

Another desirable attribute would be the capability to generate exemplar motions that represent the average within a category or identify the best-fit motion from existing data for a specific category of

movement for visual motion comparison. Visual comparison of motions may not be sufficient for understanding how or why the movements are different. Therefore, it would be ideal if the framework could automatically identify key features that differentiate the motions. Finally, human motion analyses often incorporate a variety of signals, so it would be desirable if the algorithm could operate on any kind of temporal signal (e.g. Cartesian trajectories, joint angle trajectories, or electromyographic signals). In this thesis, we propose a motion analysis framework to achieve all of the aforementioned desired qualities. In particular our focus is on movement strategy differences exhibited through variations in kinematic trajectories and we show that the proposed approach can be applied to both joint angle data and Cartesian data by simply providing different input data to the algorithm.

1.1 Contributions

There are three main contributions of this thesis:

Development of an Unsupervised Approach for Movement Strategy Detection

In order to recognize unique movement strategies, an algorithm for unsupervised clustering of multivariate time series trajectories is developed. High dimensional time series trajectories representing human movements are converted to a lower dimensional (latent) space using Gaussian Process Dynamical Models (GPDM). The dimensionality of the latent space is determined iteratively by using the reconstruction error. In order to achieve improved clustering speed and performance over clustering observation trajectories directly, we propose clustering latent space trajectories, as they theoretically encapsulate only the critical information needed for representing the motion. This clustering is done using a Hidden Markov Model (HMM) based approach.

Correlation of Variance in Low Dimensional Latent Space with High Dimensional Joint Space

The variations in the low dimensional space are then correlated back to the high dimensional space using a novel HMM-based approach. An automated method is used to quantitively compare the strategy movement models extracted from the nodes of the motion tree. This method is used to detect differences in the latent space variables between the low-dimensional trajectory models and to detect differences in the variables (joints or Degrees-of-Freedom) between the corresponding high-dimensional (original) trajectory models. Then differences in movement synergies are discovered by correlating latent space variable variations with DoF variations.

Comparison of Various Approaches for Strategy Identification and Analysis

The utility and accuracy of the proposed motion analysis framework is validated on three different datasets, which demonstrates the ability of the proposed framework to detect movement strategies from full body movement data. First, validation on a synthetic dataset shows that the proposed framework performs well under scenarios where movements have high variation and minimal difference in movement strategy. A second dataset consisting of known squat and stoop lifting strategies shows that our framework can correctly identify known strategies. Finally, validation on a dataset with unknown strategies demonstrates that the proposed algorithm accurately detects strategies and sub-strategies without any apriori knowledge of the movement type to be analyzed despite the complex nature of the movement and a limited size dataset. The proposed framework is compared against existing strategies in the literature that have been developed to solve the problem of movement strategy identification and analysis. These methods include a feature-based clustering approach (Joint Vector Component), kinematic synergy approach using matrix factorization, and an HMM-based clustering approach.

1.2 Thesis Organization

This thesis is organized as follows. Chapter 2 discusses related work in the field of human movement analysis. This chapter starts by reviewing feature-based approaches before describing how HMMs, dynamic modeling, neural networks, and kinematic synergies can be used for human motion analysis. Chapter 3 provides background theory on HMMs and GPDMs for human motion modeling. Additionally, we provide an overview of comparison algorithms (Joint Component Vector and Kinematic Synergies) from the literature, which are compared with our proposed approach in the validation chapter (Chapter 5). Next the proposed GPDM-HMM hybrid framework for analyzing human motions is explained in Chapter 4. This includes details of the proposed HMM based divisive clustering algorithm for grouping similar time series trajectories, as well as augmentations to the basic GPDM algorithm for learning latent space trajectories with appropriate number of latent variables. Chapter 5 validates the utility and accuracy of the proposed framework with three different datasets: a synthetic dataset, a real dataset where the main motion strategies are known a priori, and a dataset where the number of strategies is not known beforehand. Conclusions and directions for future work are provided in Chapter 6.

Chapter 2
Related Work

The use of artificial intelligence techniques to categorize and analyze human motions has been gaining popularity in the fields of robotics, machine learning, and computer vision [16] [17]. In robotics, there is a focus on learning a variety of motion primitives for the purposes of motion recognition and generation [18] [19] [20]. Many computer vision based works focus on the extraction and classification of motions from video [21] [22]. Due to the increase in popularity of various motion capture technologies [23], there has been a growing interest in the use of artificial intelligence techniques to improve motion analysis in the fields of biomechanics and kinesiology [24]. In these fields there is a need to understand why movement variations occur for similar motions or tasks. For example, various artificial intelligence techniques have been applied to gait analysis to understand why gait variations occur [25] [26] [27].

In some applications, class labels may be known beforehand and it is desirable to recognize when new participants exhibit behaviors from a pre-specified set of motion classes. Typically, supervised learning algorithms are used in these scenarios, whereby motions or feature sets are labeled, and a classifier is trained to recognize motions that are similar to those in the labeled datasets. Once a classifier has been trained to recognize a particular motion, the underlying statistics of the feature sets or models can be analyzed to understand the differences between the motions. In prior work, a variety of techniques such as Neural networks [28] [29], Support Vector Machines [9] [30] [31], HMM [32] [33], and Decision Trees [34] have been used for this purpose.

Given our interest in detecting motion classifications from unlabeled data, we will focus our discussion of the related works on unsupervised learning algorithms. For this purpose, methods that can classify temporal data [35] [36] in an unsupervised manner are of interest. Some algorithms that can be used for classifying human motions include discriminative / feature-based approaches [37], HMM based approaches, dynamic modeling approaches [38], neural networks, and kinematic synergy approaches [6].

2.1 Discriminative / Feature-Based Approaches

A number of prior works have classified human motions with discriminative clustering. In these approaches, the temporal motion data is abstracted to a set of features which are then used to cluster the motions with an algorithm such as k-means clustering. This approach has been used extensively to

classify gait patterns [39] [40] [41] [42] [43]. For example, in the work by Toro et al, the k-means clustering algorithm was used to detect classes of gait patterns for children with cerebral palsy [39]. In order to determine K (the number of strategies) various numbers of clusters were used to classify the data until the results met three criteria for stable clusters: (i) Normal children should be in their own cluster, (ii) Kinematic displacement should be different between clusters, and (iii) Standard deviations should be low within clusters.

In a slightly different approach, O'Malley used fuzzy c-means clustering to classify the gait data of children with cerebral palsy using two features: cadence and stride length [12]. Their datasets included pre- and postoperative test data and the clusters were used to indicate how a patient's gait improved when the patient moved from one cluster to another (before and after surgery). The disadvantage of discriminative based approaches such as these is that the features that are used are very specific to a singular motion type and often cannot be extended to other motions.

Park et al proposed a more general approach to discriminative clustering by defining a generic feature set that accounted for the contribution of each joint to a motion [5]. They proposed abstracting human motion to a feature vector called the Joint Component Vector (JCV) that contained the normalized contributions of each DoF to perform a goal-oriented task (in the Cartesian domain). They used k-means clustering to cluster the JCV vectors, where the number of clusters was determined either based on prior knowledge or by using the Multi-Dimensional Scaling algorithm [44]. They tested their algorithms on two different lifting data sets with > 80% accuracy for a labeled dataset. Menceur et al [45] extended this approach by proposing an agglomerative clustering-based approach (using dendograms) to identify motion clusters.

Dynamic Time Warping (DTW) has also been used in human motion analysis, due to DTW's ability to compare temporal data of varying lengths. Kulbacki et al proposed an unsupervised learning algorithm for learning motion models [46]. In their approach, motion primitives were represented as spline curves (abstracted to spline parameters) and motion models were represented as distributions of the spline parameters. DTW was used to find the measure of discrepancy between two motion sequences, where their DTW metric accounted for Euclidean distance between data points as well as the square of the difference of the estimated derivatives. They used the DTW distance metric to perform agglomerative clustering to categorize motions. Gueguin et al took a similar approach with DTW distance to perform agglomerative clustering [47]. In their application, they were trying to cluster time-series data from cardiac-implantable devices. To reduce the dimensionality of the data, they used Multiple Correspondence Analysis (MCA) and fuzzy coding. They found the first three axes of the

MCA contained more than 90% of the total variance linked to the functional state of their patients. The results of their experiment showed the discovery of three clusters, where intra-cluster participants showed similar patterns of activity and ventilation.

In the work by Nakamura et al, the group attempted to translate the numerical differences between skilled movement and unskilled movement for athletes into human readable text [48]. They proposed breaking up a full motion into sub-motions or primitives, where they proposed that correlation of sub-motions for skilled and unskilled athletes may indicate differences in skill level. Similarity between all sub-motions was calculated using a new proposed metric named Angular Metrics for Shape Similarity (AMSS) and a DBSCAN [49] algorithm for clustering the motions. The motions were labeled after clustering in order to identify the content's sub-motion clusters as skilled/unskilled motions. In order to generate human understandable text from the numeric data representing the skills, Bayesian Networks were used. Here the states of motions had corresponding sentence templates that were used to construct the description.

2.2 HMM Based Approaches

An advantage of using HMM over discriminative or feature-based approaches is that they are able to capture the temporal progression of a time series dataset by modeling the temporal evolution of a hidden state variable as a stochastic first order process [50]. This has made them a good candidate for human motion analysis in prior works.

In the work by Kulic et al, the effect of a several week training process on the motions of a marathon runner was analyzed [10]. In this work, motions were modeled with HMMs and an incremental agglomerative clustering algorithm (stored in a tree format) was used to classify the motions - after the training period, new sub-nodes within the motion tree were identified. This indicated that their method was capable of determining when the quality of motions was changing.

Vlasko et al. [51] proposed clustering the states within an HMM to understand the similarity of motions. HMM's were used to estimate the topology of repetitive lifting data for patients who underwent treatment for lower back pain given three labeled categories of lifting motions: a control group, pre-treatment patients, and post-treatment patients [51]. The number of states was determined by initializing with a high number of states and removing states until the "knee" was reached for the probability that the models generated the data. The states that were above the "knee" were then used to model possible structures for the lifting motion. Note that the "knee" is the point at which increasing the number of states results in marginal increase in probability that the model generated the observation

sequence. Initialization for each of the states was done by performing k-means clustering on the motion vectors and pruning of states was done to find the minimal representation of the model. State transitions and various features for each state were then compared in detail to understand the lifting dynamics of the individual groups.

In addition to the above-mentioned works that pertain specifically to the applications of HMM for motion clustering, there is considerable prior work in the field of HMM clustering of multivariate time series data. For example, Li et al proposed HMM clustering focused on the automated selection model sizes and number of clusters (based on the BIC criterion) [52]. In each iteration of the algorithm, a seed was selected based on the lowest probability of belonging to a cluster and then sequences were redistributed with highest likelihood. They used a pseudo-hierarchical divisive technique, modeling the training set first with a single HMM and then adding more components so as to split the set into progressively smaller groups.

Butler et al compared different versions of HMM-based Divisive and Agglomerative algorithms for clustering acoustic data [53]. In our work, we have proposed several improvements over his algorithms such as the use of the KL distance for agglomerative clustering in comparison to his "average link method." Their proposed divisive algorithm is seed based, where the poorest fit cluster is split by choosing a seed item at random from this cluster to form a new cluster. This approach leads to a "fracturing" of the clusters that is undesirable, whereas in our approach we use subtractive and *c*-means clustering with iterative training to avoid this problem.

2.3 Dynamic Modeling Approaches

Dynamic modeling serves as an alternative to the discriminative or stochastic models discussed so far. Dynamic models can be divided into two main categories - linear dynamical systems (LDS) and nonlinear dynamical systems. As human motions are inherently nonlinear, we focus on nonlinear approaches.

Schaal et al introduced dynamic motion primitives (DMP), an approach that models the attractor behavior (behavior that shows progress towards some goal state) of nonlinear dynamical systems [4]. This was accomplished by parameterizing the dynamic model with a set of linear differential equations with the use of a learnable autonomous forcing term. The work then analyzed DMPs from a biological point of view and found parallels between the expected dynamics and the learned dynamics.

Wang et al developed an alternative approach to modeling the dynamics of a system called Gaussian Process Dynamical Model [4]. Their approach is based on the work of Lawrence et al on the concept

of Gaussian Process Latent Variable Model (GPLVM) [54]. GPLVM uses Gaussian Processes (GP) for dimensionality reduction by finding a non-linear function that maps low-dimensional latent space vectors to a high-dimensional observation space. While GPLVM can be used on any kind of time series multivariate data, GPDMs introduce constraints between successive data points that capture the dynamics of the system. Wang et al tested their GPDM approach on motion capture data of various participants walking and performing golf swings. They found that their latent space representation of the movements was able to capture the dynamics of the movement and was able to generate smooth motions when trying to generate missing frames of data for the walking dataset. In our work, we will be using GPDM to model human motions because of their ability to reduce the dimensionality to a latent space, where the latent variables can characterize the motions.

In the work by Hong et al, the authors used GPDMs to process a gait trajectory for a rehabilitation robot that assisted with walking and ensured no collision occurred between the ground and foot during the swing phase [55]. The robot assistance to modify the trajectory was activated when the participant's trajectory strayed too far from the reference trajectory. Their simulation results showed that their approach recognized instances where collisions would occur and reduced the 15% collision rate down to 0% in simulation. While Hong et al's work focused mainly on deviation from a predefined trajectory, our work focuses on granular differences between joints that will help us better describe motion variability.

In [38], Fan et al conducted literature review that compared various approaches based on GP for modeling and analyzing gait. These included a comparison of variants of GPLVM, GDPM and a newly proposed Joint Gait-Pose Manifold (JGPM). The Joint Gait-Pose Manifold is an approach specific to gait analysis where the manifold is assumed to be toroidal in structure due to the cyclic nature of walking. Their results showed that LL-GPDM (a version of GDPM where motion trajectories are forced into a cylindrical manifold) and the JGPM approach outperformed the other approaches in extrapolation, filtering, and recognition test scenarios. While Fan et al's work showed that constraining the manifold worked well when the motion type is known apriori, in our models we do not utilize any of these approaches to allow our method to handle arbitrary (both cyclic and non-cyclic) motion that is not specified to the algorithm apriori.

2.4 Neural Network Approaches

Neural networks more have also been used for the purposes of time series classification and analysis. One common approach involving neural networks involves taking features computed from

time series data and then training neural networks (in a supervised manner) to learn how to classify motions into categories known apriori [28] [56] [57]. There are also other neural network architectures that can predict the future based on a time series input [58] [59] [60]. In this section we will not focus on these approaches as our focus is on the ability to analyze and compare motions and these approaches do not provide an easy way to compare trained models. Alternatively, there exists a class of techniques that involve learning metrics using neural networks for comparing time series datasets [61] [62]. Such an approach can quantitatively allow us to compare movements. Additional there are also approaches use self-organizing maps (SOM) to produce an output that can visualize movement trajectory variability [63] [64] [65].

In the work done by Coskun et al, the authors generate a similarity metric for comparing human motions using a novel neural network architecture [62]. Their architecture includes a novel metric learning objective based on a triplet architecture and also uses attentive recurrent neural network. A triplet architecture consists of same three feed forward networks (with shared parameters), that encodes the pair of distances between 3 inputs x, x^+, and x^-, where the objective is to correctly classify which of x^+ and x^- is of the same class as x. The attentive recurrent neural network allows processing of time series datasets of variable lengths to a fixed size of embedding. They compared their architecture against various time warping techniques as well as various configurations of their architecture. Their results showed that the metric they learned was able to more accurately capture the contextual information about the motions and was able to achieve similarity results up to 20% better than competing metrics.

Self-organizing maps differ from traditional neural networks as they use competitive learning as opposed to error-based learning and typically produce two-dimensional discretized representation of the time series data set. Lamb et al proposed to use SOM to classify the movement patterns of participants taking different kinds of basketball shots from various distances [63]. They visualized the output of the SOM using a U-matrix (a grid that shows the average distance between nodes weight vector and that of its neighbors). Using this approach, the authors were able to show groupings in the various shooting conditions that were unexpected. In a different approach involving clustering, Wu et al proposed the use of hierarchical clustering to index and retrieve human motion data [64]. By utilizing SOM and the Smith–Waterman algorithm for measuring temporal similarity, a distance matrix is created for the motions. The distances are used in an agglomerative clustering algorithm to create a tree where the root of the tree consists of a single cluster containing all observations while the leaves correspond to individual observations. In another approach, Perl proposed the use of Dynamically

Controlled Networks (a special type of SOM that is able to recognize patterns as members of pattern clusters) for unsupervised learning of motions [65]. Once the network has been trained with motion input values, the data effects the distribution of the neurons and therefore the cluster of neurons can be used to determine the cluster of motions. In most of the SOM based approaches, if motions need to be compared, the motion trajectories need to be pre-processed so that the motion sequences are of equal duration.

2.5 Motion Synergies

In addition to categorizing movement variations, it is also important to understand how the movements are different. In the field of kinematics, the theory of motion synergies provides a way to describe how motions differ. The roots of this approach can be traced back to Jackson Hughling's work [66], however in modern literature the theory is most commonly associated with N.A. Bernstein [67]. The theory of motion synergy assumes that the central nervous system does not control individual muscles but unites them in groups and controls them in unison. Therefore, the number of control variables is not necessarily the number of muscles involved in the movement but is a reduced number of variables depending on the synergy (usually called muscle modes or M-modes). When there are differences in motions, the theory is that the control signal to the synergy changes which leads to coordinated changes in the muscle movements. Note that because groups of muscles can span multiple joints, it is possible to conceptualize that motion synergies could be mapped to the coordination of multiple joints (or Degrees-of-Freedom) and their contribution to a movement.

Earlier studies of movement synergies [68] [69] proposed using principal component analysis (PCA) and cluster analysis to detect movement synergies. These techniques typically first apply PCA to EMG's of muscles during a movement task (for example movement of elbow where the joint position is fixed). This would reduce the movement down to a few factors. The PCA results from different participants then undergo clustering analysis where groups of similar PCA results are interpreted as distinct muscle synergies.

More recent studies use techniques like Non-Negative Matrix Factorization (NNMF) to determine movement synergies. For example in the work by Coscia et al [70], the authors use this technique to study how participants compensate for the effect of gravity in rehabilitation exercises by providing arm-weight support (WS), which results in a reduction in activation of upper limb muscles. They used EMG data collected from upper limb muscles and discovered muscle synergies with the use of NNMF.

Because NNMF requires initial conditions to be specified which can impact the resulting synergies, NNMF was repeated 50 times from a random initial condition. Out of the 50 runs, the movement synergies that explained the most variance were chosen as the variation of choice. Using this approach the authors were able to organize the arm reaching movements and study the effects of WS in stroke survivors and how they could benefit from upper limb rehabilitation.

Steele et al [71] compare five common algorithms for identifying muscle synergies. These include NNMF, PCA, Fast Independent Component Analysis (fICA), a combination PCA and fICA (PCAICA), and probabilistic ICA. Each of these approaches produces estimated synergies as a function of muscle activations but each approach has different underlying assumptions about the model (for example some approaches maximize variance, while others aim to reduce the minimum sum of squared error between estimated and original muscle activations, whereas other approaches allow for estimates of noise in the model). Their results found that these techniques in general performed well as long as the impact of biomechanical and task constraints were minor in comparison. In a study where movements were analyzed with various degrees of variability, it was found that NNMF consistently outperformed other techniques in terms of capturing the movement decomposition, while PCA was consistently the worst.

Additionally, Lambert-Shirzad et al [6] apply the concept of muscle synergies (which take EMG time series data as an input) and apply them to joint angle data. The term for this type of synergy is called kinematic joint synergy. Lambert-Shirzad applied various factorization algorithms like PCA, ICA and NNMF on joint angle data and found that ICA and NNMF perform well, while PCA performed poorly when reconstructing the original motions. They used these techniques to analyze the impact stroke has on human movements that have no physical constraints. Through their analysis they found that for stroke survivors, motor synergies were preserved in the less-affected arm but were altered in stroke-affected arm through merging and fractionation of healthy synergies.

2.6 Comparison

Looking at the competing approaches that was covered in the related works, we find that at a high level that the approaches have the following shortcomings:

- **Discriminative/Feature Based:** Accuracy relies on correct set of features and often relies on abstracting a time series dataset to features which results in loss of temporal information.
- **HMM Based Modeling:** Hard clustering of HMM models fail to capture how movements can be categorized in more than one way.

- **Dynamic Modeling:** When movements are modeled in a latent space, it can be challenging to interpret the physical meaning of the latent variables.
- **Neural Networks:** Specific kinds of neural networks can aid in comparing and grouping movements, but these networks are often limited in the type of quantitative data that can be extracted for the purpose of analysis.
- **Kinematic Synergies:** Limitations like movements constraints can impact the quality of results.

Our proposed framework will attempt deal with the shortcomings that are faced by some of the above algorithms. Though we use some of these techniques (HMMs and dynamic modeling) in the proposed framework, we show how they can be used in together to overcome their individual shortcomings.

Chapter 3
Background

3.1 Overview

In this chapter we provide the background theory required for understanding our proposed approach (to be introduced in Chapter 4). We review Hidden Markov Models (HMM) and Gaussian Process Dynamical Models (GPDM), as both of these algorithms are used in the proposed approach to model human motions. Both of these algorithms allow us to train generative models for human motions and provide mechanisms by which we can compare the models to get insights into differences between human motions.

3.2 Hidden Markov Models

3.2.1 Introduction

Hidden Markov Models (HMM) are a stochastic method of characterizing signals to model a discrete-state dynamical system

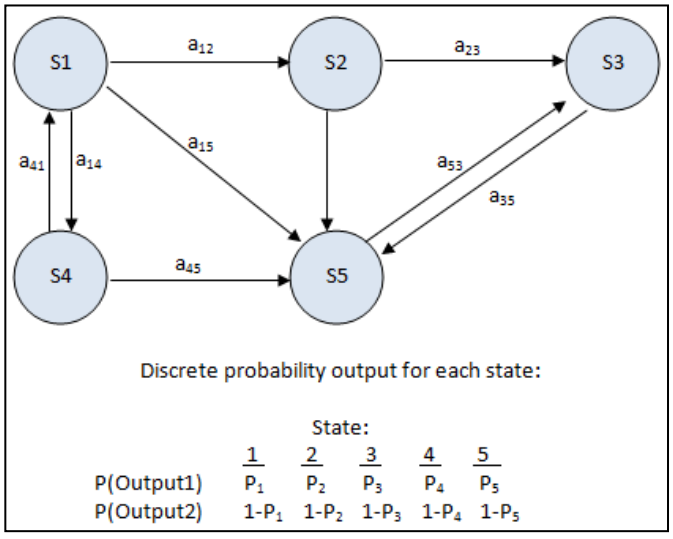


Figure 3.1 – Example of a Markov Model

HMMs model the temporal evolution of a hidden state variable as a stochastic first order process. At each time step, observations are drawn from an observation function, which is dependent on the current value of the hidden state variable. Every HMM λ is composed of three main parameters:

$$\lambda = (\pi, A, B) \tag{3.1}$$

π is the initial probability vector, A is the transition matrix and B is the probability distribution function. The initial probability vector is used to describe the probability of the initial state of an HMM sequence,

$$\pi = [\pi_1, \pi_2, \ldots \pi_N], \quad 0 \leq \pi_i \leq 1, \qquad \sum_{j=1}^{N} \pi_i = 1 \tag{3.2}$$

N is the total number of states of the system and π_i is the probability that the HMM will begin in S_i. Elements of the transition matrix A are defined as:

$$a_{ij} = P\{q_t = S_j | q_{t-1} = S_i\}, \quad a_{ij} \geq 0, \quad \sum_{j=1}^{N} a_{ij} = 1 \tag{3.3}$$

a_{ij} is the probability that the model will transition to state S_j at time t given that the model was previously at state S_i at time t-1. Depending on the structure of the transition probabilities, an HMM model can fall into one of several categories. Some typical categories are shown in the figure below:

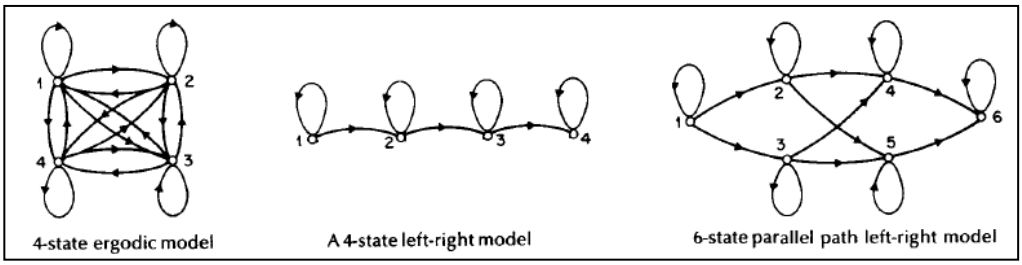


Figure 3.2 – Example of HMM Model Types **[50]**

The observation probability distribution B is defined on a per state basis,

$$b_i(\bar{Y}) = P_i(\bar{Y}) \tag{3.4}$$

Where P_i is a probability distribution for state i, which is dependent on the observation sequence $\bar{Y}$. Depending on the nature of the system being modeled, the observation probability distribution function can be configured to be one of the following: discrete probabilities, univariate Gaussian distributions, Gaussian mixture distributions, and multivariate Gaussian distributions.

3.2.2 Modeling Human Motions

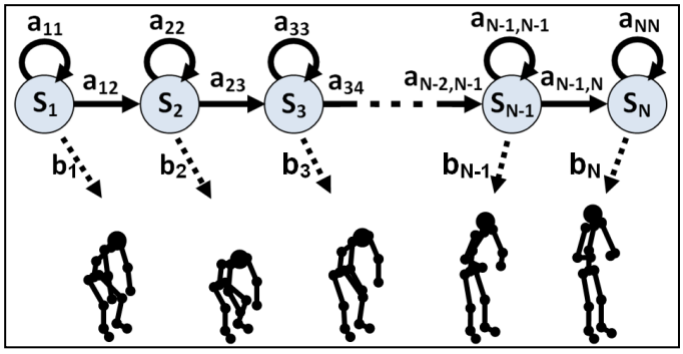


Figure 3.3 – Left-to-Right HMM and its Parameters

For the purposes of our work, we will be studying human movements that are acyclic and have no redundant states (i.e. are motion primitives). These will include tasks like lifting or reaching movements. Under these assumptions we can conclude that our HMM models will be limited to being left-to-right HMM models. This means the following:

$$\pi_0 = 1, \quad [\pi_2, \ldots \pi_N] = [0, \ldots, 0] \tag{3.5}$$

The transition matrix will only have non-zero values for elements on the diagonal and the diagonal adjacent elements to the right,

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ 0 & a_{22} & a_{23} & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} & 0 \\ 0 & 0 & 0 & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & a_{NN} \end{bmatrix} \tag{3.6}$$

For the observation probability distribution B we choose to use multivariate Gaussian distributions as this will allow us to capture variance information between states and how variables correlate to one another,

$$b_i(\bar{Y}) = \mathcal{N}_K(\mu_i, U_i) \tag{3.7}$$

b_i is the observation probability function of state S_i, $\bar{Y}$ is the observation vector generated by state S_i, and $\mathcal{N}_K$ is a K-dimensional Gaussian consisting of the mean vector μ_i and the covariance matrix U_i.

An important property that can be derived from the transition matrix for a left-to-right HMM is the state duration d_i, the length of time a HMM will stay in state i before transitioning to another state. This is estimated as:

$$d_i = 1 / (1 - a_{ii}) \tag{3.8}$$

The unit for state duration is in time steps. The above relation intuitively makes sense as the larger a_{ii} is, the larger the probability that we will remain in S_i. Typically, the state transition probability for the last state in a left-to-right model is 1, because once an HMM enters this state it remains in this state.

3.2.3 Training

To train an HMM model there are two kinds of parameters to consider, ones that need to be set before training can begin and then the parameters that need to be learned. The number of states and dimensionality of the observation sequences are specified beforehand. Parameters π, A, and B are learned during the training process and these parameters need to be initialized to appropriate values before learning. The Baum-Welch algorithm [36] is then used to train A and B from the data. These parameters will be discussed in more detail below.

Number of States

In order to train any HMM model, the number of states needs to be specified. Leave-One-Out Cross validation can be used to estimate the number of states:

```
Total Likelihood [Maximum Number of States – Minimum Number of States] = 0;
Observation Sequence Set = List of All Observation Sequences
For X = Minimum Number of States to Maximum Number of States
    For Y = 1 to Number of Observation Sequences
        Modified Observation Sequence Set = Observation Sequence Set with Observation Y excluded
        Train HMM with Modified Observation Sequence Set
        Calculate Likelihood of HMM generating Observation Sequence Y
        Total Likelihood [X] = Total Likelihood[X] + Likelihood
    End
    Total Likelihood [X] = Total Likelihood [X] / Length(Observation Sequence Set)
End
Use Total Likelihood to calculate ideal number of states based on either:
  - first inflection point (Chosen for minimizing training time)
  - slope = 0 (Chosen for ideal number of states)
```

Figure 3.4 – Pseudo-Code for Leave-One-Out Cross Validation

Running the above algorithm to generate a probability for each value for the number of states should result in the figure with a curve similar to the one shown in **Figure** 3.5.

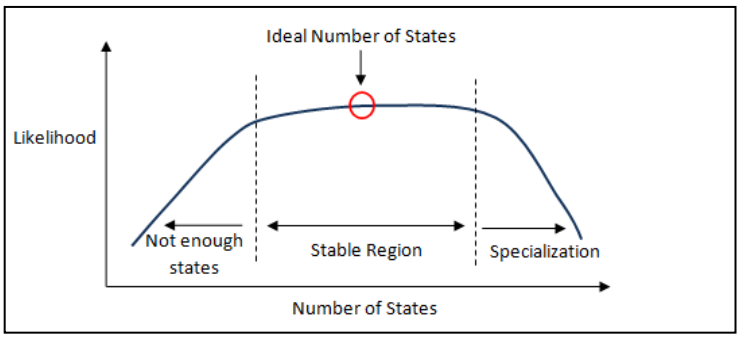


Figure 3.5 – Leave-One-Out Cross Validation to Determine Number of States

From the results of the cross-validation, it is possible to estimate the optimal number of states needed. Ideally, we would like the number of states to be in the middle of the flat region to allow for maximum stability, but usually the more states an HMM has the more computationally expensive it is to train the model. Thus, often a point that is on the plateau but closer to the first inflection point is selected.

Initial Values

Another important consideration for training an HMM model λ is to initialize the variables of π, A, B to good estimates. This will reduce training time and it also helps prevent the training algorithm from falling into a local minima.

For a left-to-right model the initialization for π is given in **Equation 3.5**. For the transition matrix A, we choose an initial condition that assumes all states have equal state duration to ensure that the movement variation is well distributed among the states. Hence all the elements in the diagonal and diagonal-adjacent are 0.5 and all other elements are set to 0.

For setting the initial values for the probability observation distributions, we used the actual observation sequence data. Assuming states of equal duration the observation sequence was split into N equal chunks (representing each state) and the means / variances were computed for each of those chunks. These means and variances were then used to define the initial values of mean vector μ_i and the covariance matrix U_i for each state i.

Baum-Welch Training Algorithm

The Baum-Welch algorithm [36] is an expectation-maximization algorithm that is used to iteratively train the model parameters to maximize the likelihood that the model generates the observation sequence. The Baum-Welch is composed of the forward algorithm, the backward algorithm, and probability calculations in order to re-estimate parameters for the model. One iteration of the Baum-Welch algorithm is shown in **Figure** 3.6.

1. Calculate the forward variable α using the forward algorithm:
   (α is the probability of the partial observation sequence $Y_1, Y_2, \ldots Y_t$ (until time t), given that the state S_i at time t, given the model λ)
   $$\alpha_T(i) = P(Y_1 Y_2 \cdots Y_T, q_T = S_i | \lambda)$$
2. Calculate the backward variable β using the backward algorithm:
   (β is the probability of the partial observation sequence from time step t+1 to the end, given the state S_i at time t and the model λ)
   $$\beta_t(i) = P(Y_{t+1} Y_{t+2} \cdots Y_T, q_t = S_i | \lambda)$$
3. Calculate the probability of being in state S_i at time t+1, given the model and the observation sequence:
   $$\xi_t(i,j) = \frac{\alpha_T(i)\, \alpha_{ij} \beta_j(Y_{t+1})\, \beta_{t+1}(j)}{P(Y|\lambda)}$$
4. Calculate the probability of being in S_i at time t, given the observation sequence of the model:
   $$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)$$
5. Calculate the expected number of transitions:
   $$\sum_{t=1}^{T-1} \gamma_t(i) = expected\ number\ of\ transitions\ from\ S_i$$
   $$\sum_{t=1}^{T-1} \xi_t(i,j) = expected\ number\ of\ transitions\ from\ S_i\ to\ S_j$$
6. Calculate the Re-estimated Model Parameters:
   $$\overline{\pi_i} = expected\ frequency\ in\ state\ S_i\ at\ time\ (t=1) = \gamma_t(i)$$
   $$\overline{a_{ij}} = \frac{expected\ number\ of\ transitions\ from\ state\ S_i\ to\ state\ S_j}{expected\ number\ of\ transitions\ from\ state\ \ S_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$
   $$\overline{b_i(k)} = \frac{expected\ number\ of\ times\ in\ state\ j\ and\ observing\ symbol\ v_k}{expected\ number\ of\ times\ in\ state\ j} = \frac{\sum_{t=1\ (st\ Y_t = v_k)}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

Figure 3.6 – One Iteration of the Baum-Welch Algorithm

With each iteration of the Baum-Welch algorithm the model estimation is expected to improve and this is measured by an increase in the probability that the model λ generated the Observation sequence Y,

$$P(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.9}$$

Iterations of the Baum-Welch typically continue until a maximum number of iterations has reached or the incremental increase in probability between iterations is below some threshold,

$$P(Y|\lambda)_{current} - P(Y|\lambda)_{previous} < threshold \tag{3.10}$$

3.2.4 Generating an Observation Trajectory

There are two main ways to generate an observation trajectory with HMM models. One way is to generate a random sequence by sampling the observation distributions in each state. The second approach is to generate a sequence deterministically by using the expected value from each observation distribution. In our work, we use the deterministic approach as we are more interested in the mean trajectory. A description of the algorithm is provided below:

```
Create a blank observation sequence OBS
For ( current_state = 1 to (Number of States -1) )
        state_duration = (1 / (1 – A(current_state,current_state) )
        state_duration = round(state_duration)
        while ( state_duration >  0 )
              state_duration  = state_duration   -  1
               Add an observation with mean of current_state to OBS
        End
End
current_state = Number of States
Add an observation with mean of current_state to OB
Return OBS
```

Figure 3.7 - Algorithm for Generating Trajectories from HMMs

The deterministic approach uses the transition matrix elements to estimate a state duration value (**Equation 3.8**). For that many time steps, the algorithm adds the mean observation of the current state to the observation sequence. This process is repeated until we reach the last state.

3.2.5 Comparing HMM Models

To compute the level of dissimilarity between any two HMM models, the Kullback Leibler (KL) distance is used [50]. This distance measure takes into account the movement progression as well as the variability of the motion and can be calculated as follows:

$$D(\lambda_1, \lambda_2) = \frac{1}{T} \left[log\ P\left(Y^{(2)} \middle| \lambda_1\right) - log\ P\left(Y^{(2)} \middle| \lambda_2\right) \right] \quad (3.11)$$

where $Y^{(2)}$ is an observation sequence generated from λ_2, $P(Y|\lambda)$ is the probability that an observation sequence Y was generated by the model λ, and T is the length of the observation sequence. An efficient algorithm for computing $P(Y|\lambda)$ is the forward algorithm [50].

The KL distance is non-symmetric [50], i.e. it only considers an observation sequence created from one model. To obtain a symmetric measure we calculate the symmetric distance D_s using the following relationship:

$$D_s(\lambda_1, \lambda_2) = \frac{D(\lambda_1,\lambda_2) + D(\lambda_2,\lambda_1)}{2} \quad (3.12)$$

3.3 Gaussian Process Dynamical Models

This section will provide the relevant background for Gaussian Process Dynamical Models (GPDM). GPDMs can be used to convert high dimensional observation sequences to a lower dimensional representation in the latent space.

3.3.1 Introduction

The GPDM model consists of two main parts: (i) the process or state which describes the lower-dimensional dynamics and (ii) the observation or measurement which is related to the latent space via a non-linear mapping. The content in this section is adapted from [4].

Assume that we have some human motion trajectory (or Observation Sequence) y_t, which consists of a sequence of D-dimensional vector-valued states with time t, and a corresponding latent space representation x_t, which consists of d-dimensional latent coordinates at time t.

$$\mathbf{Y}=[y_1, \ldots y_t, \ldots, y_N]^T \tag{3.13}$$

$$\mathbf{X}=[x_1, \ldots x_t, \ldots, x_N]^T \tag{3.14}$$

Since the goal is to model human trajectories which are generated by a biomechanical system, a suitable way to represent the second-order Markov dynamics (1) and latent variable mapping (2) would be:

$$y_t = g(x_t; \mathbf{B}) + \boldsymbol{n}_{y,t} \tag{3.15}$$

$$x_t = f(x_{t-1}, x_{t-2}; \mathbf{A}) + \boldsymbol{n}_{x,t} \tag{3.16}$$

The functions f and g are mappings parameterized by weights $\mathbf{A}=[a_1, a_2, \ldots]^T$ and $\mathbf{B}=[b_1, b_2, \ldots]^T$ and are linear combinations of nonlinear basis functions Φ_i and ψ_j

$$f(x; \boldsymbol{A}) = \sum_i a_i \Phi_i(x) \tag{3.17}$$

$$g(x; \boldsymbol{B}) = \sum_j b_j \psi_j(x) \tag{3.18}$$

Assuming an isotropic Gaussian prior on a_i and b_j and assuming the $n_{x,t}$ and $n_{y,t}$ are zero-mean isotropic white Gaussian noise processes, we get:

$$p(\boldsymbol{X}|\bar{\alpha}) = \frac{p(x_1)}{\sqrt{(2\pi^{(N-1)d}|\boldsymbol{K}_X|^d)}} exp\left(-\frac{1}{2} tr(\boldsymbol{K}_X^{-1} \boldsymbol{X}_{2:N} \boldsymbol{X}_{2:N}^T)\right) \tag{3.19}$$

$$p(\boldsymbol{Y}|\boldsymbol{X}, \bar{\beta}, \boldsymbol{W}) = \frac{|\boldsymbol{W}|^N}{\sqrt{(2\pi^{ND}|\boldsymbol{K}_Y|^D)}} exp\left(-\frac{1}{2} tr(\boldsymbol{K}_Y^{-1} \boldsymbol{Y} \boldsymbol{W}^2 \boldsymbol{Y}^T)\right) \tag{3.20}$$

Where $\bar{\alpha} = \{\alpha_1, \ldots \alpha_6\}$ and $\bar{\beta} = \{\beta_1, \beta_2\}$ are hyper parameters (described further below). $\boldsymbol{W}$ includes D scale hyper parameters ($\boldsymbol{W} \equiv$ diag($w_1, \ldots, w_D$) that account for the overall scale GPs in each data dimension – since each data dimension can differ in variability over time. $\mathbf{K}_X$, and $\mathbf{K}_Y$ are kernel

functions. Here we use RBF kernels. Because this is a second order system, the kernel matrix $\mathbf{K}_X$ depends on the current and previous latent space coordinates

$$k_X([x_t, x_{t-1}], [x_\tau, x_{\tau-1}]) =$$
$$\alpha_1 \exp\left(-\frac{\alpha_2}{2}\|x_t - x_\tau\|^2 - \frac{\alpha_3}{2}\|x_{t-1} - x_{\tau-1}\|^2\right) + \alpha_4 x_t^T x_\tau + \alpha_5 x_{t-1}^T x_{\tau-1} + \alpha_6^{-1}\delta_{t,\tau} \quad (3.21)$$

and $\mathbf{K}_y$ is as follows

$$k_Y(x, x') = exp\left(-\frac{\beta_1}{2}\|x - x'\|^2\right) + \beta_2^{-1}\delta_{x,x'} \quad (3.22)$$

In the kernal functions above:

- α_1 controls the output scale of the RBF
- α_2, α_3 represent the inverse width of the RBF terms
- α_4, α_5 controls the output scale of the linear terms
- α_6 represents the inverse variance of the process noise
- $\beta_1$represent the width of the RBF
- β_2 is the inverse of the variance of the isotropic noise

3.3.2 Training

Given a set of observations Y,

$$Y = \{y_1, \dots, y_N\} \quad (3.23)$$

The Maximum-A-Posteriori (MAP) algorithm is used to estimate the latent trajectories and hyper-parameters. In this approach the goal of the learning algorithm is to minimize the joint negative log-posterior of the unknowns that is given by:

$$L = L_Y + L_X + \sum_j ln\beta_j + \frac{1}{2\kappa^2} tr(\boldsymbol{W}^2) + \sum_j ln\alpha_j \quad (3.24)$$

Where

$$L_Y = \frac{D}{2} ln|\boldsymbol{K}_Y| + \frac{1}{2} tr(\boldsymbol{K}_Y^{-1}\boldsymbol{Y}\boldsymbol{W}^2\boldsymbol{Y}^T) - Nln|\boldsymbol{W}| \quad (3.25)$$

$$L_X = \frac{d}{2} ln|\boldsymbol{K}_X| + \frac{1}{2} tr(\boldsymbol{K}_X^{-1}\boldsymbol{X}_{2:N}\boldsymbol{X}_{2:N}^T) + \frac{1}{2} x_1^T x_1 \quad (3.26)$$

3.3.3 Generating an Observation Trajectory

In order to generate latent trajectories from the model, fair samples of entire trajectories are drawn using a Markov chain Monte Carlo Sampler. The Markov chain is initialized with a sequence consisting of the expected value, and then is generated from $x_{t-1}^{(*)}$by simulating the dynamical process one frame at a time.

$$x_t^* \sim N\left(\mu_X\left(x_{t-1}^{(*)}\right), \sigma_X^2\left(X_{t-1}^{(*)}\right) I\right) \quad (3.27)$$

$$\mu_X(x) = \boldsymbol{X}_{2:N}^T \boldsymbol{K}_X^{-1} \boldsymbol{k}_X(x) \quad (3.28)$$

$$\sigma_X^2(x) = k_X(x,x) - \boldsymbol{k}_X(x)^T \boldsymbol{K}_X^{-1} \boldsymbol{k}_X(x) \quad (3.29)$$

Here the i^{th} entry in the vector $k_x(x)$ is $k_x(x,x_i)$, and x_i is the i^{th} training vector. At each step of mean prediction, we set the latent position to be the mean latent position conditioned on the previous state.

In order to reconstruct a trajectory in the original high dimensional domain we use the following relationships:

$$\mu_Y(x) = \boldsymbol{Y}^T \boldsymbol{K}_Y^{-1} \boldsymbol{k}_Y(x) \quad (3.30)$$

$$\sigma_Y^2(x) = k_Y(x,x) - \boldsymbol{k}_Y(x)^T \boldsymbol{K}_Y^{-1} \boldsymbol{k}_Y(x) \quad (3.31)$$

3.4 Comparison Algorithms

Our proposed approach focuses on understanding differences between motions and furthermore explaining how groups of joints (or degrees-of-freedom) move in coordination. To evaluate the effectiveness of our approach we compare our proposed approach against two approaches that have similar goals: Joint Component Vector (JCV) [5] and Kinematic Joint Synergies [6]. In this section we will introduce these comparison algorithms.

3.4.1 JCV Algorithm

3.4.1.1 Algorithm Overview

The Joint Contribution Vector (JCV) was introduced by Park et al [5] as a quantitative index for representing motion in terms of the individual contributions of each DOF to the achievement of a task goal. Given a set of motion data, each motion is abstracted to a JCV vector and then K-means clustering is used to cluster these vectors to determine strategies. **Figure** 3.8 describes the JCV algorithm.

1. For each motion described by the trajectory Y(t) do the following:
    a. For each DOF j in the motion where $Y(t) = [y_1(t) \dots y_j(t) \dots y_J(t)]^T$ do the following:
        i. Set $y_j(t) = y_j(0)$ for the interval [0,T], where T is at the end of the trajectory
        ii. Using a kinematic model of the participant compute the end effector Cartesian coordinates for each time step using the modified vector Y(t), and calculate the Contribution (C) and Percent Contribution (PC) of DOF j on the end effector Cartesian coordinates.
    b. Aggregate the JCV vectors for motion i into the final JCV vector
2. Perform K-means Clustering on the JCV Vector
    a. If the number of motions are known a-priori use this for K
    b. If the number of motions are not known a-priori use Multi-Dimensional Scaling (MDS) to visually determine the number of clusters and then use this for K

Figure 3.8 – JCV Algorithm

3.4.1.2 Joint Component Vector

To compute the JCV vector for a specific motion we begin with a trajectory θ(t) described as

$$Y(t) = [y_1(t) \dots y_j(t) \dots y_J(t)]^T \tag{3.32}$$

Where j = 1, 2, … J is an index for the j^{th} DOF and t represents the time on the interval [0, T]. The contribution of the i^{th} DOF to the motion trajectory can be calculated by comparing the full motion Y(t) with a similar motion $Y^i(t)$ where the effect of i^{th} DOF is eliminated. Since the end-effector trajectory in the task space can be directly related to the achievement of a task goal, the relative contributions of the i^{th} DOF can be analyzed by the difference in the task space when that DOF is eliminated (held at its initial value). Therefore $Y^i(t)$ is defined such that $Y^i_j(t) = Y_j(t)$ if $j \neq i$ and $Y^i_j(t) = Y_j(0)$ if j=i for all t. To calculate the sum of the contributions of the individual DOFs, the following relationships are used:

$$C_x^i = \int_{t=0}^{T} \left(x(t) - x^i(t)\right) dt, \tag{3.33}$$

$$C_y^i = \int_{t=0}^{T} \left(y(t) - y^i(t)\right) dt, \tag{3.34}$$

$$C_z^i = \int_{t=0}^{T} \left(z(t) - z^i(t)\right) dt, \tag{3.35}$$

These individual contributions of each DOF are then normalized using the total contributions of all the joints for the motion such that we get the normalized x, y, z contribution (or Percent Contribution):

$$PC_x^i = 100\frac{C_x^i}{\sum_{j=1}^{J}|C_x^j|} \tag{3.36}$$

$$PC_y^i = 100\ \frac{C_y^i}{\sum_{j=1}^{J}|C_y^j|} \tag{3.37}$$

$$PC_z^i = 100\ \frac{C_z^i}{\sum_{j=1}^{J}|C_z^j|} \tag{3.38}$$

Three J-element vectors are formed using the 3 PC values for each DOF, such that

$$JCV_x = \left[PC_x^i \ldots PC_x^j\right] \tag{3.39}$$

$$JCV_y = [PC_y^i \ldots PC_y^j] \tag{3.40}$$

$$JCV_z = [PC_z^i \ldots PC_z^j] \tag{3.41}$$

Then in the final step the JCV vector can be created by combining the three j-element vectors, such that:

$$\mathrm{JCV} = [\mathrm{JCV_x}\ \ \mathrm{JCV_y}\ \ \mathrm{JCV_z}] \tag{3.42}$$

The JCV vector is computed for each motion and then K-means clustering is used to determine the motion strategy. As mentioned previously, if the number of motion strategies is known a-priori then k-means clustering can be used to classify motions that belong to one of the known categories. If the number of clusters is not known, then multidimensional scaling can be used to visualize the data and manually specify the number of clusters. Once the number of clusters is specified, the k-means algorithm is used to classify the motion sequences into strategies.

3.4.1.3 Analyzing Joint Contributions

The PC values that are used to create the JCV vectors can also be used for visually representing which DOF is contributing the most for a specific type of motion. By creating box-plots using the PC values across all motion sequences for a specific joint, it is possible to look at the mean and spread of the range of PC values. This information could then also be potentially used to analyze the differences between various motions to discern which joint is most important in a specific motion strategy. The difference for a specific joint between two strategies could be calculated by taking the absolute of the delta between mean PC values for the same joint across two strategies.

3.4.1.4 Algorithm properties

One of the key things to consider when using this algorithm is the definition of the model. The way the model is used results in various restrictions and requirements:

1. Anthropometric data of participants must be collected so that accurate models can be created
2. A different model must be created for each type of data-set if special assumptions such as motion symmetry are used to simplify the model
3. The models must be open-link kinematic structures, which may not be suitable for all motion types
4. Joint angle data is a requirement for this algorithm as the end effector coordinates are calculated using forward kinematics and joint angles. A second key consideration is the determination of the number of clusters. Park et al [5] use two approaches to create clusters:
    - The number of strategies are known a-priori, this number is set as the 'K' in K-means
    - The number of strategies are determined visually using Multi-Dimensional Scaling

Both of these approaches are non-ideal and it means the process of detecting the number of strategies cannot be fully automated using these approaches.

3.4.2 Kinematic Joint Synergy Detection

3.4.2.1 Algorithm overview

The theory of motion synergies assumes that the central nervous system does not control individual muscles but unites them in groups to control them in a synergy. Therefore, the number of control variables is not necessarily equal to the number of muscles involved in the movement but is a reduced number of variables depending on the synergy. The concept of kinematic joint synergy is an extension of muscle synergy theory that assumes that groups of joints (or degrees-of-freedom) can have the same kind of coordinated movements [67].

One of the most common ways to detect synergies from a time series dataset is to use matrix factorization. Matrix Factorization is the process of taking a t by m matrix R and finding two smaller matrices P (size t by n) and Q (size n by m) such that their product approximates R.

$$R = P \times Q + E \tag{3.43}$$

Here P is known as the feature matrix (containing the latent feature), Q is known as the weights matrix, and E is the unexplained variation. In this process adjusting n allows for more accuracy when

reconstructing the original matrix. Non-negative matrix factorization is a further specialization of matrix factorization technique that requires that weights have non-negative values (i.e. ≥ 0). This constraint makes the optimization problem convex and ensures that the outcomes are a true global minima.

In the use case of synergy detection, the matrix R will contain time-series data representing the human movement, where m is the number of motion effectors (muscle EMG's or joint angles) and t is the number of time steps. The matrix Q is the synergy matrix consisting of n synergy vectors (with dimensions of 1 by m). The matrix P is the activation matrix, which contains information about the degree of activation for each synergy group over each time step. The figure below provides an overview of the gradient descent algorithm that is used to determine P and Q values for a given value of n (fixed number of synergies):

1. P and Q are initialized with small random numbers
2. For each step until **max number of iterations**:
    a. For each row and column in R
        i. If R[row][column] > 0:
            1. Compute the predicted element

$$\hat{r}_{ij} = p_i^T q_j = \sum_{n=1}^{n} p_{in} q_{nj}$$

            2. Compute the squared error for each pair

$$e_{ij}^2 = (r_{ij} - \hat{r}_{ij})^2 = (r_{ij} - \sum_{n=1}^{n} p_{in} q_{nj})^2$$

            3. Compute gradient by taking the differential of the error

$$\frac{\partial}{\partial p_{in}} e_{ij}^2 = -2\left(r_{ij} - \hat{r}_{ij}\right)\left(q_{nj}\right) = -2e_{ij} q_{nj}$$

$$\frac{\partial}{\partial q_{in}} e_{ij}^2 = -2\left(r_{ij} - \hat{r}_{ij}\right)(p_{in}) = -2e_{ij} q_{in}$$

            4. Update P and Q at a rate proportional to the learning rate α

$$p'_{in} = p_{in} + \alpha \frac{\partial}{\partial p_{in}} e_{ij}^2 = p_{in} + 2\alpha e_{ij} q_{nj}$$

$$q'_{nj} = q_{nj} + \alpha \frac{\partial}{\partial q_{nj}} e_{ij}^2 = p_{in} + 2\alpha e_{ij} p_{in}$$

        ii. Compute total error

$$E = \sum_{(p_i, q_j, r_{ij})} (r_{ij} - \sum_{n=1}^{n} p_{in} q_{nj})^2$$

        iii. If error is less than error threshold, then break
    b. Return matrices P and Q

Figure 3.9 – Gradient Descent Algorithm for NNMF

The number of synergies m are determined in an iterative manner using a metric called Variance Accounted For (VAF) to ensure that the appropriate amount of variance is captured by the synergies:

$$VAF = 100 \times (1 - \frac{SSE}{SST}) \tag{3.44}$$

Where SSE is the sum of the squared residuals and SST is the sum of the squared EMG or joint data. This requirement is that a global VAF (across the entire data set) of 90% be achieved and that iteratively increasing the amount of synergies results in less than 5% increase in the global VAF (i.e. diminishing returns). Additionally, there is a local VAF requirement that each EMG or joint VAF must exceed 50% to ensure that any one time series data set is modeled relatively well.

Once the Synergy matrix Q has been discovered, then the next step is to discover groups of movement synergies (i.e. movement strategies) using clustering algorithms. Subtractive clustering is used to discover the number of strategy clusters and then k-means is used to do the actual clustering. The result is a set of synergy clusters in synergy vector space.

In order to classify new data points, the NNMF algorithm needs to be re-run with the new data points and then clustering has to happen again. This means there may be no guarantee that the newly generated clusters will be the same as the previous ones. Initialization of cluster centers can help mitigate this issue as long as the new data points have a similar distribution to the old data points.

3.4.2.2 Analyzing Kinematic Motion Synergies

Kinematic motion synergies are typically analyzed by plotting the synergy vectors and the activation vectors as bar graphs comparing them against each other. In particular, the dominant synergies are of interest as they show how a movement pattern utilized muscles or joints to accomplish the same task in a different way.

3.4.2.3 Implementation Considerations

In order to use the NNMF algorithm described in this section, some pre-processing was done on the dataset. Ideally for NNMF an equal number of time series data points should be defined for all the EMG and/or joint dataset sequences. In order to accomplish this, the following was done:

1. Input data (joint angle or Cartesian data) was normalized between 0 to 1 across the entire dataset.
2. One HMM model with each observation sequence that had 10 states was trained (number of states determined through the algorithm presented in chapter 3)

3. Then the mean vectors for each state were extracted and concatenated together to create a mean matrix that had mean values across all the states. Here the state will correspond to the timestep t.

Chapter 4
Proposed Framework

4.1 Purpose and Overview

The purpose of our framework is to analyze human movement data and determine the variations (or movement strategies) that participants use to perform tasks. Additionally, the framework aims to provide insights that explain differences between the movement strategies and a way to describe those strategies with low-dimensional strategy descriptor parameters (the latent space variables). The strategy descriptors will be related back to variations in the joints to assist with interpretation. This proposed framework utilizes GPDM and HMM based algorithms. **Figure** 4.1 provides a high-level overview of the framework.

The following sections will describe the framework in more detail. The input is a high dimensional time series dataset. The trajectories are first converted to latent space trajectories with GPDM. Next, we use the GPDM-HMM Divisive clustering algorithm to cluster the trajectories, by converting the latent space trajectories into HMM models, performing HMM divisive clustering, and then generating a new latent space for the sub-clusters. The result of repeating this divisive clustering process on the sub-clusters is a motion tree that organizes the movements into a hierarchy. In addition to the motion hierarchy, the latent variables for the subspace at each node allow us to determine if there are relationships that exist between sub-motions that may be continuous or discrete in nature. Once the motion tree is constructed, the next steps focus on analyzing the movement models to extract information about the movement strategies, to determine the differences between the movement strategies and mapping those changes to the latent variables.

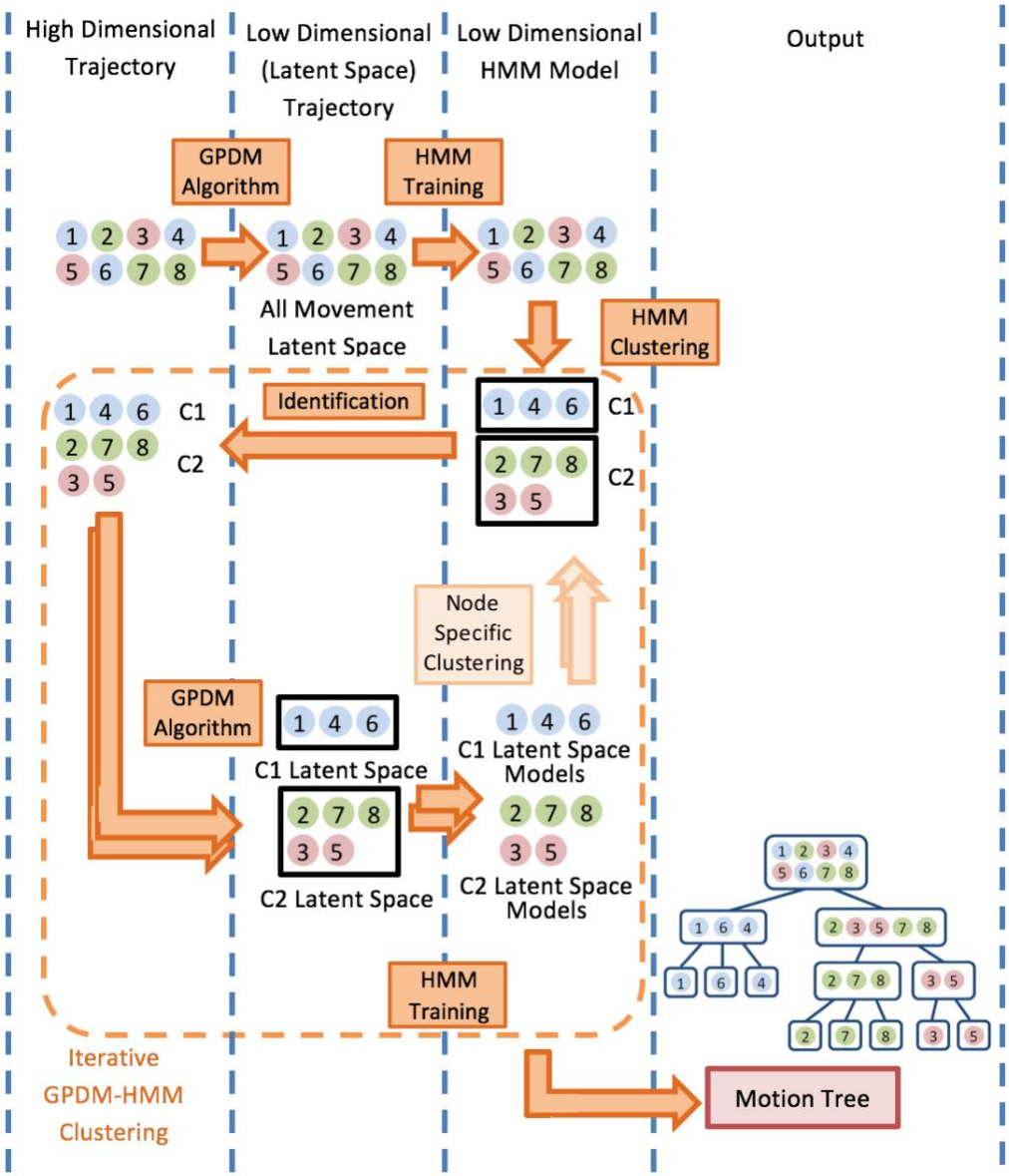


Figure 4.1 – Overview of Proposed Framework

4.2 Input: High Dimensional Time Series Dataset

The input is a dataset consisting of high dimensional observation trajectories for performing a specific task. This input data can theoretically be any kind of time series data including but not limited to cartesian coordinate data, joint angle data, or EMG data. For illustrating how the proposed approach works, we will focus on cartesian or joint angle data. One observation trajectory will typically consist

of one exemplar movement performed by a participant. The entire dataset will then consist of multiple observation trajectories captured for multiple participants performing some prescribed acyclic task.

In the case of cartesian data, the observation sequence data will consist of a multivariate time-series dataset for the x-y-z coordinates of each motion capture maker. Note that groups of time series data for a specific marker (like for x, y, and z for the left knee) will be referred to as joint positions.

Alternatively, we can also consider using joint angle data for analysis. Joint angle data is typically generated by converting the cartesian coordinate data to joint angle space by using inverse kinematics. In the case of joint angle data, groups of angles like θ-ϕ-φ can also be referred to as joint positions.

4.3 Latent Space Trajectories with GPDM

The GPDM training algorithm takes the high dimensional trajectories and converts them to a lower dimensional space (i.e. the latent space) where the dynamics of the motions are preserved. The algorithm as defined by Jack et al [4], requires the dimensionality d of the latent space to be specified as an input to the GPDM algorithm. We propose automating the selection of the latent space dimensionality through an iterative process.

The iterations start from a value of d=2 and incrementally increase by 1 in each iteration. During each iteration, the observation trajectories are re-generated from the GPDM model and the reconstruction error is computed for each observation sequence. If the rate of change in the average reconstruction error falls below a pre-specified threshold, then the algorithm terminates. The dimensionality of the last iteration is considered the optimal value.

4.4 Clustering Latent Space Trajectories

4.4.1 Overview

The latent variable trajectories are inputs to the next stage: the GPDM-HMM Clustering algorithm. The goal of the algorithm is two-fold:

(i) Group similar observation trajectories so that it is possible to identify movement strategies.

(ii) Determine if there are any latent parameters that can help define relationships between movements in a continuous or discrete domain.

The result of the clustering algorithm is a tree (see **Figure** 4.2), where the root node of the tree contains all observation trajectories. The root node then splits into 2 or more nodes, where each node contains similar observation trajectories and is considered a "motion strategy." As one proceeds further

down the tree towards the leaf nodes, more specialized sub-strategies will be observed until reaching the leaf nodes, which consist of individual observation trajectories.

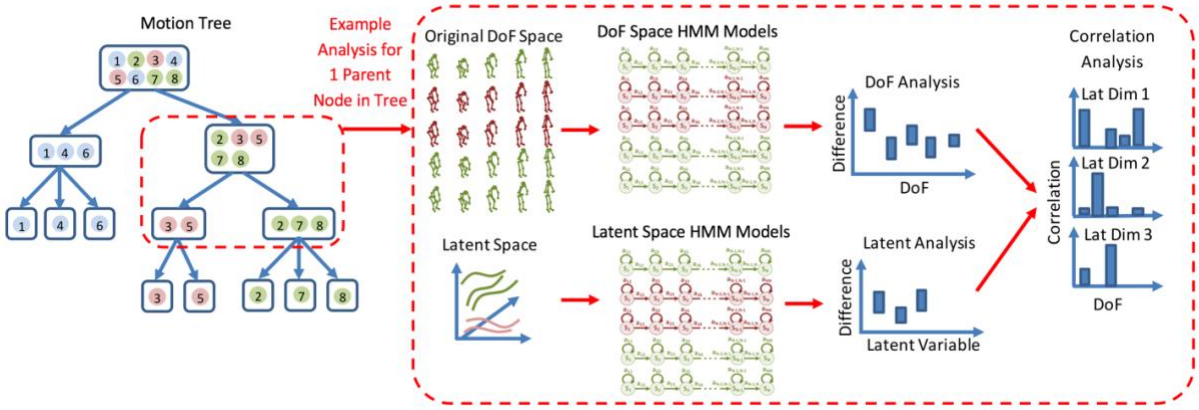


Figure 4.2 – Example Clustering Results and Analysis

Each node in the tree has the following:

a) A latent space that was trained using only the trajectories in that node
b) Low dimensional latent space HMM models:
    i) One model that is trained with all latent space trajectories in that node
    ii) One model for each latent space trajectory
c) High dimensional latent space HMM models:
    i) One model that is trained with all original observation trajectories in that node
    ii) One model for each original observation trajectory

By using data in the parent node and sub-strategy information from the child-nodes, we do a correlation analysis between impact sub-strategies have on DoF and on the latent variables. This provides us with a means to understand how changes in latent variables map to changes in DoF.

The process for generating the motion tree is shown visually as the orange box in **Figure** 4.1. The pseudo-code for this is presented in **Figure** 4.3.

1. Model the latent trajectories for the incoming node N_{parent} with one HMM λ_G
2. Use the HMM divisive clustering algorithm to cluster the latent trajectories into *n* sub-strategies. The result of this process is
   a. Sub-strategy grouping of trajectories in the parent node's latent space
   b. Generating child nodes $N_0 \ldots N_n$, each with a subset of observation trajectories
3. For each child node $\{N_0 \ldots N_n\}$
   a. Train a GPDM model using only the trajectories in the child node
   b. The result is latent space trajectories in the child node's latent space
4. The trajectories in the child node latent spaces can then be again divisively clustered using HMM divisive clustering.

Figure 4.3 – GPDM-HMM Divisive Clustering Algorithm Overview

4.4.2 HMM Divisive Clustering Algorithm

The proposed divisive clustering algorithm [2] starts with one general model λ_G at the root, trained using all observation sequences. This general model is then divided into strategy-specific models through hierarchical clustering of the observation sequences in the likelihood space. The algorithm iteratively improves upon each cluster model by using the highest likelihood to re-allocate sequences to the best cluster and to retrain the model. The result is a motion tree in which non-leaf nodes represent strategy clusters and HMM models at each node can be compared to determine the differences between motion strategies. A general overview of the algorithm is provided in **Figure** 4.4.

1. Initialize one HMM λ_G
2. Train a general HMM model λ_G with all N observation sequences using Baum-Welch
3. Compute: $p = [P(Y_1 \mid \lambda_G) , P(Y_2 \mid \lambda_G) , \ldots , P(Y_N \mid \lambda_G)]$
4. Cluster p to create membership matrix M:
   a. Use subtractive clustering to determine optimum number of clusters C in p
   b. Initialize the C-means clustering algorithm use results from the subtractive clustering algorithm
   c. Use C-means clustering to determine the membership matrix M of size N x C
   d. Train models $\lambda_{cluster\ i}$ for $0 < i < C$ with those Y_i that have the greatest membership in cluster i.
5. While there are no new Y_i that have be introduced into any cluster i
   a. For i = 1 to C compute:
      $q = [P(Y_1 \mid \lambda_{cluster\ i}) , P(Y_2 \mid \lambda_{cluster\ i}) , \ldots , P(Y_N \mid \lambda_{cluster\ i})]$
   b. Reallocate sequences to $\lambda_{cluster\ i,}$ for which they have the highest probability
6. Calculate the distance matrix D for cluster models
7. merge threshold = minimum distance in D * merge factor
8. While no distance in D < merge threshold
   a. Merge two clusters if the distance between the clusters is below the merge threshold
   b. Re-train cluster models
   c. Recalculate the distance matrix D for cluster models

Figure 4.4 – Proposed iterative divisive clustering algorithm

Determining the number of clusters for any dataset is perhaps of the most challenging aspect of clustering data accurately [37]. We use subtractive clustering to predict the optimal number of clusters as proposed in [72] in addition to determining the location of the cluster centers (as shown in step 2-a). Subtractive clustering initiates by assuming that each data point can be a cluster center and the measure of potential for each cluster point x_i is defined as:

$$\mathbf{P_i} = \sum_{j=0}^{n} e^{-\alpha||x_i - x_j||^2} \quad \text{and} \quad \alpha = \frac{4}{r_a^2} \tag{4.1}$$

Where r_a defines the effective radius of the surrounding neighborhood and n is the number of points. Once the potentials are computed for all the points then $\boldsymbol{x_1^*}$ is selected as the point with the highest potential $\boldsymbol{P_1^*}$. Then the potential at each point is revised by subtracting the potential of the first point from all the other points. The revised potential $\boldsymbol{P_i^*}$ for each point $\boldsymbol{x_i^*}$ is calculated as:

$$\boldsymbol{P_i} = \boldsymbol{P_i} - \boldsymbol{P_k^* e^{-\beta||x_i - x_k||^2}} \quad \text{and} \quad \boldsymbol{\beta} = \frac{4}{r_b^2} \tag{4.2}$$

Where k = 1, for the first cluster, and r_b defines the radius that will have measurable reductions in the potential. r_b is chosen to be somewhat larger than r_a so that the cluster centers are not too closely spaced. Once the potentials have been revised a new candidate cluster center is selected and the process continues until all possible clusters have been identified and the following condition on potentials is met:

$$\boldsymbol{P_k^* < \varepsilon P_1^*} \tag{4.3}$$

Determining a value of ε can be challenging so we use the methodology proposed by Chiu [72] that allows for more accurate selection of cluster centers by providing a trade-off between having sufficient potential and the centers being sufficiently far from existing cluster centers.

Using the number of clusters detected by the subtractive clustering algorithm to initialize the C-means algorithm [73], we use the C-means to perform accurate clustering of the motion sequences. This algorithm is very similar to the K-means algorithm where the difference is that the membership matrix U in K-means is binary while in C-Means the membership U is continuous between 0 and 1:

$$\sum_i^c u_{ij} = 1 \quad \forall\, j = 1, \ldots, n \tag{4.4}$$

where u_{ij} is the membership value for the i^{th} cluster and j^{th} observation sequence number with c clusters. Next the cluster centers are calculated using:

$$c_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{4.5}$$

Where c_i is the cluster center and x_j is the j^{th} data point and m is the weighting exponent that is used to set fuzziness of the membership matrix. A value of 1 creates hard clusters and as $m \to \infty$ tends to defocus

the membership towards the fuzziest state. We set m=1 to form hard clusters. The membership matrix is updated with the following relationship:

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{||x_k - v_i||}{||x_k - v_j||}\right)^{2/(m-1)}} \tag{4.6}$$

In the next step the cost function is evaluated:

$$J_m(U, v) = \sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ik})^m ||x_k - v_i||_A^2 \tag{4.7}$$

Where v_i is the center of cluster i. If the cost function is above a certain threshold, then we continue with another iteration of the clustering be evaluating the equations again.

Once the c-means algorithm has defined the clusters, the clusters are iteratively improved by calculating the likelihood that each sequence was generated by one of the strategies. Sequences with the highest likelihood of belonging to a cluster will be re-allocated to that model and the model will be re-trained. This process continues until no more relocation of sequences is required or the maximum number of iterations has been reached.

4.4.3 Latent Spaces for Sub-Clusters

Once divisive clustering of the motions is completed, a set of a sub-strategies (or child nodes) for all motions (parent node) has been identified. Recall that the parent node consists of trajectories in a latent space generated by training a GPDM model with all motions. Therefore, the latent variables (in the parent node latent space) can help determine relationships between the all the sequences. If we want to analyze the sub-strategies of motions at the child node level, it is possible that latent variables that were important for distinguishing motions at the parent node level aren't necessarily relevant at the child node level. Therefore, we propose that a new latent space be learned for each of the child nodes based on the observation trajectories associated with the child node. This approach provides a couple of key benefits:

1. The GPDM algorithm training for latent spaces at higher levels of the tree can be more generic. Hence, when the GPDM algorithm is being trained with observation trajectories with large number of time samples, they can be sub-sampled and additionally we can reduce the number of iterations needed for training. For nodes at levels of the tree where there are fewer observation sequences, we can reduce the sub-sampling and increase the number of iterations for training.
2. By learning a latent space for each node, it is possible to train GPDM models that have lower reconstruction error in comparison to only learning one global GPDM model that has been trained on all the observation sequences.

3. It is possible to identify latent space variables that are pertinent to only the trajectories in each node, without being impacted by other trajectories that are outside of the node.

4.4.4 Parameter Considerations

An important consideration is the selection of the parameters for the subtractive clustering algorithm. To tune these parameters we used a synthetic dataset of three gaussian distributions (100 samples per distribution). For the dataset we varied the mean of the gaussian distributions and the distance between the means while leaving the variance constant between the distributions. Since the mean values were known apriori we could use them to tune the subtraction clustering parameters to obtain the correct values. Additionally, we accounted for scenarios where gaussian distributions overlapped within a pre-specified margin of error (such that it looked like one distribution) and under those scenarios we would assume it was one cluster. This algorithm was only tuned once for the experiments described in this thesis.

For our experiments we used 15 states for the models, determined through a leave-one-out-cross validation process. For details please refer to **Section 3.2** (background on HMM).

It was discovered through experimentation that setting a high covariance limit, i.e. limiting the covariance values of the HMM models, during the training process yielded clustering of higher accuracy. Please see **Section 5.2.1** for details.

4.5 Correlating Latent Variables to Changes in Observation Sequences for GPDM

To correlate the variance in the latent variables with the variance of the signals in the high dimensional observation trajectories, we use the Pearson correlation coefficient,

$$Correl(X,Y) = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sqrt{\sum(x-\bar{x})^2 \sum(y-\bar{y})^2}} \tag{4.8}$$

Where X and Y are the two variables being correlated. $\bar{x}$ and $\bar{y}$ are the average values of all data points in X and Y.

4.6 Motion Analysis using HMM Models

4.6.1 Overview

In this section we present our proposed techniques to compare HMM models to determine the differences between them. All of these approaches are an extension of the Kulback-Liebler distance

introduced in Chapter 3. We introduce the excluded-DoF and excluded-joint analysis for determining what DoF or joints are different between two different HMM models. Then these techniques are extended to correlate changes in the latent space variables to changes in the joint trajectories.

4.6.2 Excluded DoF Analysis: DoF Comparison of HMM Models

To compare the difference in a DOF between two motion models λ_1 and λ_2, the information regarding that DOF is removed from the mean vectors μ and covariance matrices U in the observation distribution function B. Then the distance between λ_1 and λ_2 is computed with the DOF excluded. This is repeated for all the DOFs in λ_1 and λ_2. When comparing these distances, the excluded DOF that results in the smallest distance is the DOF that separates the two motions the most. Let us now formalize the procedure.

Given a set of T trained models denoted as:

$$\lambda = \{ \lambda_1, \lambda_2, \lambda_3, \ldots \lambda_T \} \tag{4.9}$$

For each model we have training observation sequences (i.e. recorded data) for F number of DOFs:

$$Y = \{ y_1, y_2, y_3, \ldots y_F \} \tag{4.10}$$

Then each model λ_i is a function of O such that:

$$\lambda_i (y_i) = \lambda_i (\{y_{i1}, y_{i2}, y_{i3}, \ldots y_{iF} \}) \tag{4.11}$$

Next the DOF e is excluded from the model such that λ_i becomes the model $\lambda_{i,e}$ with excluded DoF e:

$$\lambda_{i,e} (y_{ie}) = \lambda_i (\{y_{i1}, y_{i2}, \ldots y_{i,(e-1)}, y_{i,(e+1)} \ldots y_{iF} \}) \tag{4.12}$$

Note that the DOF information is excluded by removing the e^{th} element in all μ vectors and removing the e^{th} rows and columns from all U matrices that constitute B for the model λ_i.

Now we can calculate the distance vector D_E comparing excluded DOF models for λ_i and λ_j as:

$$D_E(\lambda_i, \lambda_j) = \begin{pmatrix} D_s(\lambda_{i1}, \lambda_{j1}) \\ D_s(\lambda_{i2}, \lambda_{j2}) \\ \cdots \\ D_s(\lambda_{iF}, \lambda_{jF}) \end{pmatrix} = \begin{pmatrix} d_{ij1} \\ d_{ij2} \\ d_{ijF} \end{pmatrix} \tag{4.13}$$

The next step is to order the values in the vector $D_E(\lambda_1,\lambda_2)$ by magnitude, where the smallest distance number corresponds to the DOF that most impacts the difference between the two motions and the largest distance number is associated with the DOF that least impacts the difference.

4.6.3 Excluded Joint Analysis: Multi-DoF (Joint) Comparison of HMM Models

The method in the previous sub-section analyzes the similarity of individual DOFs between motions, but we may also be interested in analyzing motions at the joint level for those joints which are

multi DOF, such as the hip or shoulder. In this case the procedure is modified to exclude sets of 3 DOF. We define a joint as a set of 3 DOF (consisting of either x, y, z in Cartesian coordinates or θ, φ, ψ in Cardan joint angles).

Equation 4.12 is modified such that when a set s is excluded (where $0 \leq s \leq F/3 - 1$) from the motion λ_i it becomes the excluded set model $\lambda_{i,s}$:

$$\lambda_{i,s}\left(y_{i,(s*3+1)}\, y_{i,(s*3+2)}\, y_{i,(s*3+2)}\right) = \lambda_i\left(\{y_{i1}, y_{i2}, \ldots y_{i,(s*3)}, y_{i,(s*3+4)} \ldots y_{iF}\}\right) \quad (4.14)$$

Then equation D_E vector in **Equation 4.13** is modified to account for computation of F/3 distance calculations instead of F distance calculations:

$$D_E(\lambda_i, \lambda_j) = \begin{pmatrix} D_s(\lambda_{i1}, \lambda_{j1}) \\ D_s(\lambda_{i2}, \lambda_{j2}) \\ \cdots \\ D_s(\lambda_{i(\frac{F}{3})}, \lambda_{j(\frac{F}{3})}) \end{pmatrix} = \begin{pmatrix} d_{ij1} \\ d_{ij2} \\ d_{ij(\frac{F}{3})} \end{pmatrix} \quad (4.15)$$

4.6.4 Correlating Changes in Latent Variables to Changes in Joints

While using the Pearson correlation coefficient is useful to determine how a latent variable impacts the observation trajectory, this can become cumbersome if the dimensionality D of the observation sequence is high. We propose further reducing the dimensionality of D by using logical grouping of scalar trajectories. For example, the x, y, z coordinates for a motion capture marker may be associated with a single joint position and so we may be interested in the impact on the joint position as opposed to the individual scalar components of the position of that joint.

This simplification can be done by using the "Excluded-DoF" or "Excluded-Joint" analysis. First, we train an HMM model for each high dimensional observation trajectory and then do excluded-joint analysis between each pair of observation trajectories. Next, we train an HMM model for each low dimensional (or latent space) trajectory and then do Excluded-Variable analysis (note that in this context we replace be Excluded-DoF with Excluded-Variable since the concept of DoF maps to latent variable in DoF space). With these two datasets generated, a correlation analysis is done between them to determine the relationship between latent space variables and joints.

4.7 Visualization Strategies

In the case where there is interest in analyzing the impact of 2 (or max 3) latent space variables and motion strategies, a visualization technique can be used to analyze the strategy detection results. With the label information generated from the HMM clustering results, we can classify all the data points

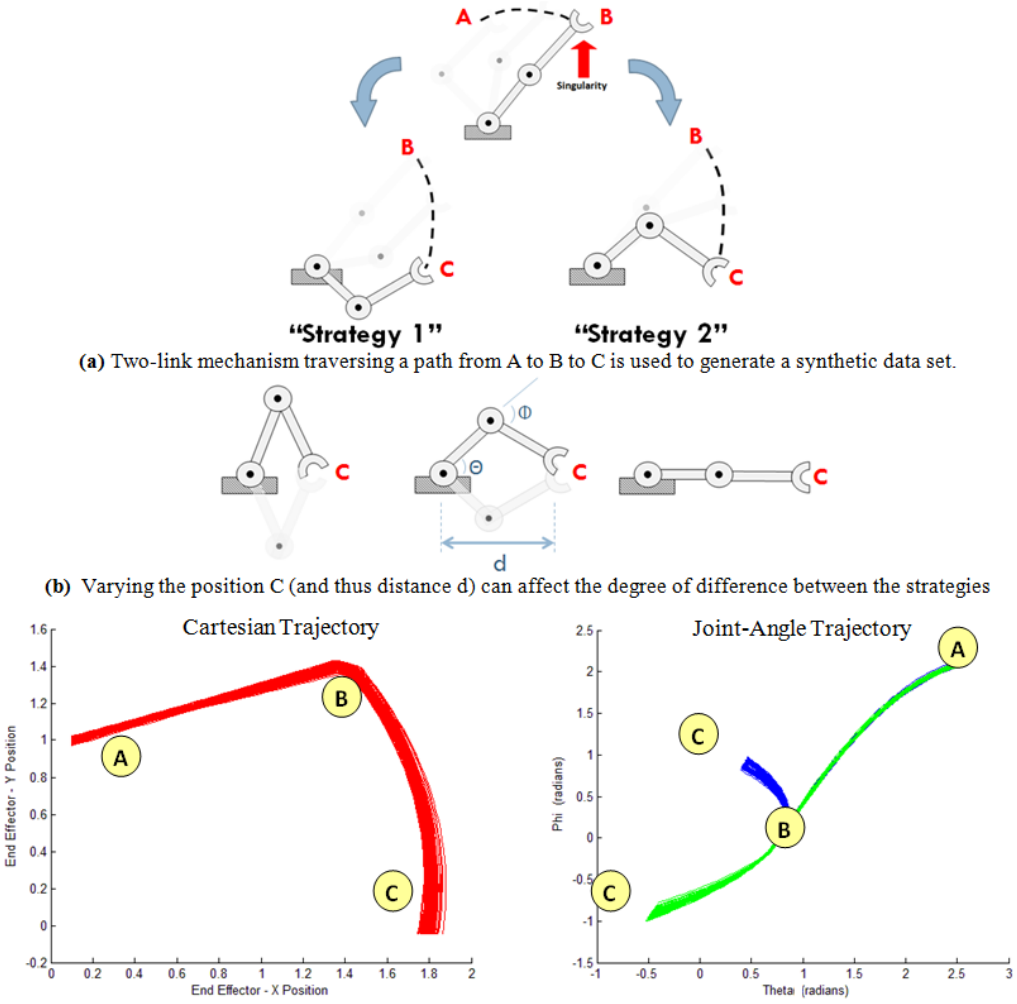
(and trajectories) in the latent space. Then using an algorithm like decision trees or support vector machines (for linear separation or data), it is possible to illustrate the boundaries between trajectories in the latent space. Coupled with added figures that represent specific poses at the extremes of the latent space or at the boundaries, it is possible to automatically create visualizations of the detected strategies. The diagrams can be easily created by reconstructing specific states in the latent space trajectory into the high dimensional space and then generating a figure depicting that state (or pose) using an assumed kinematic model.

Chapter 5
Experiments and Validation

5.1 Synthetic Dataset

5.1.1 Experiment Setup

In order to fist test the proposed framework with known labels and class distributions, a synthetic dataset was generated where the motion strategies and noise could be systematically varied. The synthetic dataset was created by simulating a two-link arm following a predefined trajectory from position A to B to C, as shown in the figure below.



(a) Two-link mechanism traversing a path from A to B to C is used to generate a synthetic data set.

(b) Varying the position C (and thus distance d) can affect the degree of difference between the strategies

(c) Sample Cartesian and joint angle trajectories for two-link mechanism

Figure 5.1 – Configurations of two-link mechanism

The two-link mechanism initially moves from position A to position B, where it encounters a singularity (trajectory from A to B consists of only one strategy). At point B, the two-link mechanism adopts one of two "strategies", elbow up or elbow down, to go to position C. The variation in strategy of the two-link mechanism exists only in the joint angle trajectory while the Cartesian trajectory is identical for both strategies.

In order to test the algorithms under various conditions, two control parameters that can change the characteristics of the trajectory were introduced: distance d and variation v. Distance d is illustrated in figure 2 and controls the extension of the two-link mechanism at its end position C. By varying d we control how different the joint trajectories are for each strategy (where d=1 yields identical strategies and decrementing d yields increasingly different strategies). The second parameter v is used to control the variability of the positional parameters: start position, end position, link lengths, and angle at which the singularity is reached. v controls the standard deviation of a Gaussian distribution offset which is applied to each positional parameter. The synthetic dataset consists of 100 trajectories (50 elbow-up and 50 elbow-down) for various combinations of d and v. Plots of the variations can be seen in Appendix A.

5.1.2 Proposed Framework Results

In order to test the accuracy of the proposed framework, the values of d and v were varied and for each configuration the accuracy was measured. In order test the robustness of the algorithm K-fold cross validation (where K=5) was used and the resulting accuracy results were averaged across the folds to produce the **Table** 5.1.

From the results is it possible to see that the proposed framework works very well in scenarios where the movement strategies are significantly different ($d > 0.75$), regardless of the variation or noise in the strategy. For scenarios where the movement strategies are similar and the variation is high, the algorithm loses some accuracy but still achieves greater than 95% accuracy.

To analyze how the GPDM algorithm can be used to convert the joint angle trajectories to the latent space, 2 trajectories (d=0.99 and d=0.5) for each of the two strategies were used for training the GPDM. By analyzing the reconstruction error, the algorithm determined that a latent space dimensionality of 2 would be sufficient for the model to capture variations about the movements. **Figure 5.2** shows the latent space trajectories of the data that was used to train the GPDM.

Table 5.1 – Proposed Algorithm Results on Synthetic Dataset

Variation v	Distance d = 0.99	0.95	0.90	0.75	0.50
0.01	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.05	C1: 49, 1 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.10	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.15	C1: 50, 0 / C2: 0, 50	C1: 49, 1 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.20	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.25	C1: 48, 2 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.30	C1: 50, 0 / C2: 4, 46	C1: 50, 0 / C2: 1, 49	C1: 49, 1 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50

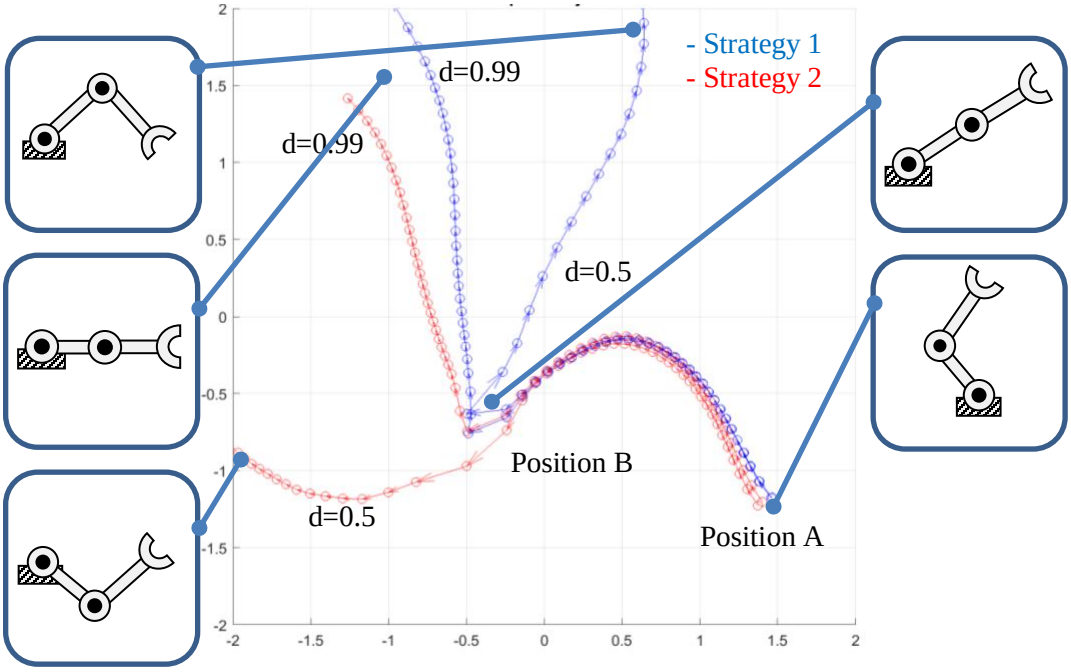


Figure 5.2 - Synthetic Dataset Latent Space Trajectory with Full Motion

The movements start at position A and progress to Position B with overlapping trajectories. This is expected since the strategies for the two motions were generated to be the same for the first half of the motion. Once at Position B, Strategy 1 progresses up and to the right, whereas Strategy 2 progresses to the left. We can infer that the x latent variable is related to the elbow up vs elbow down strategy (where anything to the left of position B (i.e. $x > 0.6$) is elbow up and anything to the right is elbow down). In this case it is challenging to determine what the y latent space variable represents. To simplify the problem, the common leg of the trajectory (position A to position B) was removed and the GPDM algorithm was used again to generate the latent space trajectory. The results are shown in **Figure 5.3**.

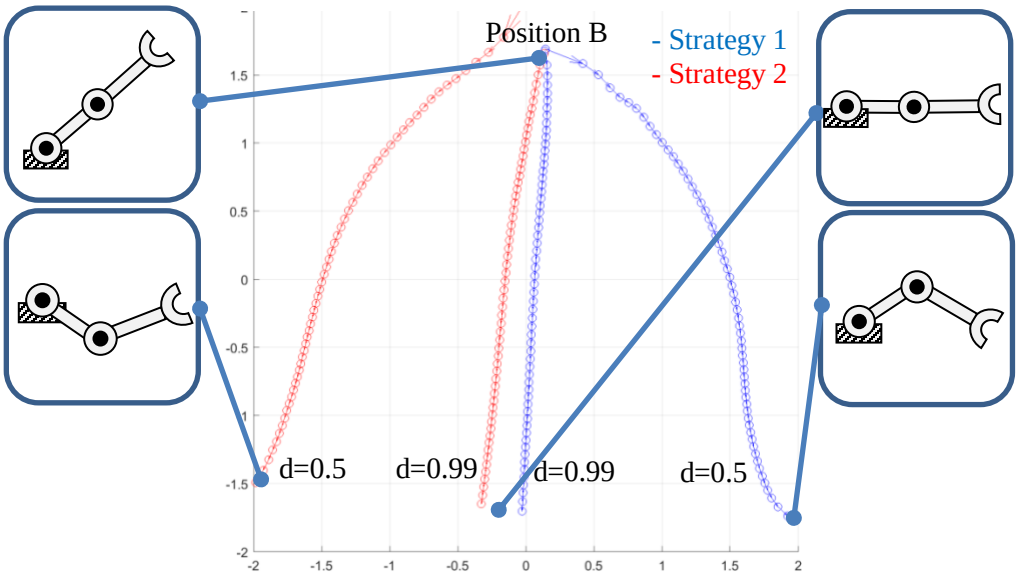


Figure 5.3 – Synthetic Dataset Latent Space Trajectory with Half Motion

With this result, we find that the x latent space variable represents the degree a strategy is elbow-up or elbow-down with the transition happening around x = 0. Additionally, the y latent space coordinate could potentially represent progression of the trajectory in the Cartesian domain.

The results seem to indicate that interpreting the latent variables may be easier for simpler motion primitives. To analyze the results further, Pearson correlation analysis was used to determine the strength of the relationship between the latent variables x and y and the DoF θ and Φ. The results for this are shown in the table below.

Table 5.2 – Correlation of Latent Variables to DoF for Synesthetic Dataset

		DoF	
		θ	Φ
Latent Variable	x	0.97	0.05
	y	0.95	0.11

Correlation analysis shows that there is strong linear correlation between latent variables x and y and the angle θ. The correlation x and y to the angle Φ is relatively weak on the other hand is expected because there exists a non-linear relationship between the strategy and Φ.

5.1.3 JCV Results

In order to apply the JCV approach it is necessary to develop an open-chain kinematic model for the human movements being studied. For the synthetic dataset this is straightforward since we defined a 2-link mechanism to generate the synthetic data. Using the model and the joint angle trajectories as input, we used the JCV algorithm to detect movement strategies. K-fold cross validation was used to validate the robustness of the algorithm (where K=5). **Table** 5.3 shows the results and has been color coded to indicate 100% accuracy (green) and <100% accuracy (red). C1 and C2 are the two clusters that were detected for two sub-strategies of elbow up and elbow and elbow down.

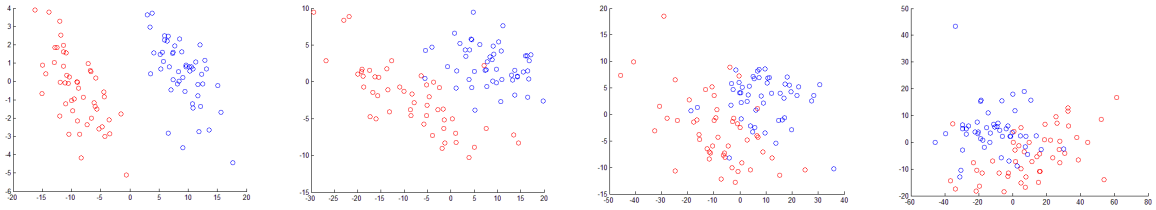
Table 5.3 – JCV Results on Synthetic Dataset

Variation v	Distance d = 0.99	0.95	0.90	0.75	0.50
0.01	C1: 40, 10 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.05	C1: 17, 33 / C2: 50, 0	C1: 50, 0 / C2: 0, 50	C1: 11, 39 / C2: 50, 0	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.10	C1: 38, 12 / C2: 1, 49	C1: 43, 7 / C2: 2, 48	C1: 9, 41 / C2: 47, 3	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.15	C1: 23, 27 / C2: 49, 1	C1: 20, 30 / C2: 41, 9	C1: 21, 29 / C2: 47, 3	C1: 7, 43 / C2: 48, 2	C1: 5, 45 / C2: 50, 0
0.20	C1: 25, 25 / C2: 49, 1	C1: 14, 36 / C2: 40, 10	C1: 20, 30 / C2: 45, 5	C1: 21, 29 / C2: 50, 0	C1: 33, 17 / C2: 0, 50
0.25	C1: 19, 31 / C2: 47, 3	C1: 22, 28 / C2: 9, 41	C1: 16, 34 / C2: 45, 5	C1: 33, 17 / C2: 3, 47	C1: 30, 20 / C2: 0, 50
0.30	C1: 17, 33 / C2: 3, 47	C1: 30, 20 / C2: 3, 47	C1: 29, 21 / C2: 9, 41	C1: 26, 24 / C2: 6, 44	C1: 17, 33 / C2: 46, 4

The results show that the algorithm performs best for cases where the motion variation is low and can handle slightly larger variations as long as the difference in strategies is significant. This result is somewhat expected, because while the joint contribution approach accounts for movement progressions

through aggregating joint contributions over time, the comparison of the overall JCV metric between two motions loses this information.

Since Park et al [5] suggested the use of MDS for visually detecting clusters, we created MDS visualizations to see the impact of increasing variation on the data set. Specifically, we take the JCV vector and use MDS to reduce the vector to two dimensions so that we can plot it. We illustrate the case of d=0.75 in which the strategies are fairly different and create this MDS plot for v={0.05,0.10,0.15,0.20}. The result is shown below where the red/blue points designate the two strategies:



(a) v=0.05 (b) v=0.10 (c) v=0.15 (d) v=0.20

Figure 5.4 – MDS Plots for JCV Clusters for d = 0.75 and v = {0.05,0.10,0.15,0.20}

We can see that as soon as the points representing the two clusters start overlapping, the JCV clustering algorithm fails to classify the two strategies accurately (v >= 0.1). This shows that the JCV algorithm does not perform well for the cases where there are distinct strategies and large variation.

5.1.4 Kinematic Synergy Results

For the kinematic joint synergy approach, the VAF metric was used to determine the ideal number of synergies for the dataset. For the synthetic dataset this was discovered to be 2 synergies. **Figure** 5.5 shows how the reconstruction error does not improve beyond 2 synergies.

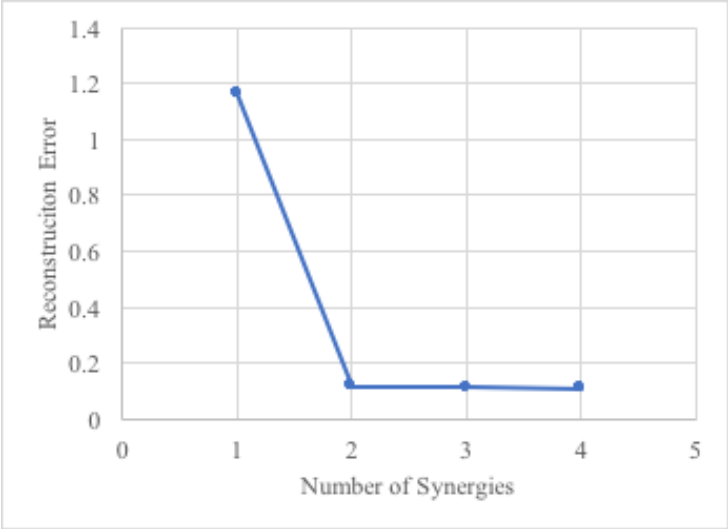


Figure 5.5 – Detecting the Ideal Number of Synergies for Synthetic Dataset

Since we chose to represent the trajectories with 2 synergies and the trajectories consist of 2 joints, there are a total of 2 [1x2] synergy vectors. To determine the usefulness of the synergy vector, the synergy vector values for both joints were plotted for a subset of the trajectories (we use one example from each value of *d* and from each strategy) – see figure below. It can be seen here that one synergy variable impacts the choice of strategy whereas the other one seems to impact the degree to which the strategy is elbow or elbow down (i.e. correlates with the variable *d*).

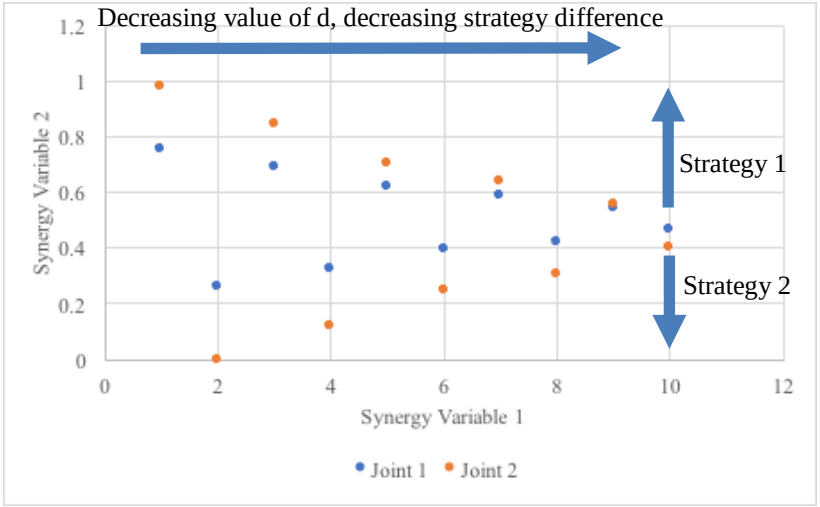


Figure 5.6 – Plotting Synergy Vectors for Synthetic Dataset

An alternative way to plot this for each motion is to use multi-dimensional scaling (MDS), where each data point is a concatenation of the synergy vectors for all joints for that motion. While this is not

necessary for the synthetic data case given the low dimensionality of the original trajectory, it will be useful to visualize high dimensional trajectories with this approach. The figure below shows this for the synthetic dataset for all low noise configurations.

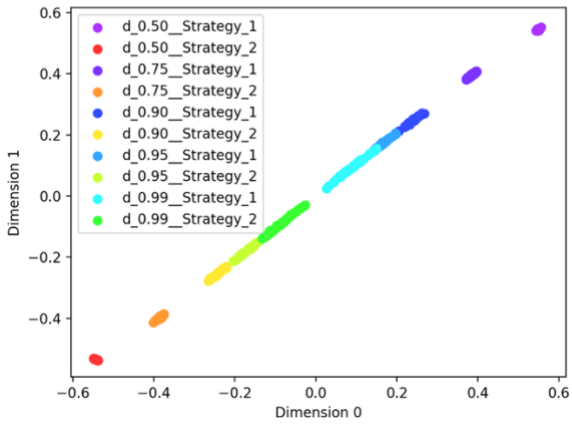


Figure 5.7 – MDS plot of Joint Synergies for Synthetic Dataset

In order to use the matrix factorization approach for classification purposes, any test motions need to be added as an input to the matrix factorization approach to re-generate synergy vectors. As a result, the resulting data points and cluster positions may change. In order to keep track of which strategy is associated with which cluster, we maintain a list of trajectories that are in each cluster and use that to map the clusters between experiments. K-fold cross validation was used to validate the robustness of the algorithm (where K=5). The results of the classification across the various experiment configurations are shown in **Table** 5.4. The classification results show that the algorithm performs very well in low noise scenarios and in scenarios where the differences in motions is high. For high noise scenarios it does not have high accuracy.

Table 5.4 – Kinematic Synergy Results on Synthetic Dataset

Variation v	Distance d = 0.99	Distance d = 0.95	Distance d = 0.90	Distance d = 0.75	Distance d = 0.50
0.01	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.05	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.10	C1: 47, 3 / C2: 1, 49	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.15	C1: 40, 10 / C2: 15, 35	C1: 47, 3 / C2: 4, 46	C1: 48, 2 / C2: 5, 45	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.20	C1: 25, 25 / C2: 49, 1	C1: 37, 13 / C2: 11, 39	C1: 50, 0 / C2: 0, 50	C1: 21, 29 / C2: 50, 0	C1: 50, 0 / C2: 0, 50
0.25	C1: 31, 19 / C2: 3, 47	C1: 22, 28 / C2: 9, 41	C1: 16, 34 / C2: 45, 5	C1: 50, 0 / C2: 0, 50	C1: 50, 0 / C2: 0, 50
0.30	C1: 33, 17 / C2: 20, 30	C1: 30, 20 / C2: 3, 47	C1: 30, 20 / C2: 8, 42	C1: 47, 3 / C2: 0, 50	C1: 48, 2 / C2: 0, 50

5.2 Discussion

Comparing the accuracy of three approaches (JCV, Kinematic Synergy, and Proposed approach), the proposed approach works when the data is noisy and when the goal is to detect slight differences in motion strategies. The next best algorithm is the kinematic strategy approach, with JCV coming in last.

This accuracy of the proposed approach comes at a cost of computational complexity in training time. As mentioned previously the proposed approach transforms the original observation sequence into different models and uses algorithms that are computationally complex. The GPDM training approach we use is the same one used by [4] which has a computational complexity of $O(N^3)$ for the matrix inversions that are done, where N is the number data points [54]. Additionally, HMM training has a computational complexity of $O(TNQQ_{max})$, where T is the length of motion, N is the number of states, Q is the number of transition parameters and Q_{max} is the maximum HMM node-out degree [74].

As such both GPDM and HMM algorithms do not generally scale well as the dimensionality of the input data and the size of the input data increases. Note that even though the synthetic data is relatively simple (motions consist of 2 DoF, with 60 time samples, and with 100 samples per noise/distance configuration), the proposed approach still takes longer to compute. The figure below shows the training performance of the three algorithms on a 2.2 Ghz, Intel Core i5 Processor:

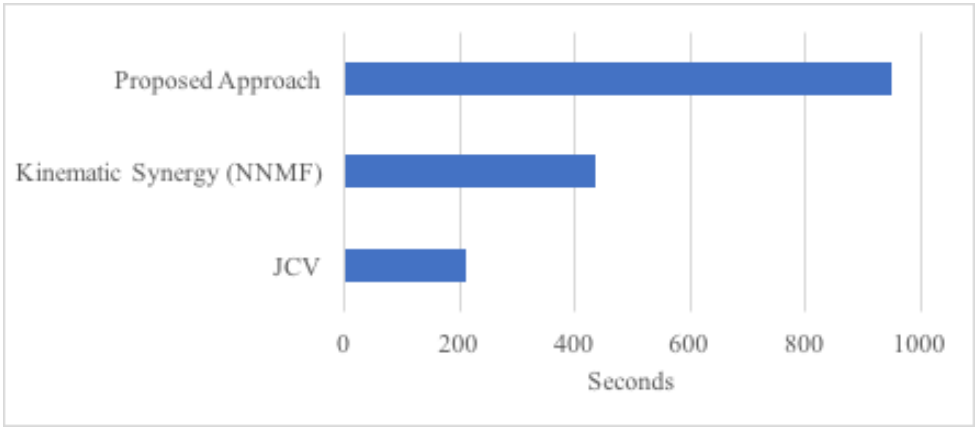


Figure 5.8 - Algorithm Training Time on Synthetic Dataset

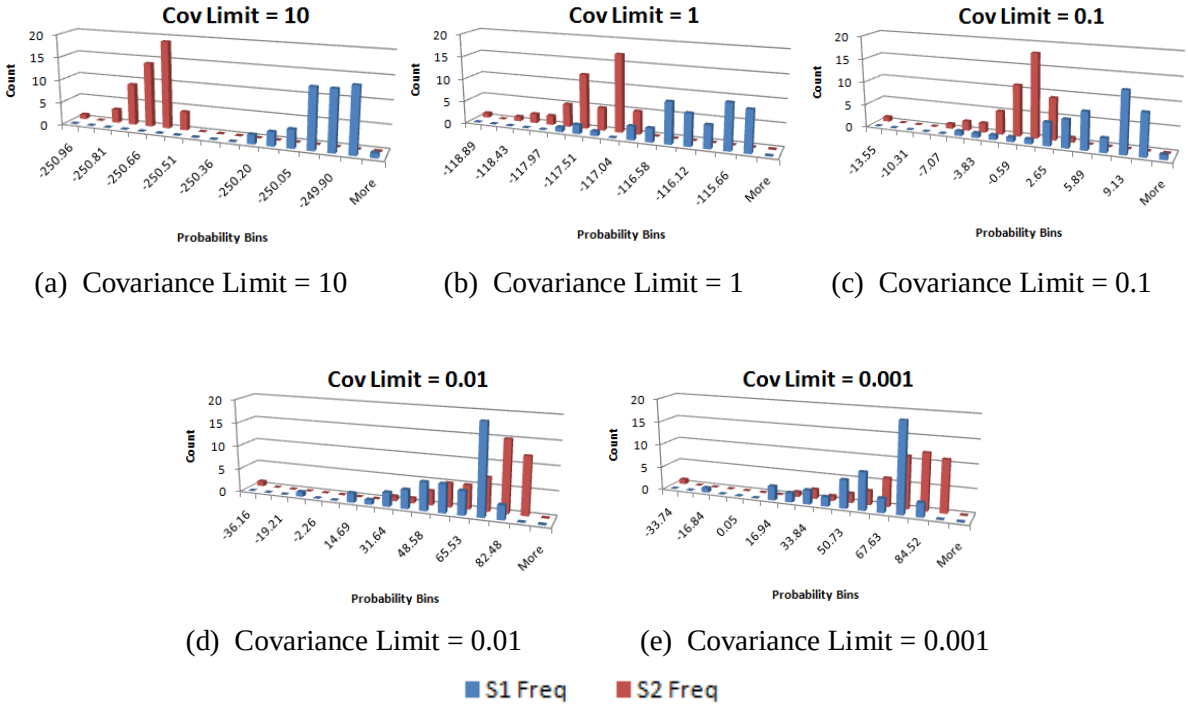
The other performance factor to consider is the classification time once the models have been trained. Recognition of motions for JCV is in the sub-second range while the proposed approach is on the order of seconds. On the other hand, the classification time for the kinematic synergy approach is roughly the same as the training time as the entire dataset needs to undergo NNMF again.

5.2.1 Proposed Approach Investigations: Impact of Covariance Limit

In this section we will provide some details on how we improved the results of the proposed approach by changing the minimum value that we allow the covariance values to be set to when training HMM models as part of the divisive clustering algorithm. This minimum covariance value will be referred to as the covariance limit. There are several reasons for why setting a covariance limit may be important. One reason is that by setting a large value for the covariance limit, the model is prevented from over-specializing and provides a more generalized model. Another reason is that if all the models have the same covariance values (i.e. if it is high enough), then the covariance of the models do not contribute as much to the distances between models.

To study the impact of the covariance limit, the synthetic dataset for the case where d=0.5 and v=0.1 will be used (i.e. movement strategies are fairly different with some amount of variance). For this dataset one HMM model will be trained with all the sequences and then the probabilities that the

model generates the sequences (i.e. the first step of the HMM based divisive clustering) are plotted. The analysis is repeated for various covariance limits (10, 1, 0.1, 0.01, 0.001). The figure below was generated showing the distributions of the probabilities for the elbow up strategy (S1, blue) and elbow down strategy (S2, red).



(a) Covariance Limit = 10 (b) Covariance Limit = 1 (c) Covariance Limit = 0.1

(d) Covariance Limit = 0.01 (e) Covariance Limit = 0.001

Figure 5.9 – Impact on Probability Distribution on Motion A-B-C

As the covariance limit increases, we see that distributions become more and more separable (less overlap between the distributions). This is likely because the covariance adds some amount of noise to the distance measurements. One interesting thing to note in the above figure is the range of bin values as the covariance limit changes – though the distributions become more separable the absolute values between the means of the distributions become lower. The larger distances are a result of the lower covariance values contributing to larger distances between more dissimilar movements.

In our particular case, we are more interested in separability of movement strategies so based on these results we use a covariance limit of 10 in our subsequent experiments. This value is larger than the covariances we will expect to observe due to the normalization that we do to the dataset on the range of -1 to 1.

5.3 Labeled Dataset (Squat-Stoop)

5.3.1 Experiment Setup

The second dataset consists of real human movement in which the participants perform a known predetermined lifting strategy (squat or stoop). A total of 10 participants performed 9.3 and/or 24.7 kg floor-to-waist lifts between 2-5 times. A total of 47 trials were included in the tests performed here (27 squat and 19 stoop lifts) [40]. Cartesian coordinates of various lower body segments (feet, shanks, thighs, trunk, pelvis) were collected for the motions using the Optotrak motion capture system. Standard inverse kinematics computations were used to convert the Cartesian coordinates into Cardan joint angles for the ankles, knees, hips, and lumbar spine (Visual3D™, C-Motion Inc.).

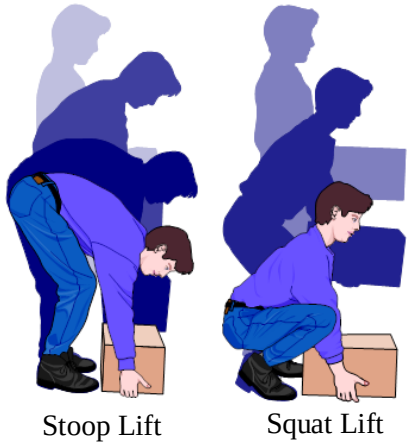


Figure 5.10 - Illustration of Stoop and Squat Lifts

5.3.2 Proposed Framework Results

For the Squat-Stoop dataset the data was input into the GPDM algorithm and through the iterative process determined that a dimensionality of 3 is optimal, where the first 2 latent dimensions capture most of the variability (41%, 33%) and the third one captures 15% of the variability.

To test the utility of the proposed approach, we tested its classification accuracy on both joint angle data and Cartesian coordinate data for the same set of motions. Additionally, we also compared how the algorithm would perform with or without the low dimensionality reduction (i.e. with or without GPDM). The proposed approach was tested in all scenarios using K-fold cross validation (where K=5). Results are presented in the table below.

Table 5.5 – Results of Proposed Framework on Squat/Stoop Dataset

<table>
<tr><th colspan="2" rowspan="2"></th><th colspan="2">Dataset Type</th></tr>
<tr><th>Joint Angles</th><th>Cartesian Coordinates</th></tr>
<tr><td rowspan="2">Algorithm Variation</td><td>HMM Divisive Clustering (without GPDM)</td><td>
<table>
<tr><td></td><td>C1</td><td>C2</td></tr>
<tr><td>Squat</td><td>0</td><td>27</td></tr>
<tr><td>Stoop</td><td>19</td><td>0</td></tr>
</table>
</td><td>
<table>
<tr><td></td><td>C1</td><td>C2</td></tr>
<tr><td>Squat</td><td>0</td><td>27</td></tr>
<tr><td>Stoop</td><td>19</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>HMM Divisive Clustering (with GPDM)</td><td>
<table>
<tr><td></td><td>C1</td><td>C2</td></tr>
<tr><td>Squat</td><td>0</td><td>27</td></tr>
<tr><td>Stoop</td><td>19</td><td>0</td></tr>
</table>
</td><td>
<table>
<tr><td></td><td>C1</td><td>C2</td></tr>
<tr><td>Squat</td><td>0</td><td>27</td></tr>
<tr><td>Stoop</td><td>19</td><td>0</td></tr>
</table>
</td></tr>
</table>

We find that the algorithm achieved 100% accuracy in all scenarios. The main difference between the GPDM and non-GPDM approach is the computational performance (shown in figure below). The initial dimensionality reduction by the GDPM helps reduce the time it takes to perform the HMM based divisive clustering since the trajectories are now in a lower dimensional space.

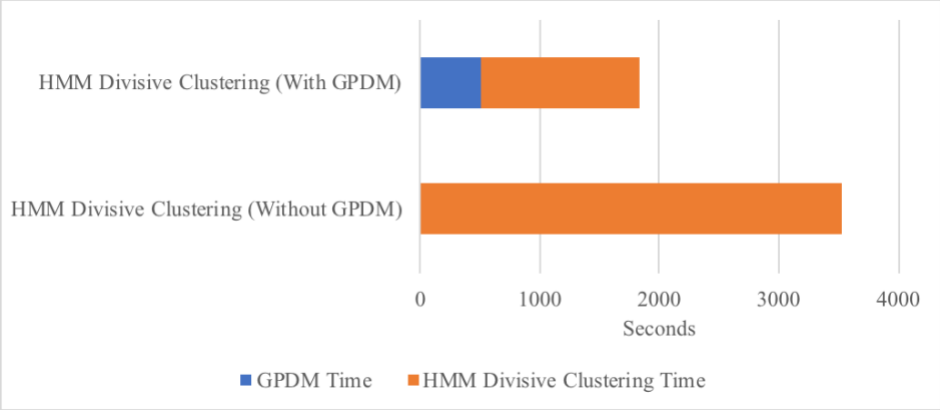


Figure 5.11 - Performance Comparison of Proposed Approach Variations

Note that in addition to detecting the squat stoop and stoop strategies, the proposed approach was able to further detect sub-strategies for squat indicating a narrow-leg and a wide-leg lift. The motion tree for this is shown in the figure below:

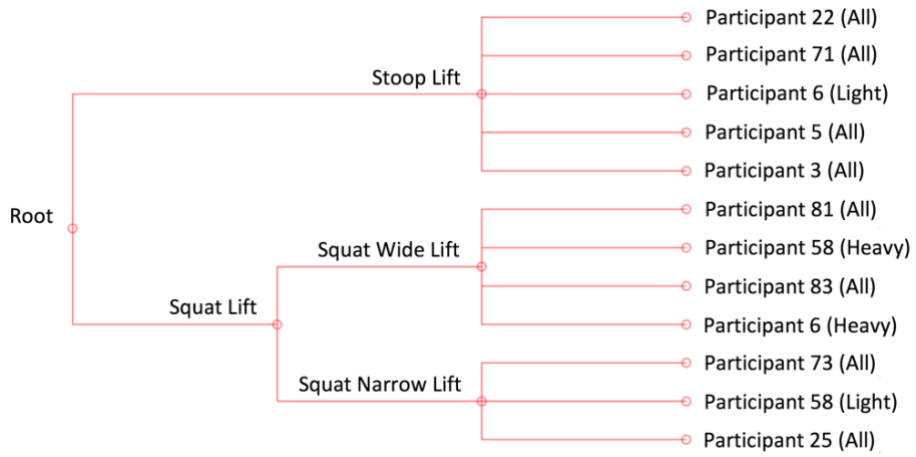


Figure 5.12 – Motion Tree for Squat/Stoop Dataset

For further in-depth analysis we will first focus on the first two latency dimensions. The latent trajectories generated by the GDPM algorithm are shown below in addition to key poses of interest at various points in the latent space.

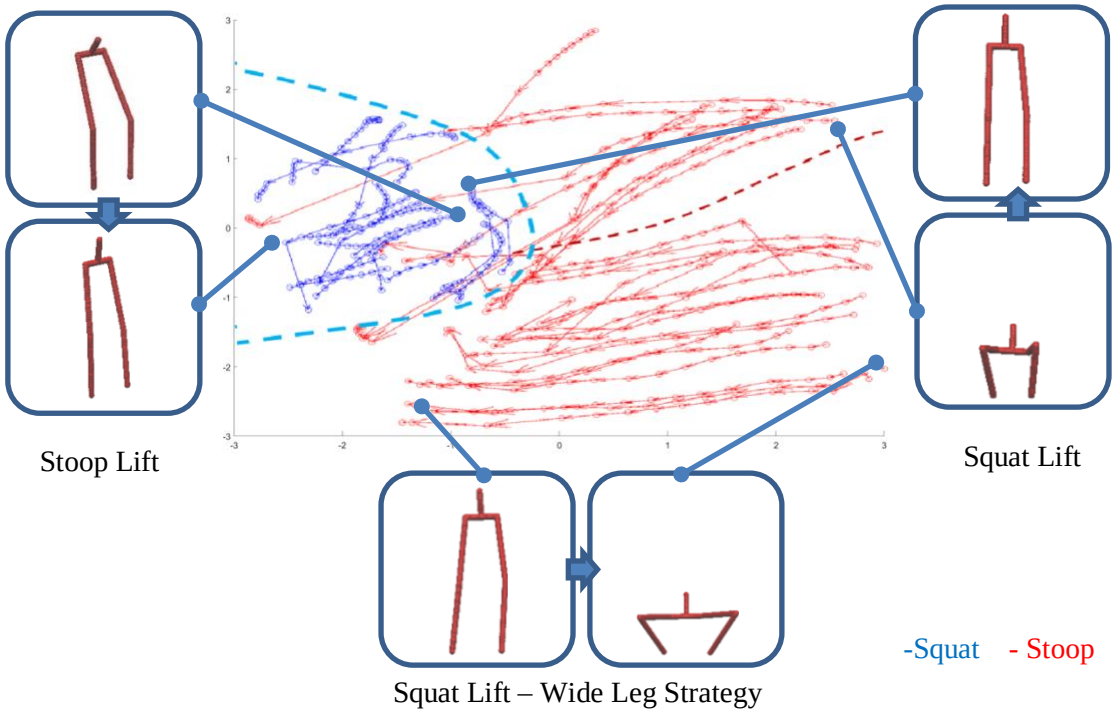
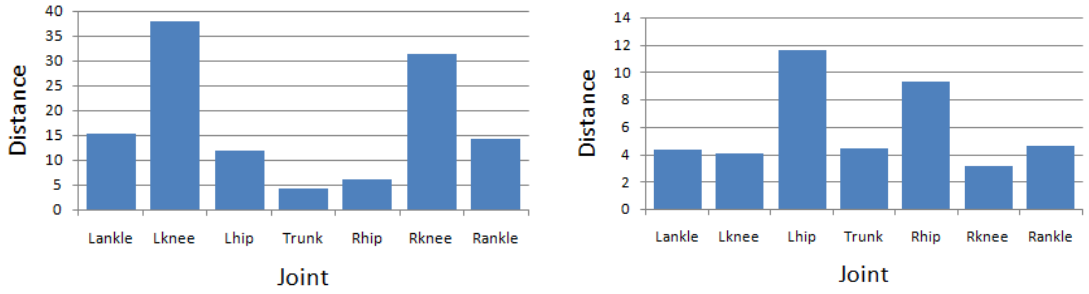


Figure 5.13 – Squat-Stoop Dataset Latent Space Trajectory (x vs y)

The x latent space variable appears to relate to the temporal state of the lift motions. The right side represents states where the pelvis is closer to the ground whereas the left-side represents a standing state. For the squat lifts, the lifts start with the pelvis close to the ground and progress to a standing state, where as for the stoop lifts the pelvis remains close to a standing posture and the back does most of the movement. In the latent space trajectories, the squat and stoop trajectories intersect at the final states, due to similar poses.

In order to understand the impact on the joints for these strategies and sub-strategies we use the proposed joint analysis. The results are shown in the figures below:



(a) Squat vs Stoop Comparison (b) Squat Sub-Strategy Comparisons

Figure 5.14 – Squat-Stoop Included DoF Comparison

For the Squat-Stoop DoF comparison we can see that the graph is almost symmetric about the pelvis/spine. This is expected because on average squat and stoop motions are symmetric. Recall that the joint with the largest distance in the included-joint analysis is the joint that is most different between the motions. In this case we find that the knee is the most different between the squat and stoop motions. We know visually from the figure above that knee barely moves in the stoop lift and is the most actively used joint in the squat move.

For the Squat Sub-Strategy DoF comparison, it appears as if the main difference between the motions is happening at the shank and the thighs where overall it seems the differences are symmetric. In this case we find that the hip is the most different between the motions. Note that the relative difference between the distance of hips in this graph (8-12) is much lower than the distance for the squat-stoop motions (30-35). This indicates that while there is a difference between the sub-strategies it is not as great as the difference between the squat stoop motions.

We next examine the correlation between the latent space variables and the variations in cartesian angle trajectories (**Figure** 5.15). This analysis shows that there is strong correlation (>89%) between the latent variable dim-1 and shank/thigh z-axis movement and as well as shank y-axis movement. All

of these are expected differences between the squat and stoop lift motions. The latent variable dim-2 seems to relate to the strategy of widening the legs during the lifts. The larger the dim-2 value the closer the knees are to each other during the lift. This is confirmed by the fact that foot/shank x-axis movement correlate well (>82%) with the dim-2.

Note that the underlying assumption in the Pearson correlation coefficient is that the relationship is linear. While this may not necessarily be true for human motion data in general, for the data being analyzed in this thesis, the results were informative. In particular, in our analysis we found that some latent variables had correlation values above ~0.8 with some of the joint trajectories indicating fairly strong linear correlation (note that a correlation value of 1 means 100% linear correlation, a value of -1 means there is 100% negative correlation, and value of 0 means that there is no correlation at all).

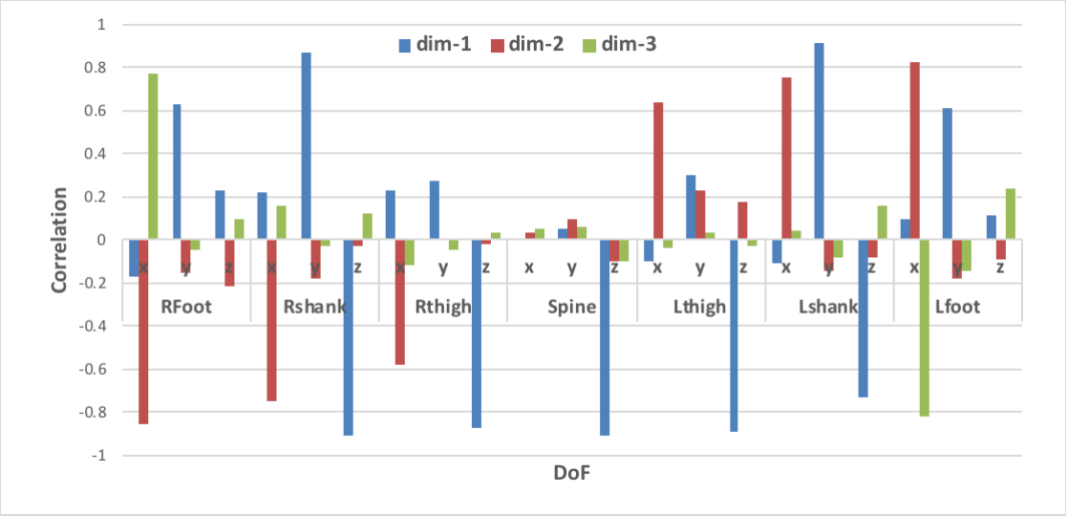


Figure 5.15 - Correlating Latent Space Variables with DoF

To analyze the 3rd latent variable, **Figure** 5.16 was generated, where the horizontal axis is the latent variable dim-1 and the vertical axis is the latent variable dim-3. The figure shows that most of the stoop motions have an upward trajectory. On the contrary some squat motions follow a parabolic trajectory where they first go down before going up. The correlation analysis shows that the x-axis of the foot also undergoes a similar trajectory pattern. To confirm this, we plotted the x-axis trajectories for the right foot for all the motions (see **Figure** 5.17). In particular it looks like some participants take on two kinds of movement strategies with their feet, one is where the move their feet and leave them there and the second one where they move the feet and then move them back close to the original position.

Note that because the foot placement strategy seems to on average relate to the squat/stoop strategies these were not detected as separate motion strategies in the motion tree.

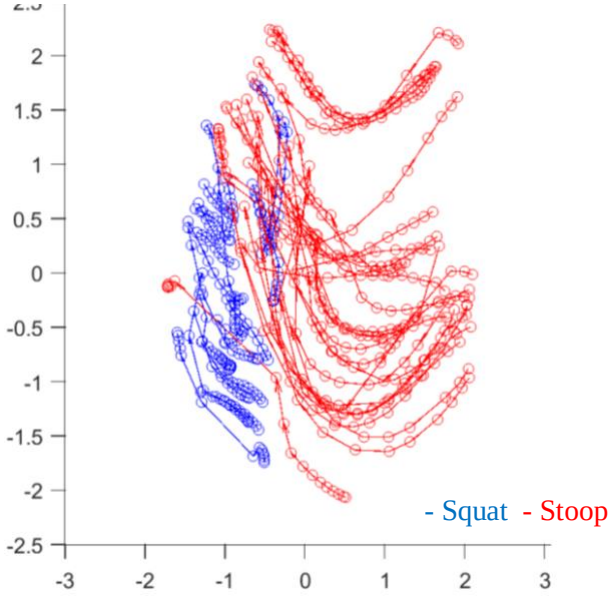


Figure 5.16 - Squat-Stoop Dataset Latent Space Trajectory (dim-1 vs dim-3)

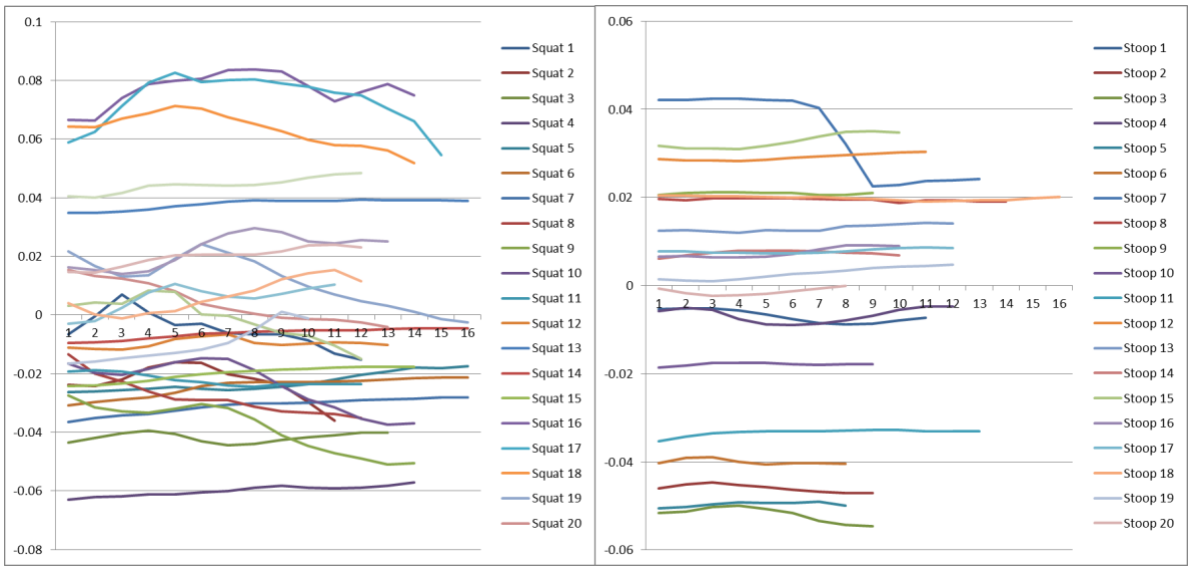


Figure 5.17 - Right Foot X-Axis Trajectories for Squat-Stoop Motions

In addition to being able to be able to detect different strategies and differences that exist between them, the proposed approach can also generate exemplar motions for visualization purposes. Below

are reconstructed examples of average squat and stoop motions and also two variations of squat stoop motions that were generated by the GPDM algorithm.

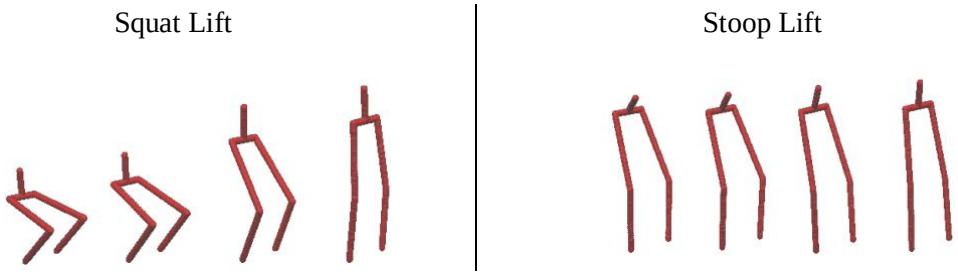


Figure 5.18 – Examples of Squat and Stop Lift Motions

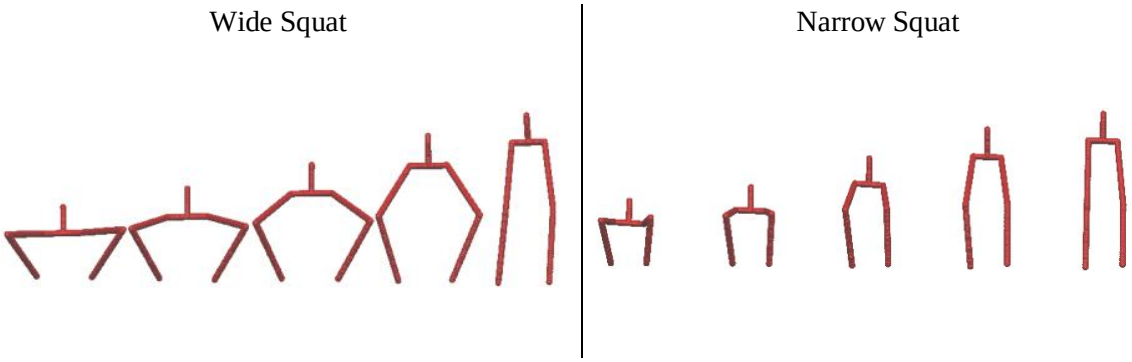


Figure 5.19 –Examples of Wide Squat and Narrow Squat

5.3.3 JCV Results

5.3.3.1 Model

In the original paper by Park et al [5], when the JCV algorithm was introduced the authors tested their algorithm on a squat/stoop data set. In their approach they use a 5-link kinematic structure to represent the motion as shown below:

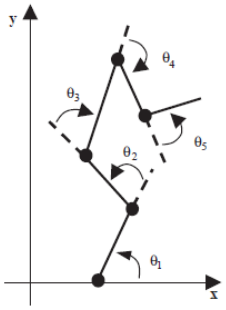


Figure 5.20 – Park et al's Model for Squat/Stoop Analysis

This modeling approach assumes that the motion is symmetrical, which is not necessarily true for natural lifting motions. The algorithm also assumes that there is no significant movement outside of the saggital plane and thus would fail to identify the strategies which rely on leg widening as shown in figure 26. An alternative to this approach would be to use a 3D kinematic model as Park et al used in their study 2. An issue with this approach is that it requires that one of the participant's foot remains stationary and that the other foot does not contribute to the motion (i.e. exert any force on the ground). Whether we use the 2D model or the 3D model, one of the requirements of the model is that it should be an open kinematic chain. Again, an open kinematic chain may not be representative of real motions.

To test the JCV algorithm, we used a 3D model of the participant utilizing a 6-link kinematic structure representing the lower body. For the end-effector we used the trunk spine Cartesian coordinates. For the model we assumed that one foot always remained stationary and thus the joint angles for the leg with the non-stationary foot did not contribute to the JCV vector. These assumptions can be made considering the symmetrical nature of the squat/stoop lift in this particular dataset.

5.3.3.2 Motion Analysis

Recall that the JCV vector contains the contributions of individual DOFs to changes in the end-effector position. To test the algorithm's versatility, we decided to test the JCV clustering under two varying conditions:

- Check to see if JCV clustering is sensitive to analyzing the contributions of individual DOF vs. contributions of joints (sets of 3 DOF)
- Check to see if JCV clustering is sensitive to whether we consider the contributions in just the X-Y coordinates of the end effector vs. the X-Y-Z coordinates of the end effector.

The results of testing the algorithm under these conditions are shown in the table below:

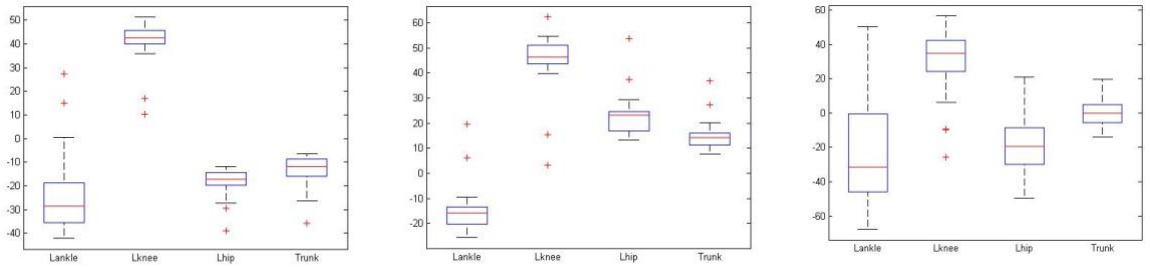
Table 5.6 – Results of JCV Algorithm on Squat/Stoop Dataset

		Contribution to End-Effector Coordinates					
		X & Y			**X, Y, & Z**		
Contribution of Angles	**DOF**		C1	C2		C1	C2
		Squat	0	27	Squat	0	27
		Stoop	19	0	Stoop	19	0
	Joint		C1	C2		C1	C2
		Squat	0	27	Squat	0	27
		Stoop	19	0	Stoop	19	0

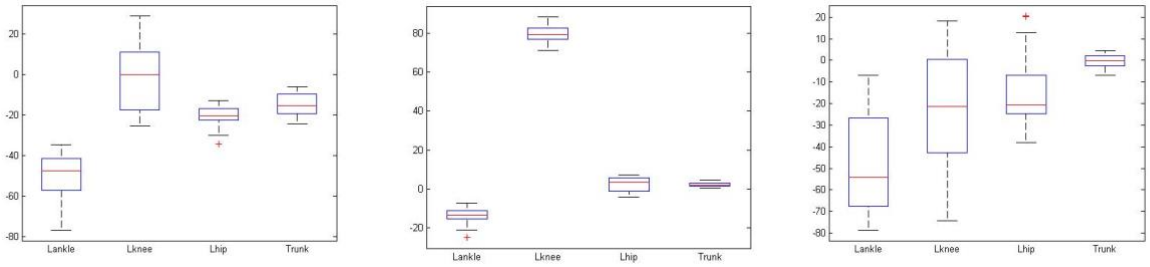
As we can see the clustering algorithm achieves 100% accuracy in all cases tested. These results are much better than those presented in Park et al's work. We believe that difference in performance could be due to several reasons:

- In Park et al's work the full body model was used. It is possible that the movement of the upper body limbs (arms) introduced additional noise into the JCV vector.
- In Park et al's work a 2D 5-link kinematic structure used. It is possible that the additional error was introduced as a result of converting a 3D motion to a 2D plane.

To analyze the joint differences, we also generated some box plots of the PC vectors (similar to the approach in Park et al's paper) which are shown below:



(a) Squat PC_x (b) Squat PC_y (c) Squat PC_z



(d) Stoop PC_x (e) Stoop PC_y (f) Stoop PC_z

Figure 5.21 – Percent Contribution Analysis for Squat/Stoop Motions

This style of box plots was used by Park to demonstrate the contributions of the DOF in the motions. Trying to analyze the PC range and distribution seems to lead to results which don't appear to be consistent with the expected results. For example, in the squat motions we can see that the knee has more contribution in the x-direction (horizontal) than stoop, but this same trend does not hold for the y-direction (vertical). In contrast the DoF analysis of the proposed approach seems to be more intuitive and appears to yield more accurate results. One possible reason why the results in figure 21 may not match up with expectations is because the results may be sensitive to the selection of the end-effector (where we chose the spine as the end-effector and Park et al chose the hand as the end-effector).

5.3.4 Kinematic Synergy Results

For the kinematic joint synergy approach, the VAF metric was used to determine the ideal number of synergies for the dataset. For the squat-stoop dataset this was discovered to be 2 synergies.

The results of running classification analysis using k-fold validation on the kinematic synergy algorithm is presented in the table below. Note that this algorithm was tested on joint angle trajectory

data as well as Cartesian trajectory data to determine its versatility. As the dataset is separable between squat and stoop motions, we find that the algorithm achieved 100% accuracy regardless of the dataset.

Table 5.7 – Results of Kinematic Synergy Detection on Squat/Stoop Dataset

Dataset Type					
Joint Angles			**Cartesian Coordinates**		
	C1	C2		C1	C2
Squat	0	27	Squat	0	27
Stoop	19	0	Stoop	19	0

The MDS plot below shows how well the Squat and Stoop clusters are separated. Each data point in this plot had a feature that was a concatenation of all the synergy vector values for all DoFs for the participant. Dimension 0 and Dimension 1 here may not have any physically meaningful interpretation.

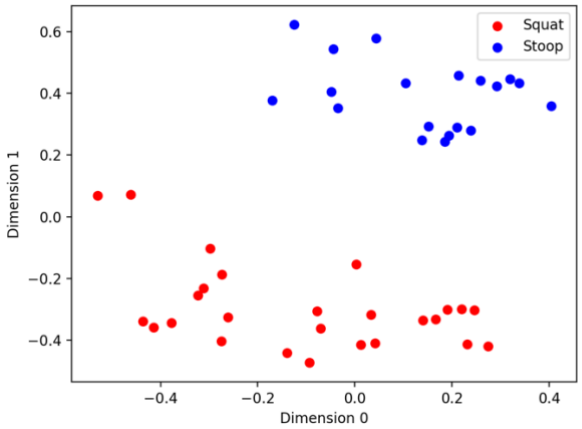


Figure 5.22 – MDS plot of Joint Synergies for Synthetic Dataset

To investigate and understand the benefits of the kinematic synergy approach, we investigate how the synergies impact the DoF for the motions. The figure below shows the distributions of synergies 1 and 2 across all participants for squat and stoop motions.

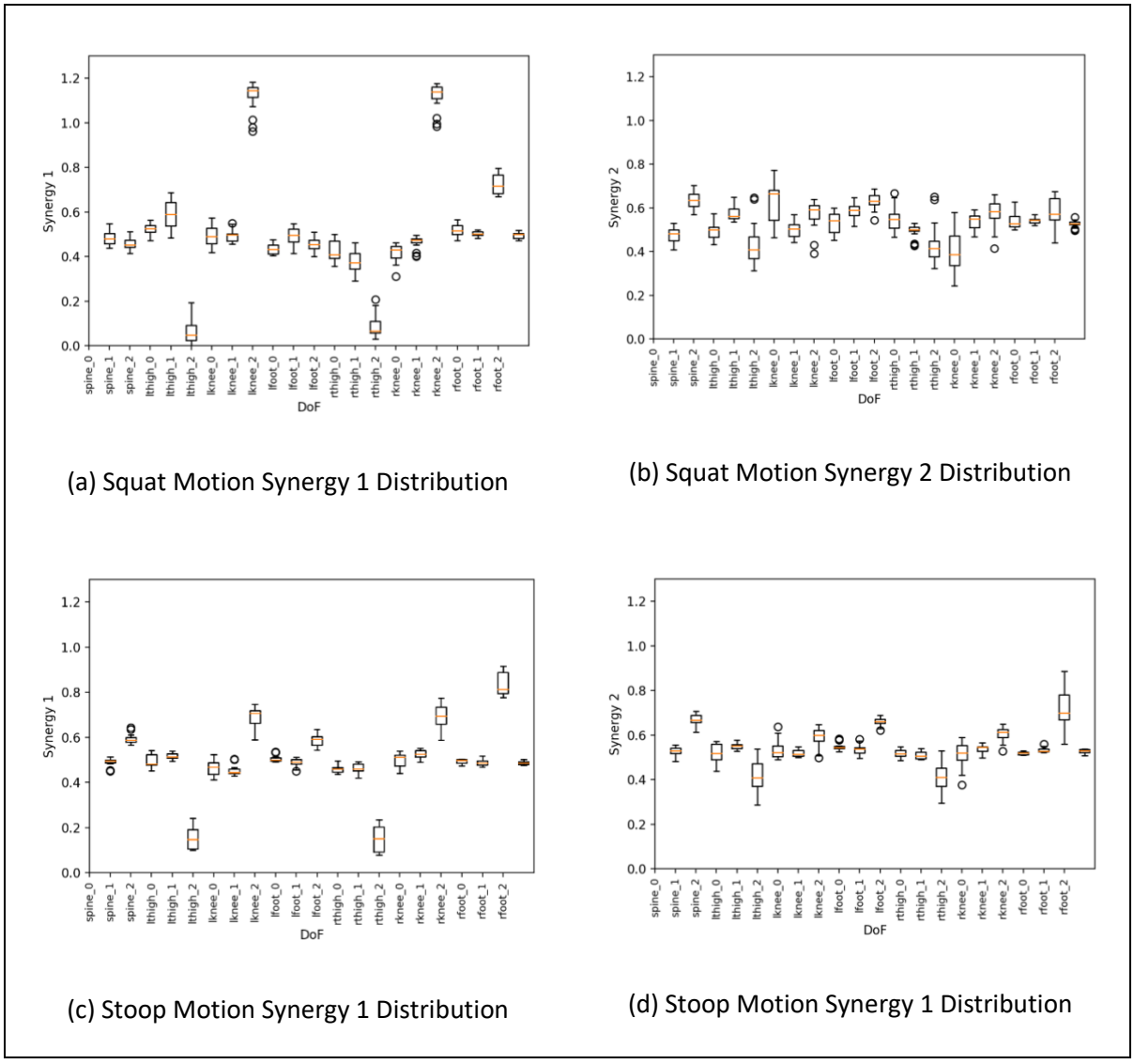


Figure 5.23 – Synergy Distributions Per DoF

Synergy 1 has particularly high values for the knees with low variance for squat motions. This makes sense since squat motions would require more movement of the knees. The knees in the stoop motions have high variance, which may indicate that participants don't use their knees the same way for stoop motions. Lastly, one of the hip joint angles have low synergy 1 values for both squat and stoop, indicating that the particular hip joint angle does not vary much during the movement. Synergy 2 does not seem as easily interpretable, but we know from the VAF calculations that it is needed for achieving low reconstruction error.

5.3.5 Discussion

For this particular dataset, all of the comparison algorithms were able to successfully classify Squat and Stoop motions. The proposed approach was able to discover additional sub-strategies that are difficult to discover with the JCV and kinematic synergy approach. These sub-strategies are the wide squat and narrow squat sub-strategies and also a sub-strategy related to foot placement.

The latent space visualization helps provide a simplified representation of what the state space for the movements look like and enables visualizing the clustering results. Doing this kind of analysis with kinematic synergies is challenging because of the number of synergy vectors and DoF that have be analyzed for this particular dataset. Additionally, because all the models in the proposed approach are generative it is possible to generate exemplar motions for any sub-strategies that are discovered.

Finally, the cost of the proposed approach is computational and time complexity (see figure below). This is particularly impacted by the fact that the dimensionality of the squat stoop dataset is larger (21 DoF) than our synthetic data set (2 DoF).

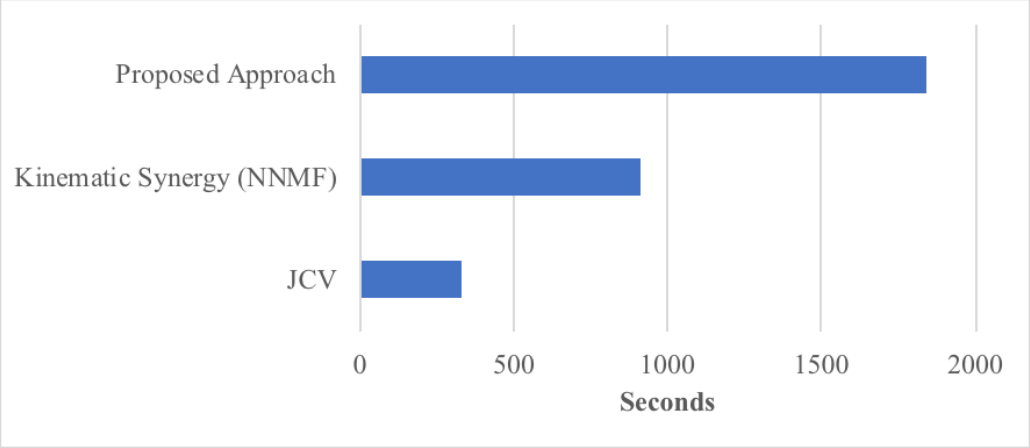


Figure 5.24 - Algorithm Training Time on Squat-Stoop Dataset

5.3.6 Proposed Approach Investigations: Impact of Number of Latent Parameters

The dimensionality of the latent space can impact the performance of the clustering algorithm in several ways. Some of the performance criteria that will be discussed in this section are reconstruction error, computational performance (speed) and the clustering quality.

To better understand the relationship between speed, reconstruction error and latent space dimensionality we did an experiment where we trained a GPDM with all motions from the squat-stoop dataset while sweeping the latent space dimensionality. For each dimension we recorded the reconstruction error and the training time. The result of this is shown in the **Figure** 5.25.

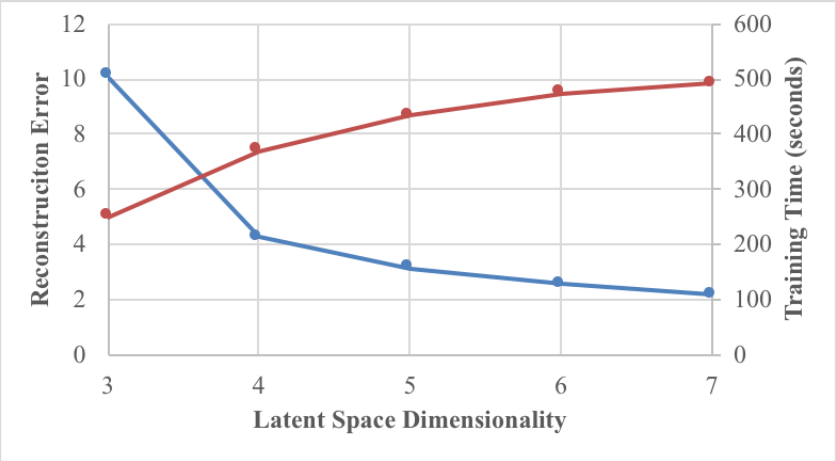
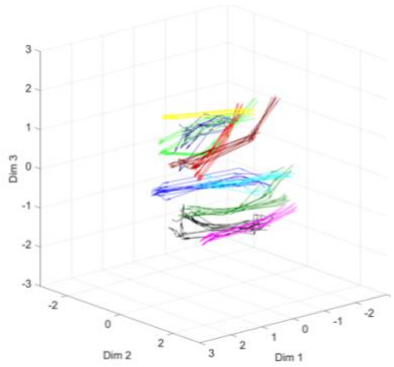


Figure 5.25 – Reconstruction Errors and GPDM Training Time vs Latent Space Dimensionality for Squat/Stoop Dataset

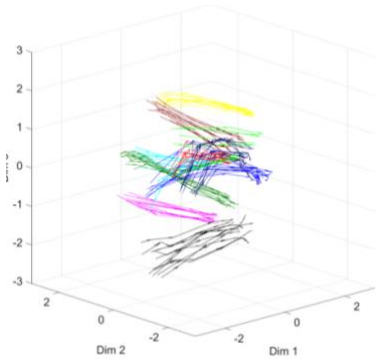
As expected, the reconstruction error becomes marginally better after the first few latent dimensions as the first few latent dimensions are able to capture most of the information regarding the variability of the movement. Additionally, the time for training the GPDM model increases for each additional latent dimension, where the increase in time is less for each additional latent dimension. This makes sense, given that each additional latent dimension doesn't contain as much information about the movement and the GPDM model parameters starts stabilizing around similar values.

Since the GPDM is used to generate latent trajectories to cluster in the HMM divisive clustering algorithm, it was important that we understand the impact of the latent space dimensionality on clustering results as well. To measure cluster coherency we use a Euclidean distance based metric to measure the sum of the distances from points on the cluster center trajectory to the observation trajectories. The average of distances for all the observation sequences is used to measure cluster coherency. As we increase the latent space trajectory we found that the average cluster distance increased, as shown in **Figure** 5.26. This indicates that even though reconstruction error reduces with increasing latent space dimensionality it also introduces some noise as well. So there is a trade-off between reconstruction accuracy and generating coherent clusters for detection purposes.



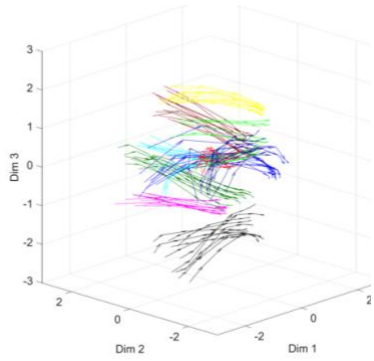
Avg Cluster Distance = 10.1

(a) Latent Dimensionality = 3



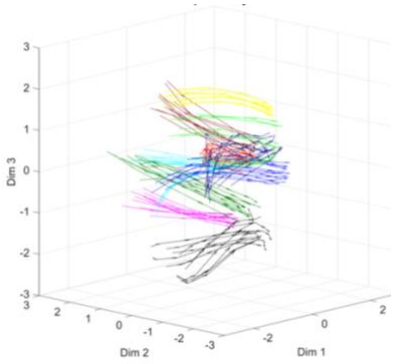
Avg Cluster Distance = 11.3

(a) Latent Dimensionality = 4



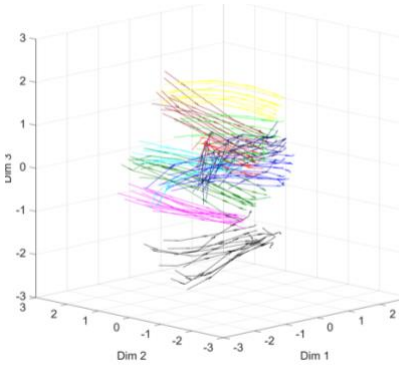
Avg Cluster Distance = 12.2

(a) Latent Dimensionality = 5



Avg Cluster Distance = 14.9

(a) Latent Dimensionality = 6



Avg Cluster Distance = 15.2

(a) Latent Dimensionality = 7

Figure 5.26 – Impact of Latent Space Dimensionality on Clustering Results

5.4 Targeted Lifting Motion Dataset

5.4.1 Experiment Setup

The final dataset consists of lifting data where the lift pick-up and drop-off locations are specified but the number of strategies is not known a priori. The lifts in the dataset are not symmetric and it is unknown whether participants will be moving one foot or both feet for the duration of the lift.

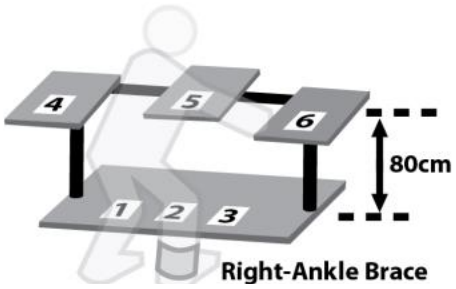


Figure 5.27 - Experiment Setup

This dataset was collected in a previous experiment examining the influence of unilateral ankle immobilization on low-back loading and injury potential during lifting [8]. Using an optoelectronic motion capture system, the positions and orientations of the feet, shanks, thighs, pelvis, and trunk of 10 male participants were captured while they performed lifting tasks. This dataset consisted of 24 DoF Cartesian time series vectors. Standard inverse kinematics computations were used to convert the positions into Cardan joint angles for the ankles, knees, hips, and lumbar spine. The joint angles formed a 21 DoF joint-angle model.

With and without their right ankle immobilized, participants lifted two masses (light = 3.7 kg; heavy = 12.7 kg) from three different origins (positions 1, 2, and 3) to three different destinations (4, 5, and 6) (**Figure 5.27**). Three repetitions of each task were performed. Ankle immobilization was achieved through the use of a brace designed to restrict ankle motion in all three anatomical planes.

We chose to use the 1-to-4, 1-to-6, 3-to-4, and 3-to-6 sequence types for testing the algorithms. This dataset includes a total of 120 observation sequences per sequence type, where there are 30 motions for each of the 'Heavy & Brace', 'Heavy & No Brace', 'Light & Brace', and 'Light & No Brace' lifting motions. Since each participant produced 3 cycles of each type of motion, there are a total of 12 motions for each participant.

5.4.2 Proposed Framework Results

By running the targeted lifting dataset through our proposed framework, the motion tree was generated by using the specified sequences (shown below):

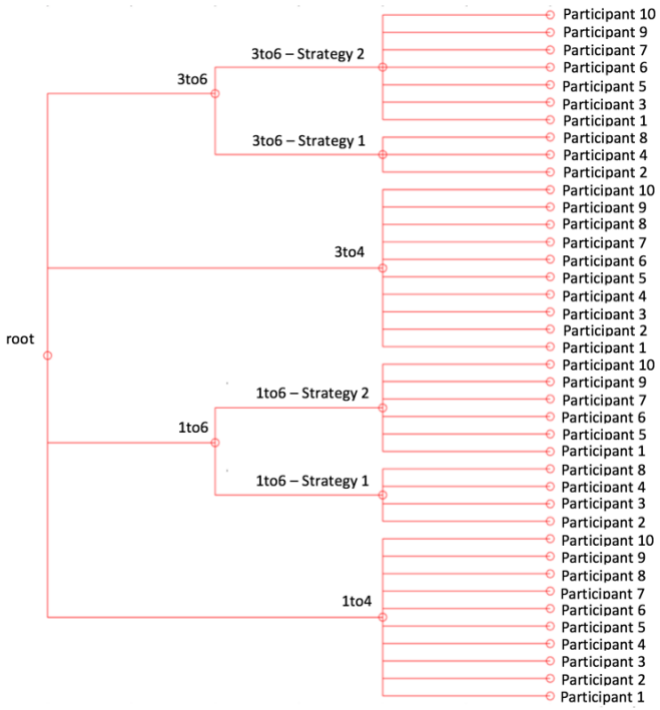


Figure 5.28 – Motion Tree for Targeted Lifting Motions Dataset (All Trajectories)

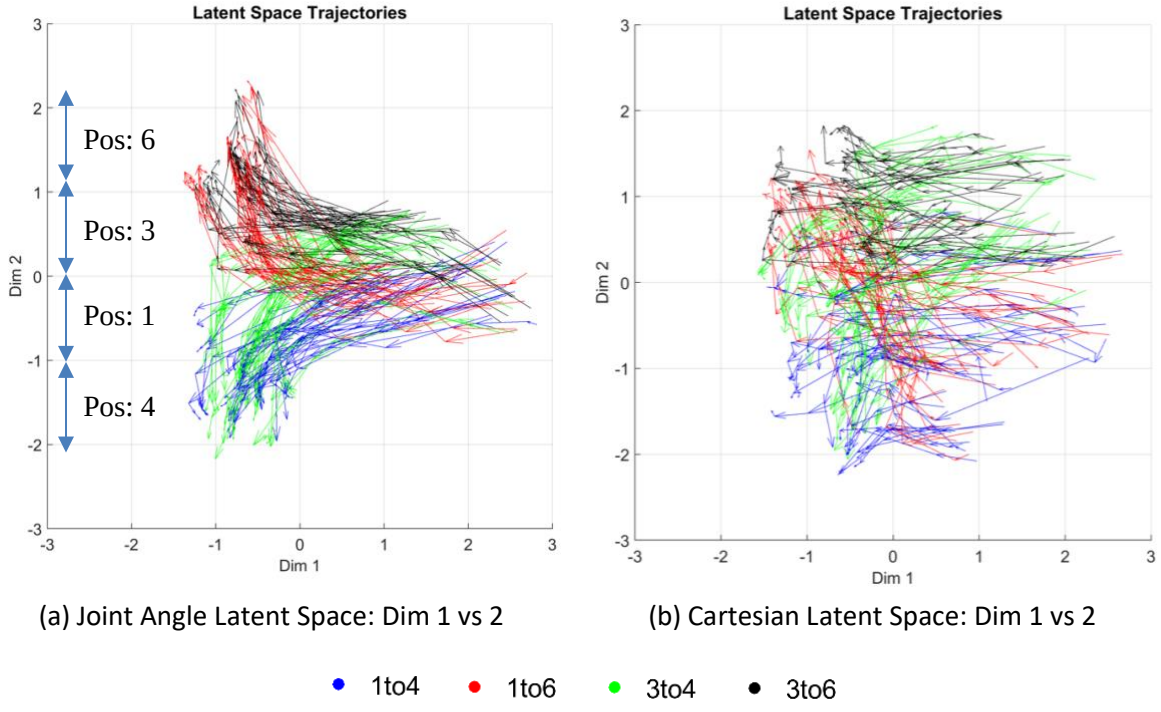
The highest node of the tree consisted of a latent space of 3 parameters and split into 4 strategies. The 4 main strategies corresponded to the 4 main movement sequences (1-to-4, 1-to-6, 3-to-4, and 3-to-6). Since these labels are known apriori, we can validate their accuracy through the use of K-Fold cross validation. The table below shows the average result for joint angle data as well as cartesian coordinates with K=5.

Table 5.8 – Results of Proposed Framework on Targeted Lifting Motion Dataset

Dataset Type									
Joint Angle					**Cartesian Coordinates**				
	1to4	1to6	3to4	3to6		1to4	1to6	3to4	3to6
1to4	118	0	2	0	1to4	109	2	5	4
1to6	0	120	0	0	1to6	0	111	1	8
3to4	0	0	120	0	3to4	10	2	107	1
3to6	0	3	0	117	3to6	0	3	2	115

Classification results had higher accuracy for the joint angle dataset in comparison to the cartesian coordinates. Part of this loss in accuracy can probably be attributed to the fact that cartesian trajectories can be impacted by participant heights. **Figure** 5.29 shows the latent space trajectories for joint angles and cartesian coordinates that were clustered above. Comparing **Figure** 5.29 (a) and (b) we can see that the trajectories in the joint angle latent space form tighter clusters in comparison to the cartesian counterparts.

Focusing on **Figure** 5.29 (a), it can be seen that dimension 1 captures information about the vertical displacement of the movement, where a value of ≈2 indicates the beginning of the motion and the value of ≈-1 indicates the end of the motion. Dimension 2 captures the horizontal movement of the lift, where the positions from left-to-right in **Figure 5.27** are numbered 4, 1, 3, and 6. Since the end effector paths do intersect each other for certain motion sequences (i.e. 1-to-6 intersects 3-to-4), we see those corresponding intersections happening in the latent space as well. In general, dimensions 1 and 2 capture positional information. Dimension 3 on the other hand is latent parameter that seems to vary for all motions and thus probably contains information that is participant-specific. Due to the high number of sequences (480 in total), we will focus on one of the motion subsets to understand participant-specific variations in more detail.



(a) Joint Angle Latent Space: Dim 1 vs 2

(b) Cartesian Latent Space: Dim 1 vs 2

Figure 5.29 – Latent Space Trajectories for All Movements

In the following, we focus on analyzing only one of the 4 movement sequences in further detail. We arbitrarily chose movement sequence 1-to-4 to do this analysis.

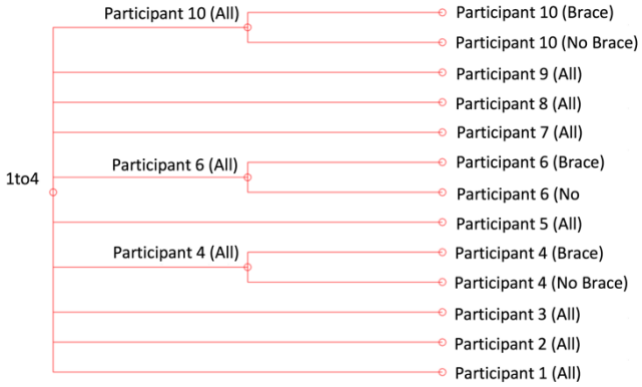
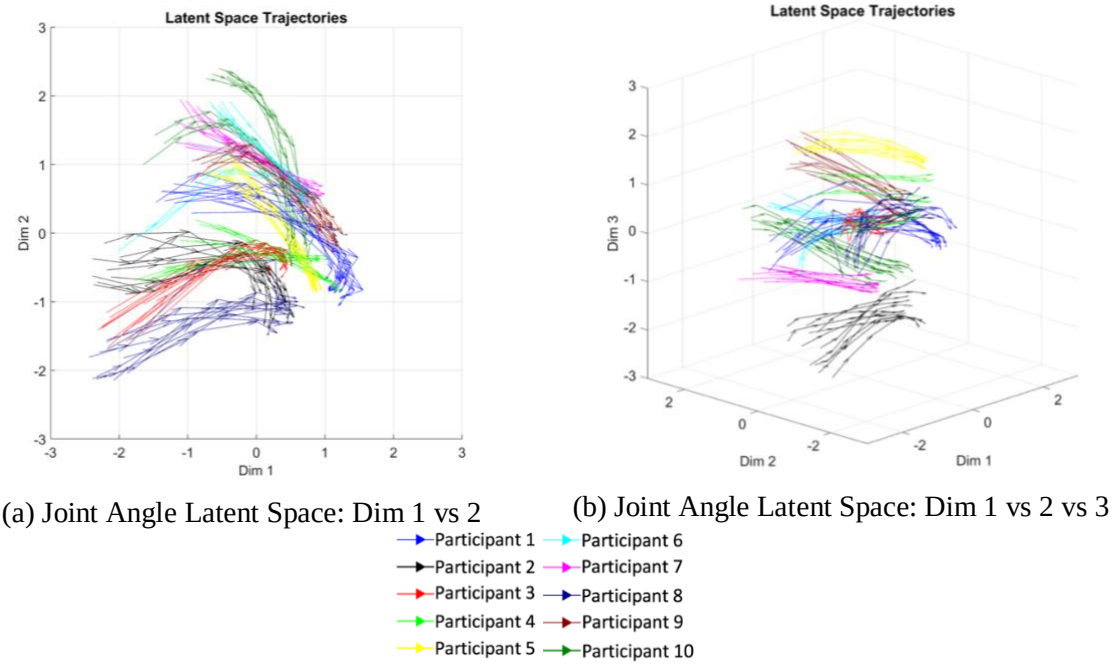


Figure 5.30 –Motion Tree for Targeted Lifting Motions Dataset (Only 1-to-4 Trajectories)

When the motion tree was created the children nodes of the node 1-to-4 were participant specific nodes. While both joint angle and Cartesian coordinate trajectories show this trend, the distance between participant trajectories is not consistent between the two, indicating that the choice of coordinate space may impact the trends that can be extracted from the data. For our following analysis we will focus on using joint angle trajectories to highlight the utility of our proposed approach. The latent space trajectories for this are shown in the figure below.



(a) Joint Angle Latent Space: Dim 1 vs 2

(b) Joint Angle Latent Space: Dim 1 vs 2 vs 3

Figure 5.31 - Latent Space Trajectories for Movements 1-to-4

We can see from **Figure** 5.31 that the movement sequences form clusters that are participant specific, where similar participant trajectories get clustered together. Next, we will show how each latent variable (or dimension) impacts the movements.

Dimension 1

Dimension 1 appears to map to movement progression. All movements have variation along this dimension, starting at a value of {-2, -1} and then progressing to a value of ≈1. Note that for the parent node we observed that dimension 1 was related to vertical displacement, which appears to be the case for the child node (or sub-cluster) as well. The starting point correlates to a starting lift position, i.e. the more negative the dim 1 value is the more the participant's pelvis is lower in their initial state.

Dimension 2

For dimension 2, we see that the various participants start off from a large range of values (roughly -2 to 2), but as the movement progresses the values seem to converge to a smaller range around 0. The figure below shows the initial lifting states from different participants (as viewed from above). These movements have a constant dim1 value, but a varying dim2 value. At a value of dim2 = -2, we find that lifting motion starts in a fairly symmetrical position with both knees equidistant from the body center. As the value of dim2 increases, the initial starting state of the lift changes such that left knee is closer to the body center (hence more weight on the left knee). Since the movement 1-to-4 requires the participant to rotate to lift the box, dimension 2 influences how much the participant rotates their lower body (vs upper body) initially to perform the lift.

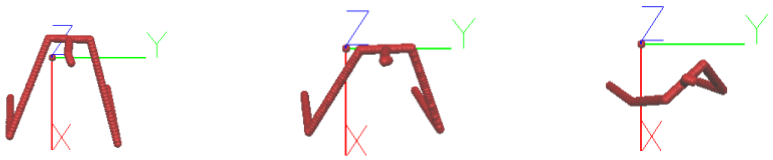


Figure 5.32 – Movements for Varying Values of Dimension 2

To understand the impact of dim-2, we used the proposed DoF analysis methodology to determine which joints were most impacted by this dimension. We selected participants that were at extreme values of dim2 (i.e. near -2 and +2) to compare. The resulting DoF comparison (shown below) shows that the hips are the most impacted by the dim-2, which aligns with our observations of the hips twisting

as the value of dim-2 changes. Additionally, we see that the knee and ankle joints progressively get less impacted as they are closer to the ground.

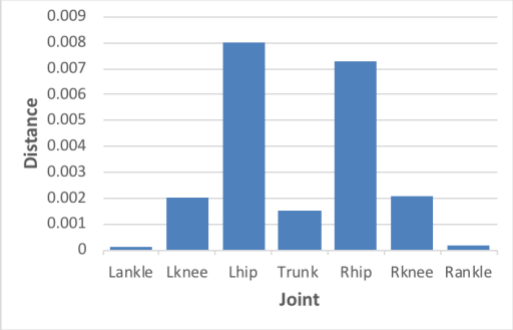


Figure 5.33 – Included DoF Comparison For Extreme Dim-2 Participants

For further validation, we did a correlation analysis of the latent variable dim-2 against the cartesian joint angles. Here we found that right hip and left hip had larger correlation values for the x direction (right being positive correlation because the right hip moves forward, while the left hip has negative correlation because it moves in a backward direction on the x-axis). We find in general that the left-side joints have slightly higher correlation values than the right-side joints and this may be a result of the fact that the left leg under goes more movement whereas the right leg does not move as much.

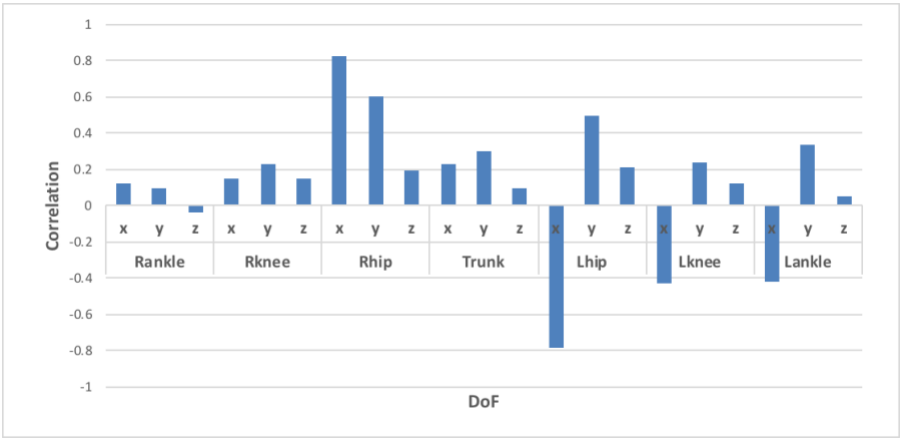
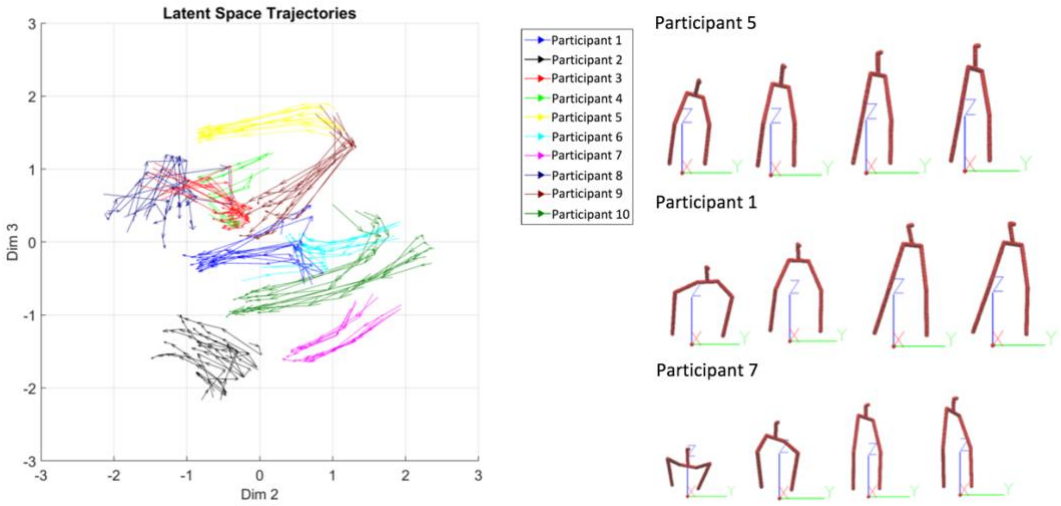


Figure 5.34 - Correlating Latent Space Variables with DoF for Dim-2

Dimension 3

Dimension 3 is a latent variable where the temporal variability within a participant's trajectory is small in the direction of dimension 3 for most participants (i.e. it is a latent variable that remains fairly

constant across the duration of the movement) (see **Figure** 5.35 **(a)**). **Figure** 5.35 **(b)** shows 3 different participants that have varying values of Dim 3. We can infer that this dimension correlates with the differences between Squat/Stoop motions. Moving from Participant 5 to Participant 1 to Participant 7, the value of Dim 3 decreases, and movements go from being stooping to squatting.



(a) Joint Angle Latent Space: Dim 2 vs 3 (b) Movements with Varying Dim 3

Figure 5.35 – Impact of Dim 3 on Movement Variability

To further validate that dim-3 impacts the squat-stoop strategy of the movement, we did a DoF analysis comparing movements that were at the extreme values of dim-3. The figure below shows the results of this analysis and we found that the results are similar to the results shown in **Figure** 5.14 from our Squat-Stoop study where the strategies were known apriori.

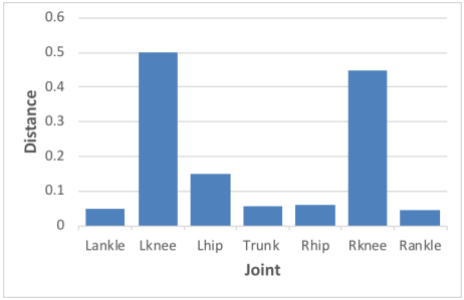


Figure 5.36 – Included DoF Comparison For Extreme Dim-3 Participants

Additionally we did a correlation analysis (**Figure** 5.37) using the latent variables and the cartesian coordinates of the movement trajectories to see if the results would be comparable to what we had

observed in **Figure** 5.15. We found that the results were similar, where there is strong correlation (>80%) between the latent variable dim-1 and shank/thigh z-axis movement and as well as shank y-axis movement.

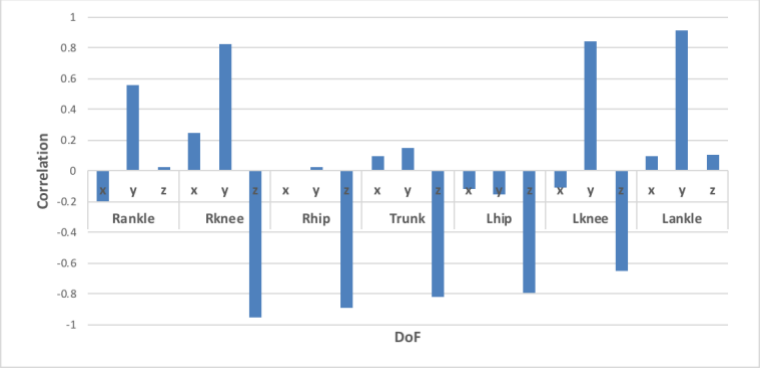
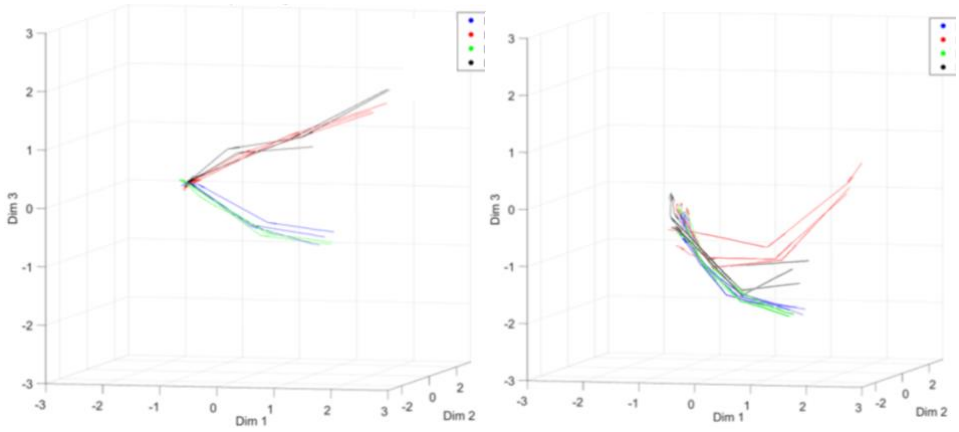


Figure 5.37 - Correlating Latent Space Variables with DoF for Dim-3

Impact of Bracing on Movements

Given an interpretation of the latent dimensions, **Figure** 5.38 shows that it is possible to see what impact bracing and/or increasing the weight has on the participants. **Figure** 5.38 illustrates the latent space trajectories for those participants who changed their strategy from squat to stoop when the bracing constraints were changed. In particular it appears that when participants go from having no brace to having a brace, they will go from squatting to stooping to lift the weight. This trend is observable with participant 4 regardless of the weight, but for participant 6 the weight has an impact on this change in strategy (for only heavy weight scenarios stooping strategy is adopted).



(a) Participant 4 (b) Participant 6

• Heavy, Brace • Heavy, No Brace • Light, Brace • Light, No Brace

Figure 5.38 – Impact of Weight and Bracing on Movements – Latent Space Trajectories

5.4.3 JCV Results

Because of the nature of the lifting motions in this dataset, there is no guarantee that both feet will be planted on the ground during the lift. The JCV approach relies on defining a close-chain kinematic model of the movement with which the joint contributions can be calculated. For this particular dataset this is not feasible.

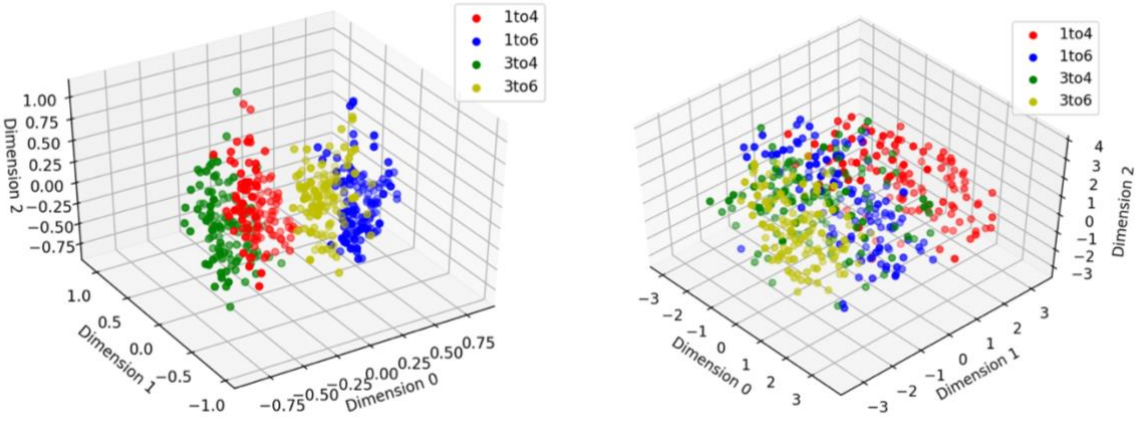
5.4.4 Kinematic Synergy Results

For the kinematic joint synergy approach, the VAF metric was used to determine the ideal number of synergies for the dataset. For the squat-stoop dataset this was discovered to be 3 synergies.

The results of running classification analysis using k-fold validation on the kinematic synergy algorithm are presented in **Table 5.9**. Note that this algorithm was tested on joint angle trajectory data as well as Cartesian trajectory data to determine its versatility.

Table 5.9 – Results of Kinematic Synergy Detection on Targeted Reaching Motion Dataset

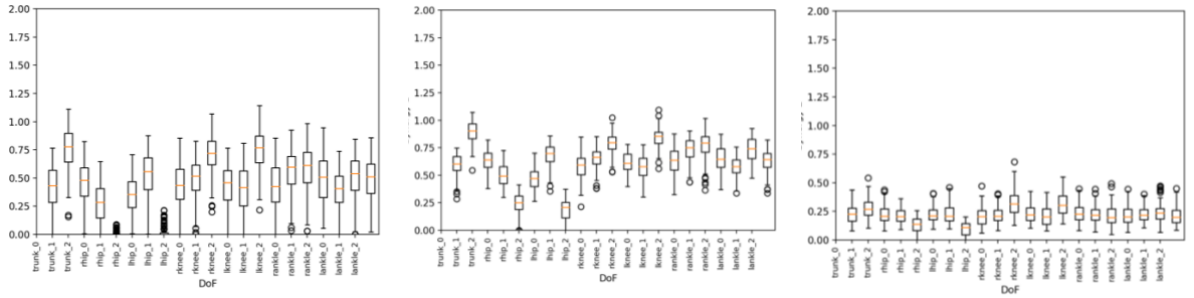
Dataset Type									
Joint Angle					**Cartesian Coordinates**				
	1to4	1to6	3to4	3to6		1to4	1to6	3to4	3to6
1to4	108	0	9	3	1to4	96	5	10	4
1to6	0	102	0	18	1to6	8	83	7	22
3to4	11	0	99	0	3to4	19	10	73	18
3to6	2	7	0	111	3to6	3	12	2	103



(a) Joint Angle Data (b) Cartesian Coordinates

Figure 5.39 – Kinematic Synergy Detection MDS Plots for All Movements

For this particular dataset, the kinematic synergy approach performs better with joint angle data than with Cartesian data. Looking at the MDS plots for both cases (**Figure** 5.39), we can see that the data is more easily separable in the joint angle data case, with some overlap between 1-to-4 / 3-to-4 and 1-to-6 / 3-to-6.



(a) Synergy 0 (b) Synergy 1 (c) Synergy 2

Figure 5.40 – Synergy Plots for Kinematic Synergy Detection

The synergy plots (**Figure** 5.40) for this dataset show an almost equal amount of variation across all joints (with the exception of one of the hip angles) making it challenging to extract any useful information from the synergy values themselves. Additionally, upon examining any of the sequence types, we expect to find clusters of movements grouped by participants (since this is what the proposed approach discovered). Instead when we look at the MDS plots of the synergies, we find that it isn't possible to distinguish by participants (see **Figure** 5.41).

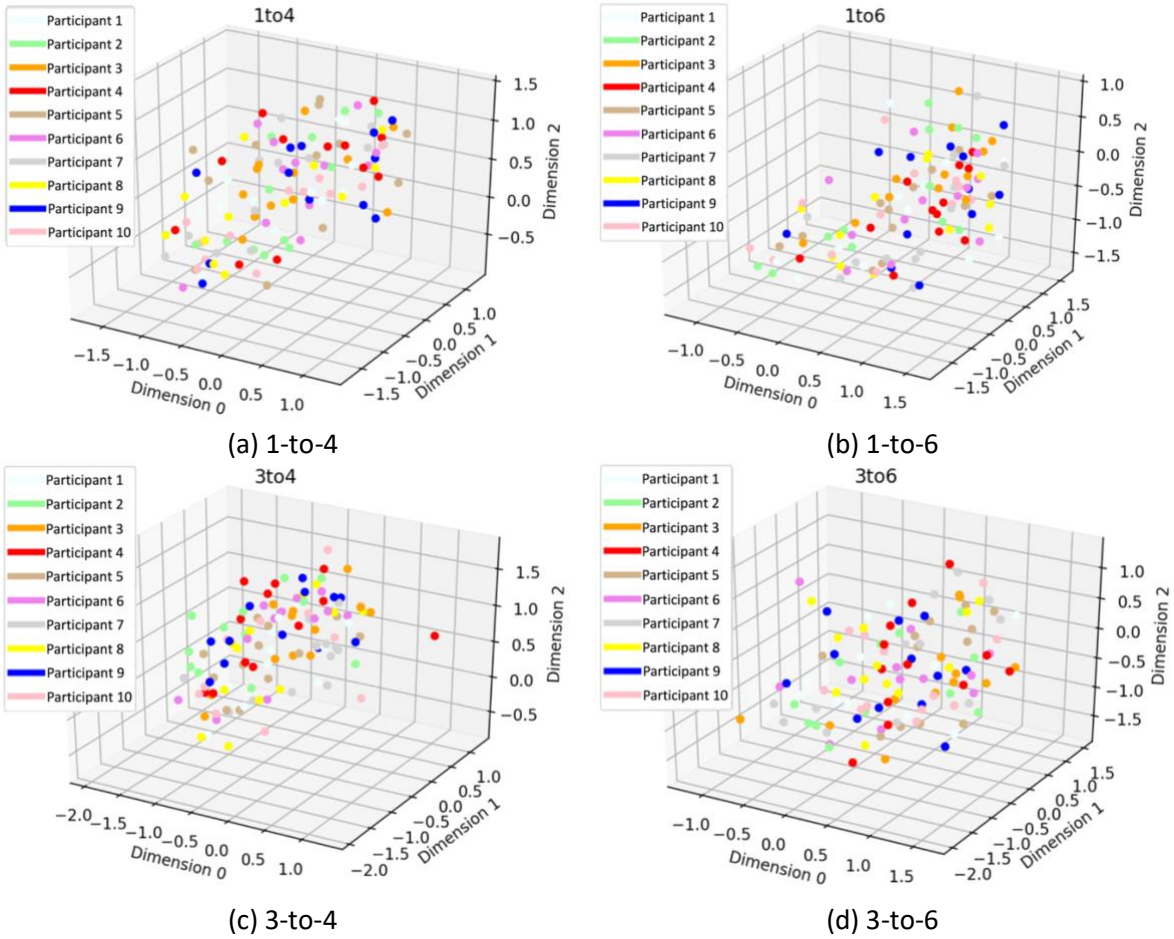
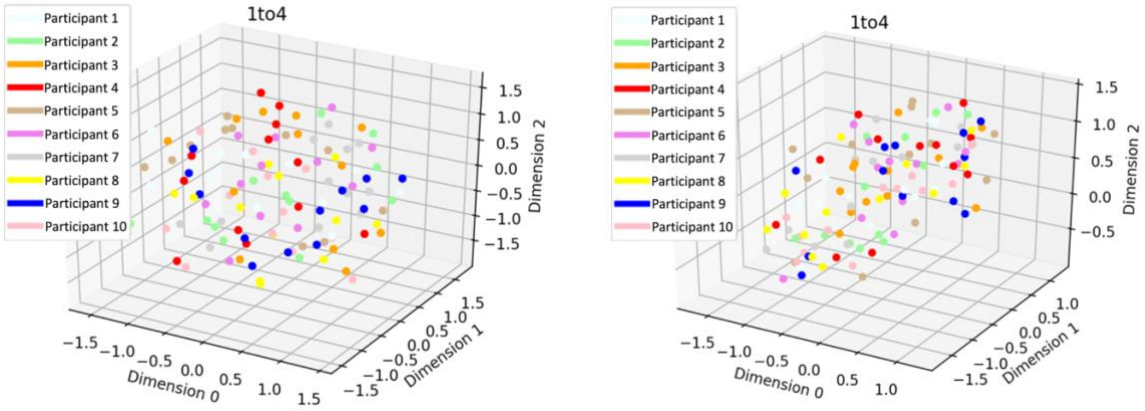


Figure 5.41 – Kinematic Synergy Detection MDS Plots By Movement Sequence Type

Note that **Figure** 5.41 was generated using a global synergy space trained on all motions. In our proposed approach we found that if we generated a new latent space for sub-strategies, the accuracy of our models was better in comparison to a global latent space. To investigate whether this might also hold true for synergy spaces we generated the synergy space for sequence 1-to-4 trained with only those sequences. **Figure** 5.42 shows a comparison of global synergy space and the local synergy space.



(a) Local synergy space (b) Global synergy space

Figure 5.42 – Comparison of Local and Global Synergy Space for Movement Sequence 1-to-4

We find that even with a local synergy space, the kinematic synergy approach isn't able group participant-specific movements together like the proposed approach.

5.4.5 Discussion

Comparing the results from the proposed approach and the comparison approach of the kinematic synergy detection, we find that the purposed approach and the comparison approach were both able to detect the high-level movement sequences (1-to-4, 1-to-6, 3-to-4, 3-to-6). The difference was that the proposed approach was able to do so with higher accuracy. Going beyond simply identifying the labeled parameters, the proposed approach was able to further identify similarities between participant movements and how they related to each other with the use of latent parameters. The comparison algorithm on the other hand was unable to identify any of those same trends at the participant level. This seems to indicate that, while the kinematic synergy approach is good for analyzing macro changes in motion, when it comes to analyzing more subtle differences in motions it may not perform as well.

From a runtime performance aspect, the kinematic synergy approach outperforms the proposed approach by almost a magnitude (see **Figure** 5.43). The final dataset studied here is far larger than the previous 2 datasets (synthetic dataset and squat/stoop dataset), and so we can see as the amount of data increases the proposed algorithm doesn't perform as well as its competitor. The performance gains of the competing algorithms come at the cost of the accuracy that previously described.

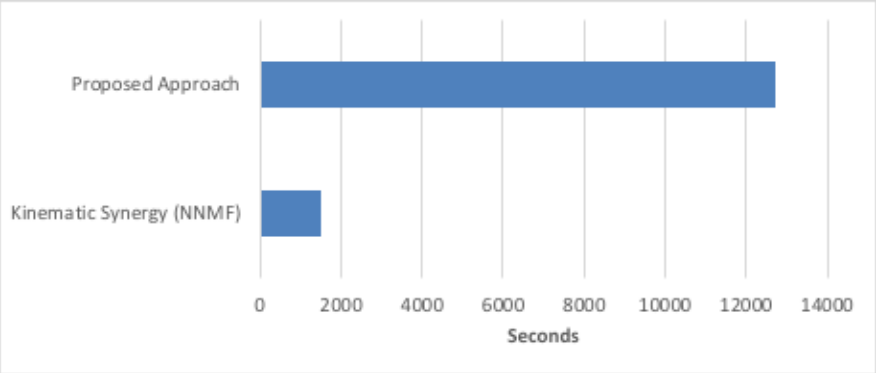


Figure 5.43 - Algorithm Training Time for Targeted Lifting Motion Dataset

5.4.6 Proposed Approach Investigations: Learning Individual Latent Spaces for Sub-Clusters

In the proposed approach, as the divisive clustering algorithm generates a tree of motions, it generates a latent space for each tree node (child cluster of motions) where further divisive clustering is done in the tree node's latent space. An alternative to this approach is to do all the clustering in the latent space that is learned by training with all the observation sequences, i.e. the global latent space or the very first latent space we learn for the root node of our divisive clustering tree. In this section we show why the proposed approach is better than the alternative approach of using the global latent space.

In order to analyze this further we compared the average reconstruction error of generating all the sequences in cluster 1to6 (from **Figure** 5.28) using both approaches. The result of this comparison is shown in the following figure:

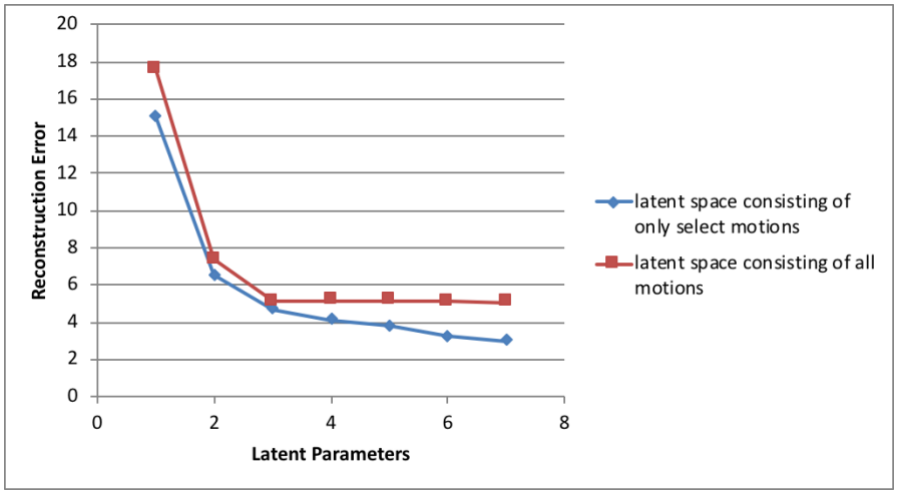


Figure 5.44 – Reconstruction Error for Global vs Local Latent Space

We find that a localized latent space is able to achieve lower reconstruction errors for the same number of latent space parameters. This indicates that the optimization process used to create a global latent space for all sequences results in loss of information when it comes to recreating the original training motions. By learning a latent space for sub-clusters (like we do in our approach), motion models at various levels of tree can learn a latent space that is relevant for grouping the motions at that level in the tree and as we proceed lower down the tree our models retain more of the details pertaining to the motion. The targeted reaching movement dataset may not illustrate the full strength of this approach, since all of the motions are similar lifting motions. In scenarios where the movements vary a lot (for example kicks vs punches), higher level latent space would have latent parameters that are relevant for comparing kicks vs punches, where lower level latent space parameters would be relevant for just comparing different kinds of kicks or different kinds of punches.

Chapter 6
Conclusions and Future Work

6.1 Conclusions

In this work we propose a novel framework for detecting human motion strategies using Gaussian Process Dynamical Models (GPDM) and Hidden Markov Models (HMM). GPDMs were used to convert human motion trajectories from high dimensional representations to low dimensional trajectories. The low dimensional trajectories are then used to train HMMs and do hierarchical clustering of HMMs to generate clusters of time series datasets organized in a motion tree. The resulting nodes (or clusters) in the motion tree were compared to understand the differences between the clusters, including understanding how variations in GPDM latent variables could translate to variations in the original high dimensional space. The result is an unsupervised algorithm that allows us to automatically determine what movement strategies were adopted to perform the specified task and understanding how those strategies differed. To verify the validity of our approach, we tested the algorithm and two competitive algorithms (Joint Contribution Vector and Kinematic Joint Strategy Detection) in three different datasets.

The first experiment consisted of a synthetic data set where controlled the motion variation as well as the degree of difference between two strategies. We found that our approach was the most accurate as noise in the dataset was artificially increased, followed by the Kinematic Joint Strategy approach and the JCV algorithm. Additionally, we were able to show that for a simplified version of the synthetic trajectory, the proposed approach allowed us to use latent variables to determine the degree of difference between movement strategy between two motions. However, the proposed approach was the slowest as the computational complexity with training GPDMs and HMMs is higher than the computations involved in the competitive approaches.

The second experiment used real data consisting of squat lifts and stoop lifts where the two main strategies were known a-priori. We found that all algorithms were able to detect the main strategies accurately, but the divisive algorithm was the most accurate at identifying the sub-strategies that were unknown. The performance results of the algorithm were the same as before, where the proposed approach was the slowest of the three algorithms being compared. We were able to show that by using the low dimensional representations generated by GPDM we were able to speed up the clustering algorithm in comparison to an approach that only used HMMs trained with high dimensional trajectories.

In the third experiment we had 4 different movements (1-to-4, 1-to-6, 3-to-4, 3-to6) where the strategies within each movement were unknown a-priori and so the algorithms needed to automatically determine the strategies. In this experiment the JCV algorithm could not be used because of algorithm limitations that require an open loop kinematic structure, so only the kinematic joint strategy approach and the proposed approach were compared. We found that the kinematic strategy approach was able to identify high level movements but was unable to find any strategies within the movement types. In comparison the proposed approach was able to find strategies and we were able to use the latent space variables to help us understand differences between the strategies. Additionally, we were able to determine the impact of known variables (like weight or ankle brace) on the movement strategy that was adopted by the participant.

Out of the three approaches that were compared, the proposed approach does the best job of creating a simplified visualization of the dataset where the latent parameters translate to physically relatable traits. This became especially apparent for high dimensionally datasets where the low dimensional representations in the latent space help visualize the time series clusters. Correlating the latent variables to high dimensional variables also helped us to discover insights as to what made the movement strategies different. The proposed approach was not as fast as competitive algorithms due to its computational complexity.

6.2 Future Work

There are several areas of improvement that the proposed framework could benefit from with further investigation: runtime performance, clustering flexibility, and further testing.

From a runtime performance aspect, the proposed approach was not as fast as comparison approaches. One suggestion to improve performance would be to investigate if it is possible to dynamically change the amount of subsampling done on trajectories and to dynamically reduce the number of states for HMMs. The root node of the tree could use more sparsely subsampled trajectories and HMM models with low number of states to reduce the amount of computations, while lower nodes in the tree would increasingly use more states or more samples from the trajectories. The rationale is that models closer to the root node represent generalized models and thus details are not as important but as we progress further down the tree details become more important. Determining the exact number of samples or states could be done based on optimizing reconstruction errors at each level of the tree.

The proposed approach at its core assumes hard cluster boundaries in order to generate the motion tree. This means that once a movement is assigned to a cluster, from there on it is assumed that the

movement is of the strategy associated with the cluster. Any sub-clusters that are generated from there on are only assumed to be sub-strategies of the parent and will only consist of motions in the parent cluster. In theory, movements could adopt various degrees of different strategies (this is something that the kinematic strategy approach assumes). In our proposed approach we tried to overcome this limitation of hard clusters by allowing latent space analysis of all motions of sibling clusters. But this approach may not be as flexible as an approach that uses fuzzy clustering to determine the strength of membership that one movement may have to a cluster. Hard clustering allows us to reduce computational complexity and allows analysis of movements to be easier by ignoring movements assumed to be no longer relevant to a strategy. Fuzzy clustering could potentially allow more expressive representation of what strategies describe a movement at the cost of computational complexity and would require additional methodologies to help with determine which cluster memberships are important.

Additionally, it would be interesting to validate the proposed approach on more datasets. In this work, we explored the use of motion capture data for movements that are relatively similar as we were interested in detecting subtle differences in human movements. It would be interesting to extend studies to movements that are very different and analyze the results in terms of tree structure and how latent space variables describe differences between very different movements. Additionally, it would be interesting to test the proposed approach on other kinds of signals, like EMGs to determine if there is sensitivity to the data signal domain. While we did show in our synthetic dataset that the proposed approach has some resiliency to noise it would be interesting to see how it fares against signal characteristics in different domains.

References

[1] M. U. Choudry, "Detecting changes in human motion using stocastic distance measures," in *IEEE EMBC*, 2011.

[2] M. U. Choudry, "A stochastic framework for movement strategy identification and analysis," *IEEE Trans of Human Machine Systems*, vol. 43, p. 315, 2013.

[3] "JAHMM," 2009. [Online]. Available: https://code.google.com/archive/p/jahmm/downloads.

[4] J. M. Wang, D. J. Fleet and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 2, pp. 283-298, Feb 2008.

[5] W. Park, "Representing and identifying alternative movement techniques for goal-directed manual tasks," *Journal of Biomechanics*, vol. 38, pp. 519-527, 2005.

[6] N. Lambert-Shirzad, "Upper-body motion coordination after stroke: insights from kinematic and muscle synergies," University of British Columbia, Vancouver, 2017.

[7] T. A. C. Beach, "Firefighter fitness, movement qualities, occupational low-back loading demands and injury potential," Doctoral Dissertation, University of Waterloo, Waterloo, 2012.

[8] T. A. C. Beach, "Does a unilateral restriction in ankle mobility affect trunk kinematics and low-back loading during manual lifting tasks?," in *American Society of Biomechanics Conference*, 2009.

[9] P. Levinger, "Support Vector Machines for detecting recovery from knee replacement surgery using quantitative gait measures," in *Proceedings of the 29th Annual International Conference of The IEEE EMBS*, 2007.

[10] D. Kulic, G. Venture and Y. Nakamura, "Detecting changes in motion characteristics during sports training," in *31st Annual International Conference of the IEEE EMBS*, Minneapolis, Minnesota, USA, 2009.

[11] W. Ilg, "Estimation of skill levels in sports based on hierarchical spatio-temporal correspondence," in *DAGM Symposium*, 2003.

[12] M. J. O'Malley, "Fuzzy Clustering of Children with Cerebral Palsy Based on Temporal-Distance Gait Parameters," *IEEE Transactions on Rehabilitation Engineering*, vol. 5, no. 4, 1997.

[13] G. Vannozzia, "A neurofuzzy inference system based on biomechanical features for the evaluation of the effects of physical training," *Computre Methods in Biomechanics and Biomedical Engineering*, vol. 11, no. 1, 2008.

[14] S. Slaugter, "Quantifying and learning human movement characteristics for fall prevention in the elderly using inertial measurement units and neural networks," in *ICERI Proceedings*, 2009.

[15] R. Bartlett, "Artificial intelligence in sports biomechanics – New dawn or false hope?," *Sports Science and Medicine*, vol. 5, pp. 474-479, 2006.

[16] J. K. Aggarwal and Q. Cai, "Human motion analysis: A review," *Computer Vision and Image Understanding*, vol. 73, pp. 428-440, 1999.

[17] L. Wang, H. Wu and T. Tan, "Recent developments in human motion analysis," *Pattern Recognition*, vol. 36, no. 3, pp. 585-601, 2003.

[18] S. Schaal, "Dynamic movement primitives a framework for motor control in humans and humanoid robotics," in *Adaptive Motion of Animals and Machines*, 2006, pp. 261-280.

[19] D. Kulic, "Incremental Learning, Clustering and Hierarchy Formation of Whole Body Motion Patterns using Adaptive Hidden Markov Chains," *IJRR*, vol. 27, no. 7, pp. 751-784, 2008.

[20] O. C. Jenkins and M. J. Mataric, "Deriving action and behavior primitives from human motion data".

[21] T. B. Moeslund and E. Granum, "A Survey of Computer Vision-Based Human Motion Capture".

[22] T. B. Moeslund, A. Hiltonb and V. Krugerc, "A survey of advances in vision-based human motion caputre and analysis".

[23] H. Zhou and H. Hu, "Human motion tracking for rehabilitation - A survey," *Biomedical Signal Processing and Control*, vol. 3, no. 1, pp. 1-18, 2008.

[24] R. Bartlett, J. Wheat and M. Robins, "Is movement variability important for sports biomechanics?," *Sports Biomechanics*, vol. 6, pp. 224-243, 2007.

[25] T. Chau, "A review of analytical techniques for gait data. Part 1: Fuzzy, statistical and fractal methods," *Gait and Posture,* vol. 13, pp. 49-66, 2001.

[26] T. Chau, "A review of analytical techniques for gait data. Part 2: neural network and wavelet methods," *Gait and Posture,* vol. 13, pp. 102-120, 2001.

[27] D. T. H. Lai, R. K. Begg and M. Palaniswami, "Computational Intelligence in Gait Research: A Perspective on Current Applications and Future Challenges," *IEEE Transactions on Information Technology in Biomedicine,* vol. 13, no. 5, pp. 687-702, 12 May 2009.

[28] R. Begg and J. Kamruzzaman, "Neural networks for detection and classification of walking pattern changes due to aging," *Australian Physical and Engineering Sciences in Medicine,* vol. 29, no. 2, 2006.

[29] A. J. Silva, "The use of neural network technology to model swimming performance," *Journal of Sports Science and Medicine,* vol. 6, pp. 117-125, 2007.

[30] R. Begg and J. Kamruzzaman, "A machine learning approach for automated recognition of movement patterns using basic, kinetic and kinematic gait data," *Journal of Biomechanics,* vol. 33, pp. 401-408, 2003.

[31] R. K. Fukuchi, B. M. Eskofier, M. Duarte and R. Ferber, "Support vector machines for detecting age-related changes in running kinematics," *Journal of Biomechanics,* vol. 44, pp. 540-542, 2011.

[32] T. E. Murphy, "Towards Objective Surgical Skill Evaluation with Hidden Markov Model-based Motion Recognition," John Hopkins University, 2004.

[33] M. Chen, B. Hunag and Y. Xu, "Human abnormal gait modeling via hidden markov model," in *IEEE ICIA*, 2007.

[34] S. Armand, E. Watelain, E. Roux, M. Mercier and F. Lepoutre, "Linking clinical measurements and kinematic gait patterns of toe-walking using fuzzy decision trees," *Gait Posture,* vol. 25, no. 3, pp. 475-484, 2007.

[35] T. W. Liao, "Clustering of time series data – A survey," *Pattern Recognition,* vol. 38, pp. 1857-1874, 2005.

[36] B. Morris and M. Trivedi, "Learning Trajectory Patterns by Clustering: Experimental Studies and Comparative Evaluation," in *IEEE CVPR*, 2009.

[37] A. K. Jain, M. N. Murthy and P. J. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, 1999.

[38] F. Guoliang, Z. Xin and M. Ding, "Gaussian process for human motion modeling: A comparative study," in *2011 IEEE International Workshop on Machine Learning for Signal Processing*, 2011.

[39] B. Torro, C. J. Nester and P. C. Farren, "Cluster analysis for the extraction of saggital gait patterns in children with cerebral palsy," *Gait & Posture*, vol. 25, no. 2, pp. 157-165, 2007.

[40] C. Giacomozzi, F. Martelli, A. Nagel, A. Schmiegel and D. Rosenbaum, "Cluster analysis to classify gait alterations in rheumatoid arthritis using peak pressure curves," *Gait Posture*, vol. 29, no. 2, pp. 220-224, 29 Feb 2009.

[41] J. M. O'Byrne, A. Jenkinson and T. M. O'Brien, "Quantitative analysis and qualification of gait patterns in cerebral palsy," *Journal of Child Neurology*, vol. 13, no. 3, pp. 101-108, 1997.

[42] G. Donà, "Principal Component Analysis for Motor Skills Characterization and Individual Monitoring in Sports Science," Universita' Di Padova, 2008.

[43] G. Steven and P. J. McNair, "Abdominal and erector spinae muscle activity during gait: the use of cluster analysis to identify patterns of activity".

[44] R. A. Johnson and D. W. Wichern, Applied Multivariate Statistical Analysis, Upper Saddle River, NJ: Prentice-Hall, 1998.

[45] M. O. A. E. Menceur, "An automatic method to identify human alternative movements," *Journal of Automation, Mobile Robotics, and Intelligent Systems*, vol. 3, no. 3, pp. 64-66, 2009.

[46] M. Kulbacki, J. Segen and A. Bak, "Unsupervised learning motion models using dynamic time warping," in *Symposium on Intelligent Information Systems*, 2002.

[47] M. Gueguin, "Clustering follow-up time series recorded by cardiac implantable devices," in *IEEE EMBC*, 2007.

[48] T. Nakamura and K. Makio, "Discovering and Translating Skills from Motion Data," Kobe University, Japan, 2006.

[49] M. Ester, H. P. Kriegel, J. Sander and X. Xu, "A density based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining*, 1996.

[50] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286.

[51] R. C. Vlasko, J. A. El-Jaroudi and J. R. Boston, "Application of Hidden Markov Model Topology Estimation to Repetitive Lifting Data," in *IEEE ICASSP*, 1997.

[52] C. Li and G. Biswas, "Applying the hidden markov model methodology for unsupervised learning of temporal data," *International Journal of Knowledge-based Intelligent Engineering Systems*, vol. 6, no. 3, pp. 152-160, 2002.

[53] M. Butler, "Hidden Markov Model Clustering of Acoustic Data," University of Edinburgh, 2003.

[54] N. D. Lawrence, "Learning for larger datasets with the gaussian process latent variable model," in *Proceedings of the 11th Conference on Artifical Intelligence and Statistics*, 2007.

[55] J. Hong, C. Changmook and S.-J. Kim, "Gaussian process gait trajectory learning and generation of collision-free motion for assist-as-needed rehabilitation," in *RAS 15th International Conference on Humanoid Robots*, 2015.

[56] J. Wang and T. Zielinska, "Gait features analysis using artificial neural networks - testing the footwear effect," *Acta of Bioengineering and Biomechanics*, vol. 19, no. 1, pp. 17-32, 2017.

[57] P. Gharani, B. Suffoletto, T. Chung and H. A. Karimi, "An artificial neural network for movement pattern analysis to estimate blood alchohol content level," *Sensors*, vol. 17, no. 12, pp. 1-13, 2017.

[58] J. Martinez, M. J. Black and J. Romero, "On Human Motion Prediction Using Recurrent Neural Networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[59] K. Fragkiadaki, S. Levine and P. Felsen, "Recurrent Network Models for Human Dynamics," in *The IEEE International Conference on Computer Vision (ICCV)*, 2015.

[60] J. Butepage, M. Black, D. Kragic and H. Kjellstrom, "Deep representation learning for human motion prediction and classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[61] Z. Che, X. He, K. Xu and Y. Liu, "DECADE: A Deep Metric Learning Model for Multivariate Time Serie," in *Workshop on Mining and Learning From Time Series*, Halifax, 2017.

[62] H. Coskun, D. J. Tan, S. Conjeti, N. Navab and F. Tombari, "Human Motion Analysis With Deep Metric Learning," 2018.

[63] P. Lamb, R. Bartlett and A. Robins, "Self-Organising Maps: An Objective Method for Clustering Complex Human Movement," *Internal Journal of Computer Science in Sport,* vol. 9, no. 1, pp. 20-29, 2010.

[64] S. Wu, z. Wang and S. Xia, "Indexing and retrieval of human motion data by hierarchical tree," in *Proceedings of the 16th ACM Symposium on Virtual Reality* , Kyoto, 2009.

[65] J. Perl, "A neural network approach to movement pattern analysis," *Human Movment Science,* vol. 23, pp. 605-620, 2004.

[66] J. H. Jackson, "On the comparative study of diseases of the nervous system," *British Medical Journal,* vol. 2, no. 1494, pp. 355-362, 1889.

[67] N. Bernstein, The co-ordination and regulation of movements, Oxford: Pergamon Press, 1967.

[68] A. M. Sabatini, "Identification of neuromuscular synergies in natural upper-arm movements," *Biological Cybernetics,* vol. 86, no. 4, pp. 253-262, 2002.

[69] J. J. Kutch and F. J. Valero-Cuevas, "Muscle redundancy does not imply robustness to muscle dysfunction," *Journal of Biomechanics,* vol. 44, no. 7, pp. 1264-1270, 2011.

[70] M. Coscia, V. C. Chenug, P. Tropea, A. Koenig, V. Monaco, C. Bennis, S. Micera and P. Bonato, "The effect of arm weight support on upper limb muscle synergies during reaching movements," *Journal of NeuroEngineering and Rehabilitation,* vol. 11, no. 22, pp. 1-15, 2014.

[71] K. M. Steele, M. C. Tresch and E. J. Perreault, "Consequences of biomechanically constrained tasks in the design and interpretation of synergy analyses," *Journal of Neurophysiology,* vol. 113, pp. 2102-2113, 2015.

[72] S. L. Chiu, "Fuzzy model estimation based on cluster estimation," *Journal of Intelligent and Fuzzy Systems,* vol. 2, pp. 267-278, 1994.

[73] J. C. Bezdek, "FCM: The Fuzzy C-means Clustering Algorithm," *Computers and Geosciences,* vol. 10, no. 2, pp. 191-203, 1984.

[74] A. Churbanov, "Implementing EM and viterbi algorithms for hidden markov model in linear memory," *BMC Bionformatics,* vol. 9, no. 224, 2008.

[75] R. E. Kalman, "Mathematical Description of Linear Dynamical Systems," *Journal of the Society for Industrial and Applied Mathematics,* vol. 1, no. 2, pp. 152-192, 1962.

[76] Z. Ghahramani and G. E. Hilton, "Parameter Estimation for Linear Dynamical Systems," University of Toronto, 1996.

[77] V. Pavlovic, J. M. Rehg and J. MacCormick, "Learning switching linear models of human motion," in *Proceedings of the 13th International Conference on Neural Information Processing Systems*, 2000.

[78] J. M. Wang, "Matlab GPDM Scripts," 1 Sept 2007. [Online]. Available: http://www.dgp.toronto.edu/~jmwang/gpdm/gpdm_code_090107.tar.gz.

[79] J. J. Hughlings, "On the comparative study of disease of the nervous system," *British Medical Journal,* pp. 355-362, 17 Aug 1889.

[80] N. A. Bernstein, The co-ordination and regulation of movements, Oxford: Pergamon Press, 1967.

[81] Y. Morita, "Analysis and modeling of upper limb motion during sanding training".

[82] P. Smyth, "Clustering Sequences with Hidden Markov Models," in *Advances in Neural Information Processing Systems*, MIT Press, 1997, pp. 648-654.

[83] S. R. Das, R. C. Wilson, M. T. Lazarewicz and L. H. Finkel, "Two-Stage PCA Extracts Spatiotemporal Features for Gait Recognition," *Journal of Multimedia,* vol. 1, no. 5, pp. 9-17, 2006.

[84] T. Nakamura, K. Makio and K. Ueharal, "Discovering and translating skills from motion data," Kobe University, 2006.

[85] M. Ito and J. Tani, "On-line imitative interaction with a humanoid robot using a dynamic neural network model of a mirror system".

[86] W. Gillard, "Detecting trunk motion changes due to pregnancy using pattern recognition techniques," in *30th Annual International IEEE EMBS Conference*, Vancouver, 2008.

[87] B. Blundell, "Bayesian Rose Trees," in *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*, 2010.

[88] S. McGill, "Evolving ergonomics?," *Ergonomics,* vol. 52, pp. 80-86, 2009.

[89] B. H. Juang and L. R. Rabiner, "A probabilistic distance measure for Hidden Markov Models," *AT&T Technical Journal,* vol. 64, no. 2, pp. 391-408, 1985.

[90] Y. Zhao and G. Karypis, "Evaluation of hierarchical clustering algorithms for document datasets," University of Minnesota, Minneapolis, MN, USA, 2002.

[91] B. Auvinet, "The interest of gait markers in the identificaiton of subgroups among fibromyalgia patients," *BMC Musculoskeletal Disorders,* vol. 12, no. 258, 2011.

Appendix A

Synthetic Dataset Configurations