


An Incremental Learning System for Interactive Gestural Control of Sound Synthesis

By

STACY SHU-YUAN HSUEH

Submitted in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

UNIVERSITÉ PIERRE ET MARIE CURIE

JULY 2015

Supervisors:
Frédéric BEVILACQUA
Jules FRANÇOISE

ÉQUIPE D'INTERACTION SON MUSIQUE MOUVEMENT
IRCAM – CENTRE POMPIDOU

Abstract

Gesture-controlled interactive systems are prevalent in the sound computing community. They have given rise to new performance paradigms and bridged the gap between sound research and other scientific disciplines by providing a rapid prototyping platform for designing gestures and respective sound outputs in application-specific contexts. In such systems, the *mapping* strategy from gestures to sounds is central to the versatility and adaptability of the design process. This thesis is concerned with improving the workflow for designing mappings between gesture features and sound synthesis parameters. To that end, we develop an incremental system for learning the gesture-sound mappings. The system is divided into two modules: unsupervised automatic segmentation and incremental clustering.

ACKNOWLEDGEMENTS

First, I would like to thank my supervisors, Frédéric and Jules for giving me an opportunity to work in such a vibrant team for my thesis and showing me great support throughout my internship. The breadth and depth of their knowledge have inspired me in every aspect of this research. It has been a joy and privilege working with them. I am also grateful for Arshia and Moreno for providing me guidance that led me to spend this unforgettable year at IRCAM. I also want to thank my amazing colleagues in ATIAM with whom I have developed wonderful friendships. I will hold memories of coffee machine conversations, late-night dinners, and various excursions around Europe and Paris dear to my heart. Lastly, I am indebted to my family, Liching, Ko-Jen, and Florence, without whose unconditional support and tireless love, nothing would have been possible. Words cannot begin to describe my love and appreciation for them. Thank you for everything.

TABLE OF CONTENTS

List of Tables

List of Figures

CHAPTER 1

INTRODUCTION

The use of gestures in controlling computer-generated sounds has long been a subject of interest to researchers and artists alike. Traditionally, gestural control of audio processing has found applications in interactive music systems such as digital instruments [57], interactive multimedia installations [48], and computer games [14]. In these applications, gestural inputs obtained from different types of motion sensors, cameras, or multi-touch interfaces are used to control and interact with sound processes [45]. More recently, with advances in sensing technologies and sound synthesis techniques as well as maturation of theoretical framework behind gestural description of sounds, ventures into other gesture-based sonic exploration are also investigated in fields such as sonification of information [53] and auditory feedback in rehabilitation [7].

The central research question in these different contexts of gesture-based audio processing is the relationship between gesture data and sonic outcomes, which is embodied in the *mapping* scheme that translates input control parameters to output synthesis parameters. In order to create gestural interfaces that are expressive and fluid in its usability, it is crucial to design meaningful mappings from the extracted gesture features to sound synthesis parameters. "Good" mapping strategies create musically satisfying results and can thus open up possibilities for interactive gesture-based compositions or real-time performances.

Our research extends an interactive gesture-to-sound mapping system shown in Figure 1.1. The system employs a strategy called *Mapping-by-Demonstration* (MbD) [21] to build the training set from the user's input examples during the *training phase*. The mapping between gesture and sound features are trained

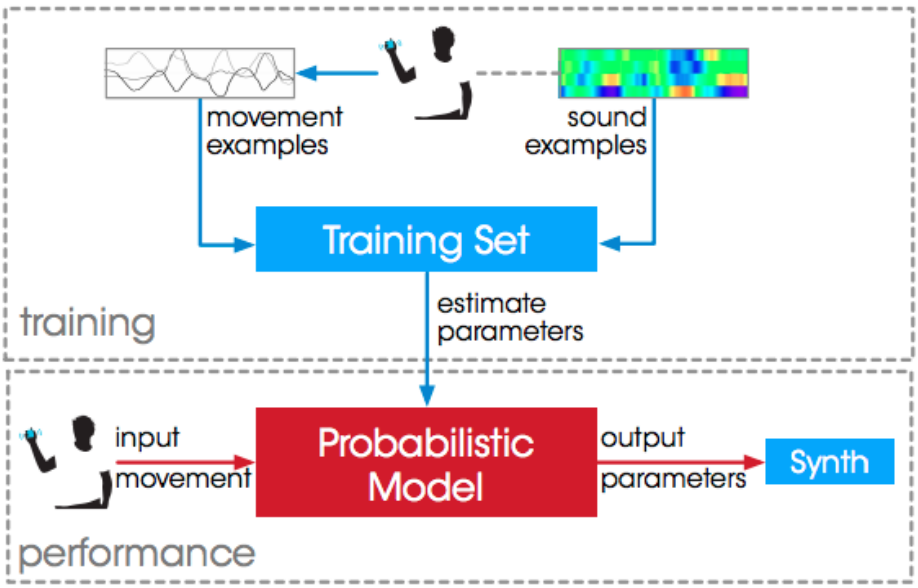


Figure 1.1: Workflow of the Mapping-by-Demonstration system. The *training* phase consists of the user showing the system example gestures performed while listening to a sound sample. The *temporal* and *dynamic* mappings between gesture and sound features are trained jointly in a multimodal model. Then during the *performance* phase, the user performs learned gestures to control the corresponding sound synthesis parameters in real-time. Source: [25]. © Copyright: J. Françoise, 2014

jointly in a multimodal model. Then during the subsequent *performance phase*, the trained models generate sound control parameters from performed gestures. Because of the close gesture-sound coupling, the system can be used to explore different modes of gesture-sound interactions such as vocalization through interactive voice synthesis or mixing recorded sounds using gestural control [26].

In this particular framework, gesture and sound sequences must be segmented and labeled before inputting them into the MbD system. The process disrupts the flow of interaction. The focus of this thesis is to improve the system's fluidity and increase its interactivity by proposing online methods for automatic segmentation of multimodal sequences and clustering of motion-sound segments. These modules will be incorporated into the current framework to enable incremental learning of gesture-to-sound mappings (Figure 1.2). In addition to improving the workflow of the current system, results of this extension serve as a first step toward investigating extended modes of interaction using longer continuous gestures for sound control.

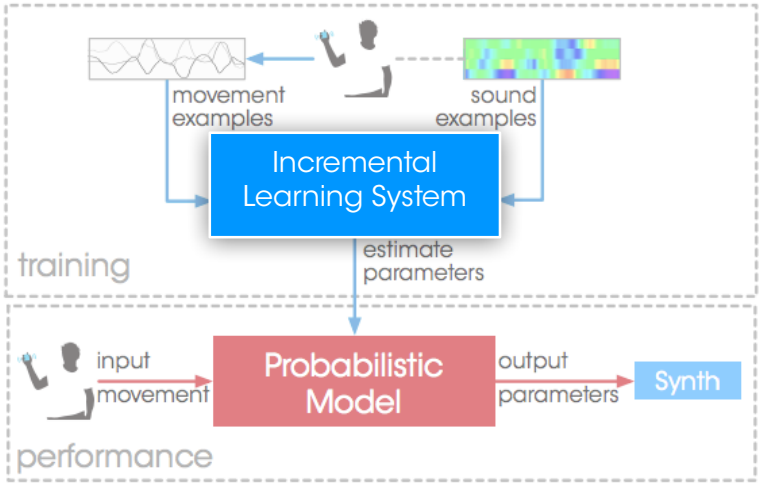


Figure 1.2: How our incremental learning system fits into the original Mapping-by-Demonstration system [21]

1.1 Problem Statement

One of the limitations of the current MbD system is that both the segmentation and learning of the gesture-sound relationships take place offline. This has two major implications:

1. Since the segmentation is offline, gesture sequences and associated sound processes must be segmented and labeled, usually by hand, in order to obtain training examples.

2. Since the learning is offline, users must premeditate all input gestures that will be used during performance in order to avoid having to retrain the models each time a new input is introduced.

However, these implications correspond to the following inconveniences:

1. Manual segmentation is labor-intensive and error-prone, and in certain cases, it may require expert knowledge.

2. Sometimes in realistic deployments, the complete input sequence may not be known beforehand.

In order to address the limitations of the current system and enhance its workflow as well as versatility, this thesis proposes a suite of machine learning techniques to perform online segmentation and learning of motion-sound mappings. Implementation of this extended system involves automatic segmentation of continuous observation sequence, which will improve the ease-of-use of the system by removing the need for annotating segments by hand. In this paper, we define gesture segmentation as the task of partitioning a streaming gesture sequence into distinct sub-gestures that contain semantic meaning. The corresponding streaming audio will also be segmented in like manner. The system will also have an incremental clustering and learning module into which the segmented motion-sound data will be fed incrementally to be labeled, trained, and updated. By removing the necessity for manual segmentation and batch-training, the online system facilitates an uninterrupted training phase, thereby increasing the learning possibilities of input data.

1.2 System Overview

A schematic overview of the incremental learning system is shown in Figure 1.3:

In this extended system, the workflow is as follows: The user performs a set of gestures according to a sound process (whether through listening or through vocalization). The motion capture data and sound data are individually inputted into the system where feature extractions take place. The motion data might also pass through dimension reduction in order to keep only the most significant dimensions. Both gesture and sound features are combined in correlation analysis where correlation between them is learned. The most correlated components are inputs to the automatic segmentation algorithm. Herein, motion-sound primitives are extracted and provided as inputs to the incremental clustering module where new motion-sound pairs are labeled by checking against an incrementally built dictionary of learned pairs based on a similarity criterion. The mapping between gesture and sound in each motion-sound pair is learned through multimodal hidden Markov model at each update.

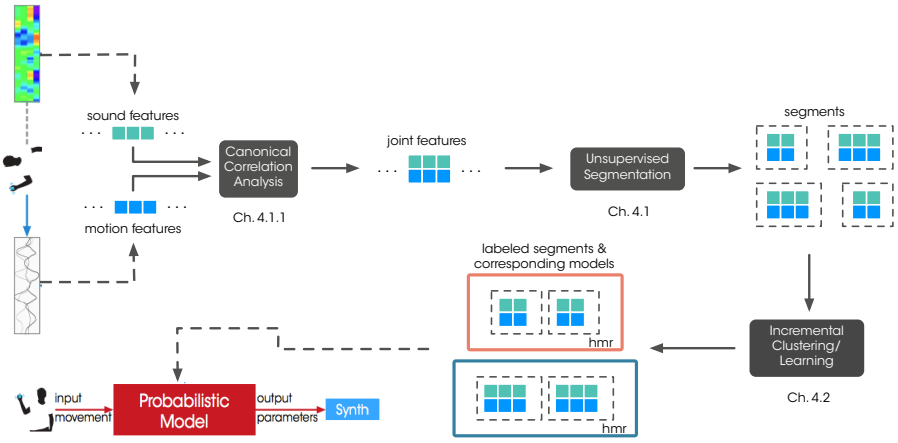


Figure 1.3: System overview: Acquired motion-sound sequences are automatically segmented whose segments are incrementally labeled and learned.

1.3 Contributions and Thesis Outline

Our main contributions are the following:

1. We extended and improved the current workflow of the Mapping-by-Demonstration system by removing the need for offline segmentation and learning.

2. We proposed an incremental learning system that combines online unsupervised segmentation of gesture-sound data streams and incremental clustering and learning of segments in one pass. This creates new avenues for real-time interactive design of gesture-sound mappings.

3. We adapted a time-series automatic segmentation algorithm to work with gesture-sound sequences. To our knowledge, this segmentation algorithm has not been used with multimodal data.

The thesis is organized as follows:

Chapter 2 In order to motivate the subsequent literature review on motion segmentation and incremental learning, this chapter reviews the state-of-the-art in mapping strategies between gesture and sound. It sets up the context for our research. It also provides an overview of related work on interactive machine learning and mapping through listening design principle in order to motivate the mapping-by-demonstration framework.

Chapter 3 Drawing from relevant literature in robotics and computer vision, this chapter gives an overview of the existing work on motion segmentation and incremental learning of motion primitives – the two related tasks our incremental learning system undertakes.

Chapter 4 This chapter presents the formulation of our incremental learning system, detailing the two modules: continuous observation sequence segmentation and incremental clustering and learning of gesture-sound segments.

Chapter 5 This chapter details the evaluation of our incremental learning system. Separate tests are performed for the two tasks and finally these tasks are integrated in a test to evaluate the overall incremental learning system.

Chapter 6 The final chapter concludes the main contributions of the research and provides concluding remarks about the proposed approach for incremental learning of gesture-sound mappings. Directions for future work are also outlined here.

CHAPTER 2

BACKGROUND

This chapter introduces background literature on the gesture-based sound control system that we propose to extend. The system is based on an extension of hidden Markov model (HMM) to model the dynamic gesture and sound relationships. We review literature on gesture-to-sound mappings with particular emphasis on machine learning based methods. Relevant review of research in interactive machine learning and mapping through listening design principle are also given in order to motivate the Mapping-by-Demonstration framework within whose context our proposed extension situates.

2.1 Gesture-to-Sound Mapping Strategies

Investigation of the relationship between gesture data and digital sound processes has received significant attention in the music computing literature. In particular, there exists great interests in interactive music communities such as New Interfaces for Musical Expression (NIME) [1] to explore the notion of *mapping* for sound synthesis controls using gestures to be used in performances or instrument design. Musicians and researchers have been investigating mapping strategies since the late 1990s [54]. These efforts have led to numerous approaches that can be roughly divided into two main classes of strategies: *explicit* parameter wiring, where input gesture data is directly "wired" to output sound synthesis parameters using an analytic expression defined by the designer, and *implicit* models, where an intermediate model lies between the gesture and sound interface

[1] Website: http://www.nime.org/.

to encapsulate complex relationships between the two modalities [32]. Within explicit mapping strategies, finer categorizations give rise to strategies that are based on taxonomy (such as one-to-one, one-to-many, or many-to-one relationships) [56], physical models [47], and geometric properties [55]. In this thesis, we are concerned mainly with the *implicit model* approach where the relationship between gesture and sound is learned implicitly from examples using machine learning techniques rather than defined explicitly using direct wiring.

Neural networks have been used to perform non-linear multi-dimensional mappings [15]. PCA methods have been used to reduce dimensions in the gesture space in order to simplify the mapping procedure [2]. Among the many probabilistic techniques applied to modeling mappings, hidden Markov model (HMM) stands out in its pervasiveness across relevant literature. Variants of HMM have been implemented to perform various types of mappings: explicit mapping through gesture recognition using discrete HMM [37], temporal mapping using a modified version of standard HMM [4], and finally multimodal mapping using hierarchical HMM [23].

2.1.1 Discrete temporal mapping

One of the most common methods for mapping design is through discrete gesture recognition, where recognized gestures are used to trigger musical events. Examples of such methods include sensor gloves [46], which is a gesture recognition system based on neural networks that uses sensor data to continuously control synthesis parameters, and FlexiGesture [44], which uses Dynamic Time Warping (DTW) to learn performers' gestures for recognition later. In addition, discrete HMM's have also been used in [37] to recognize and analyze conducting gestures with the aim of expressive mapping. These techniques have been refined over the years to perform recognition in real-time. However, the types of interaction are quite limited to triggering as interaction mode and explicit mapping as design strategy in this context.

2.1.2 Single-level continuous temporal mapping

In an effort to move from instantaneous triggering of musical events to continuous gestural control, methods based on representations of temporal variations in gestures have been proposed. This paradigm shift is enabled by research efforts by Bevilacqua et al. [4] who developed *Gesture Follower* to continuously recognize and follow gestures in real-time. The system employs a left-right HMM to model the temporal structure of a gesture (as shown in Figure 2.1.

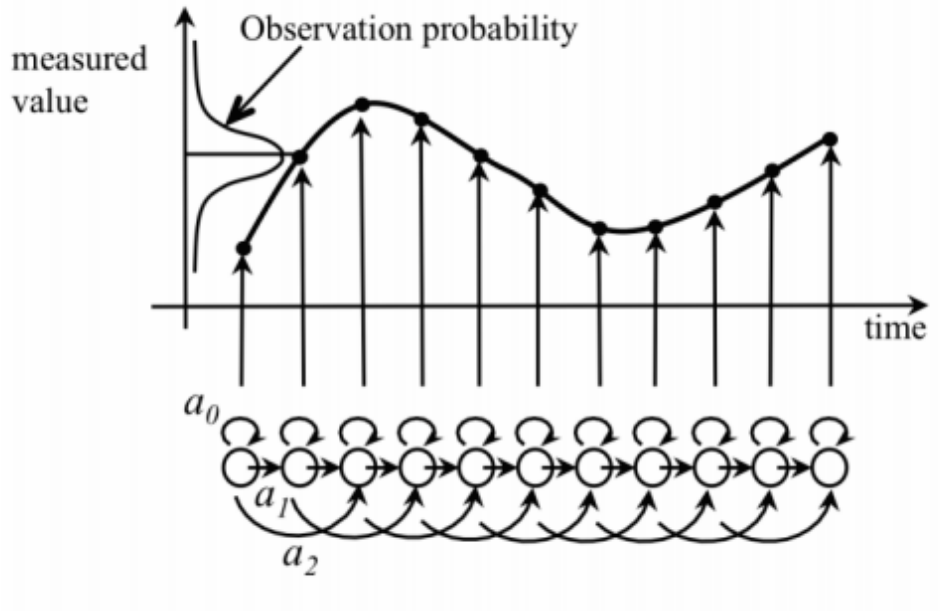


Figure 2.1: Left-to-right HMM structure trained on one gesture template in order to encode temporal information. Source: [3]. © Copyright: Bevilacqua et al., 2011.

It takes as input a single gesture example and treats the time-series data as a template whose frames are associated with states in the HMM. During performance, the system continuously reports the estimated position of the input gesture within a sample template. This method has been applied to gesture-controlled audio processing [3]. In this application, the gesture time progression is directly mapped to the audio time progression, thus allowing audio to be aligned to sound during live performance of the gesture. The system's ability to track the current position of gesture across time allows for temporal control of sound synthesis parameters. For example, a stretched gesture could correspond to a slower audio playback speed. This method allows us to move beyond discrete activation of sounds toward continuous control of the temporal properties of the mapped signal. Figure 2.2 shows a concrete example of the application where the user associates a gesture to the sound sample while listening and plays back the sound by stretching it or compressing it using gesture. Our Mapping-by-Demonstration system is built on this idea of training gesture-sound relationships while listening and subsequently controlling properties of the sound based on the learned model.

In order to address the limitation of Gesture Follower in capturing gesture variations across examples of the same gesture, Caramiaux et al. proposed an adaptive algorithm based on particle filtering called Gesture Variation Follower

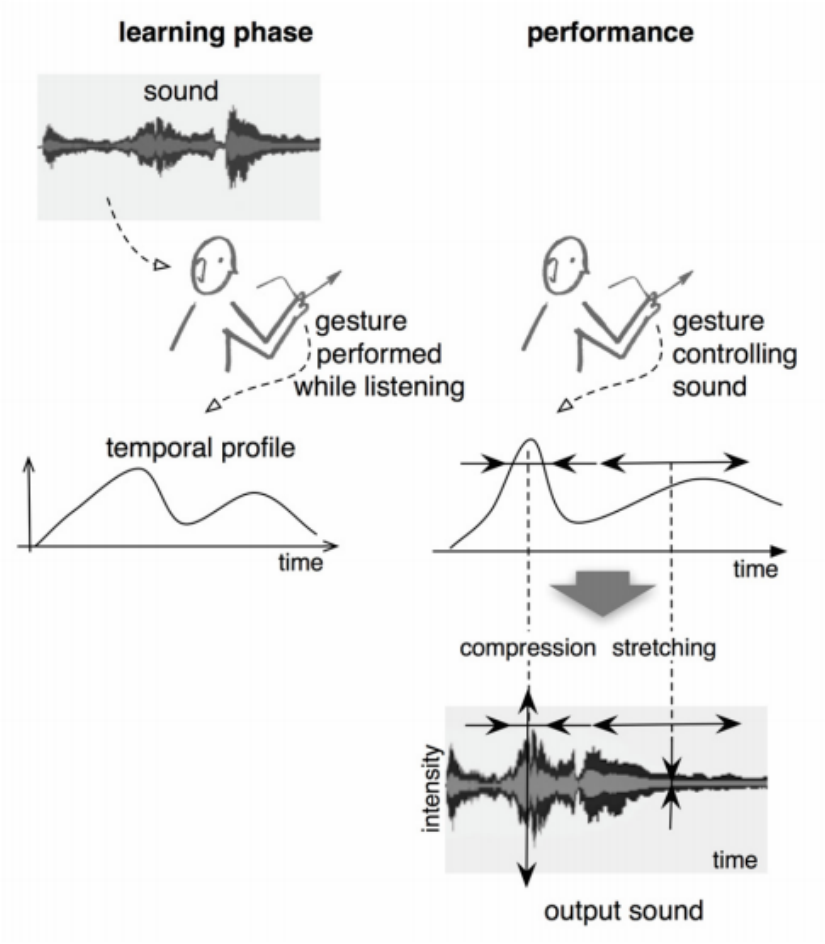


Figure 2.2: Example use-case of temporal HMM for gestural sound control. Source: [3]. © Copyright: Bevilacqua et al., 2011.

(GVF) [11]. More specifically, the method dynamically adapts to gesture variations using a sequential Monte Carlo inference technique. Similar to the Gesture Follower, it performs temporal mapping of the gesture. But in addition to tracking the time progression of the gesture, GVF also tracks changes in characteristics of the gesture that capture its variations, such as position, speed, and rotation. This allows for gesture recognition that is adaptive to variations during a gesture performance without requiring the users to provide additional examples of different variations as training set. In other words, GVF is able to train an adaptable model based on single templates of the gestures to be performed.

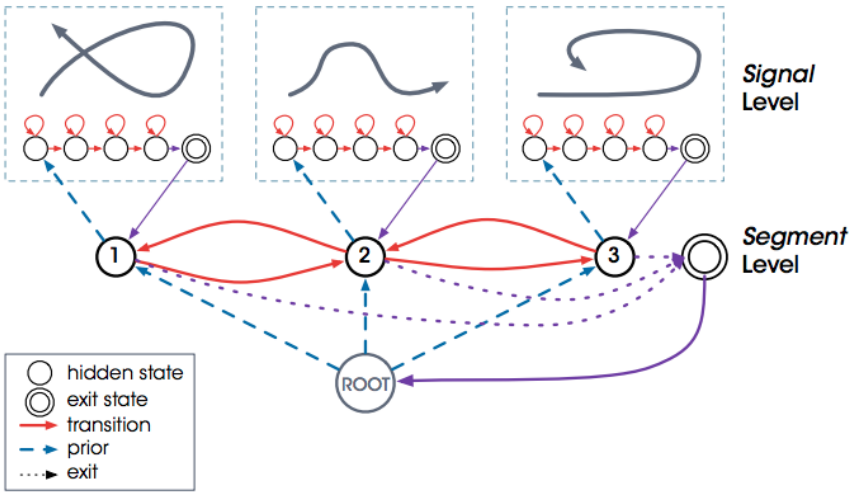


Figure 2.3: Hierarchical structure of a short gesture sample prepared for the training phase. Each segment in the sequence is associated with a state, S_i. And a submodel for temporal within-gesture tracking can be extracted from each of these segments. Source: [21]. © Copyright: J. Françoise, 2015

2.1.3 Multi-level continuous temporal mapping

In Gesture Follower, gestures are represented using a single-level time structure. This implies that each gesture is treated as one unbreakable unit of time-series sequence. However, different findings [28] [40] suggest that associated gesture and sound sequences are broken up into smaller units of information during music perception, called "chunks". A holistic musical idea is formed, hence, by "fusing" and "transforming" these segment-level musical units into larger units at varying structural levels (i.e. timescales) [9]. In order to address the multi-level structure inherent in gesture and sound that is insufficiently described by the original HMM setup in Gesture Follower, hierarchical HMM is introduced to enable control over gestures at segment-level [23]. In this hierarchical approach, each gesture is represented as a 2-level time structure where the micro-level *signal states* capture the same fine-grained temporal information as the original Gesture Follower for tracking time progression of the gesture frame-by-frame; then the macro-level *segment states* encode transitions between different segments. Figure 2.3 illustrates this structure in a 3-segment sample.

For an illustrative example of how inter-gesture transitions can be realized given more than one gesture, refer to Figure 2.4.

Note that segments in this context refer to the low-level segments within a given *gesture instance*, whereas segments this thesis is interested in refer to

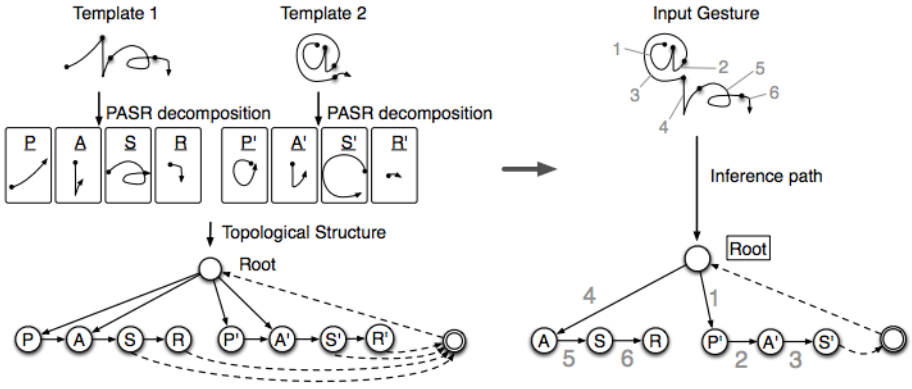


Figure 2.4: Inter-gesture transitions using an example structure configuration. The top left image shows how the two gesture templates are segmented and learned. The bottom left image shows an example hierarchical structure of the model. The different segments can be sequenced as shown in the top right image during performance, whose inference path is shown in the bottom right image. Source: [23]. © Copyright: Françoise et al., 2012

higher level segments within a given *gesture sequence*. While the segmentation algorithm proposed by this thesis could be used to examine segmentation at a finer *in-gesture* level, we believe control of segmentation at that level should remain with the users based on their creative judgments. Our work more importantly examines segmentation at the *inter-gesture* level where boundaries can be found at transitions to new gestures.

2.1.4 Regression mapping

The temporal mapping strategies reviewed thus far use discrete and continuous gesture recognition as the primary method to drive sound synthesis. In other words, gesture models are trained independently from the sound process where the resulting gesture models are used to activate sound parameters using an explicit formulation defined by the user. The lack of joint model between gesture and sound implies that there is no direct correlation between gestural information and acoustic information [11]: sound synthesis parameters cannot be generated from motion parameters directly; they must pass through an analytical formulation layer (such as triggering or alignment mentioned in previous sections) in order to arrive at the mapped sound parameters. A more integrated approach to mapping exploits regression methods to directly learn the relationship between motion and sound features. Most early approaches rely on neural networks to learn the non-linear

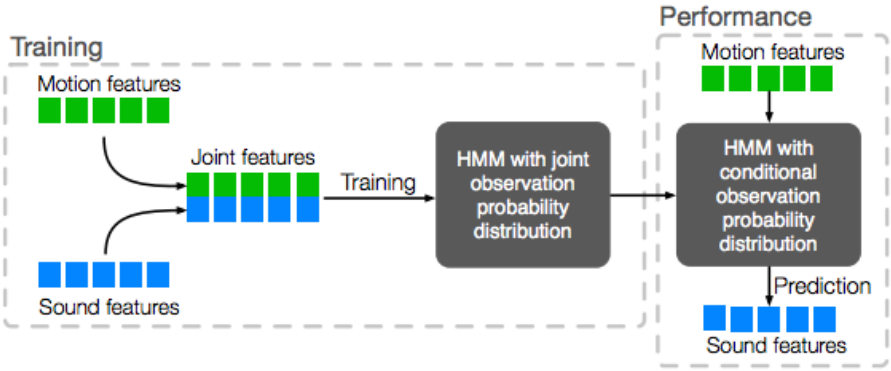


Figure 2.5: Gesture-to-sound mapping using multimodal HMM. Source: [24]. © Copyright: Françoise et al., 2014.

relationships between two modalities, allowing for parameter estimation from one modality (e.g. gesture) to another (e.g. sound) [41]. However, in these approaches, while the non-linear relationship between gesture and sound is modeled through supervised learning of example "true" input/output pairs [17], this relationship is often static and lacks the temporal dimension offered by the temporal mapping approach. Françoise et al. [24] developed a method for multimodal learning of gesture-sound relationships using hidden Markov regression (also termed multimodal HMM) that retains temporal relationship. The general idea behind this mapping strategy is to learn an HMM on joint gesture and sound data, from which we can extract a conditional model to estimate associated sound features given a new gesture during the performance phase. Figure 2.5 provides a closer look into this process.

2.2 Mapping-by-Demonstration (MbD) Framework

Armed with the probabilistic tools detailed in the previous section to model gesture-sound relationships, we now need to formalize the conceptual framework for mapping design.

Recent research [19, 27] has shown growing interest in an “interaction-driven” approach to the learning of mappings. This framework allows users to define mappings interactively by providing training examples of gesture-sound relationships. The current mapping system proposed by Françoise et al. [21] operates within a similar interactive learning framework where particular emphasis is placed on designing through listening.

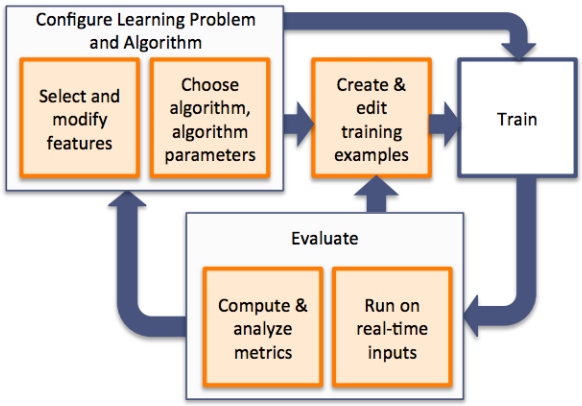


Figure 2.6: Interactive machine learning workflow implemented in the Wekinator. Source: [19]. © Copyright: R. Fiebrink, 2011.

2.2.1 Interactive machine learning

Interactive machine learning (IML) is a computational design methodology for integrating user manipulation in complex machine learning techniques. It increases accessibility of complex concepts which were previously only accessible to experts. It also aims to improve performance of machine learning algorithms by incorporating end-user interaction and feedback.

IML has been used in the sound computing community to rapidly prototype gesture-controlled instruments without programming them through machine commands [18]. Its property of allowing the users to bypass programming and directly interact with underlying machine learning algorithms enable more natural interpretation of the parameters used by the models and yield more customized results, which are particularly suitable for designing gesture-sound mappings. Applications of IML include gestural interfaces that allow skilled musicians to create experimental music, such as Wekinator [16].

The general workflow of an interactive learning system in the context of gesture-sound design is as follows (Wekinator example in Figure 2.6: 1. Training: The user builds the training set by demonstrating to the system examples of gesture and sound pairings. The users are able to manually tune the model parameters for training. 2. Performance: The user interacts with the trained model in real-time by performing gestures from the training phase and evaluates the synthesized sonic feedback. 3. The user can go back to step 1 to retrain the model by providing additional examples or by adjusting the model parameters. The process iterates until the he/she is satisfied with the performance results.

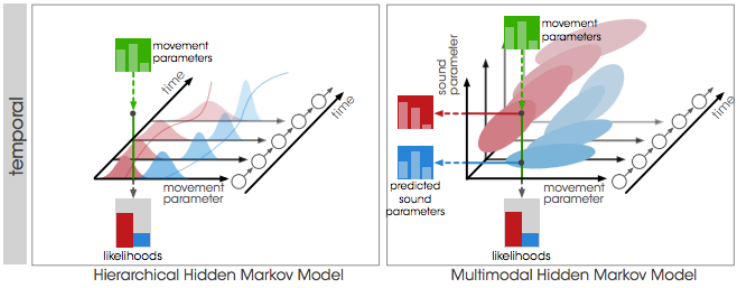


Figure 2.7: Summary of the temporal probabilistic models used with in the MbD system. Source: [21]. © Copyright: J. Françoise, 2015.

2.2.2 Programming-by-Demonstration

Both focused on end-user development, interactive machine learning serves as a framework for improving the design of machine learning systems, whereas Programming-by-Demonstration (PbD) is a framework for teaching systems new concepts or behaviors through demonstrations of example tasks, all without use of code. It has been applied to interactive computer music in work by Merill et al. [44], where an electronic instrument is built by learning which gestures trigger which sound samples through demonstrations from the user.

2.2.3 A unifying framework: Mapping-by-Demonstration

The set of machine learning techniques and design methodologies for interactive systems reviewed in this chapter have yielded new mapping and design possibilities. Mapping-by-Demonstration [21] is a general framework that unifies these different research efforts in tackling gesture-sound mapping. Specifically, it implements 4 probabilistic models to address different mapping scenarios resulting from variations in *temporality* and *multimodality* (See Figure 2.7 for summary of the models' characteristics). These models follow a Mapping-by-Demonstration methodology in designing interactions. This thesis is situated in this MbD framework and we aim to evaluate our extension specifically in the context of 2 temporal models with varying degree of modalities: hierarchical HMM and multimodal HMM (refer to Figure 2.7).

CHAPTER 3

RELATED WORK

All the techniques for gesture-sound mapping reviewed in the previous chapter consider the case where gestures and its associated sound parameters to be learned are pre-segmented a priori and clustered off-line by the designer. The training phase for learning the mappings between them is therefore a one-shot, offline process. In order to devise an online training algorithm where the observation sequence becomes available sequentially rather than in a batch so as to support more continuous interaction, strategies to autonomously segment and cluster the motion-sound data must be studied. We consider segmentation algorithms that automatically decompose a continuous stream of time-series data into distinct gestures. The algorithm is also expected to jointly segment the corresponding sound process data so that the relationship between motion and sound can be learned. In addition, since the model used for mapping gestures to sounds requires labeled segments for training, a clustering strategy is needed to incrementally label the algorithm-generated segments on-the-fly. Finally, to fully integrate our extension into the existing Mapping-by-Demonstration system, we build the motion-sound models incrementally. We briefly detail here further considerations for the segmentation and clustering modules in our extended system, thereby motivating our literature review.

Segmentation There has been active treatment of motion segmentation in the vision, graphics, and robotics literature. We focus on approaches that do not require a pre-annotated set of training samples. Furthermore, we are mainly interested in extracting high-level rather than low-level segments, i.e. segments that represent complete gestures instead of their lower level gesture components. Finally, we

require that the segmentation takes place online in order to meet our constraint that data are to be received sequentially.

Clustering As for the clustering technique used in labeling the segments, we focus on incremental approaches where clusters are built in the order data arrives. We consider techniques that also simultaneously learn the motion-sound relationships during the incremental clustering step.

This chapter reviews existing literature on automatic unsupervised human motion segmentation which will inform segmentation of multimodal motion-sound data. It then provides an overview of common approaches to incremental clustering and learning of motion primitives in order to contextualize our approach.

3.1 Segmentation of Human Motion

We define segmentation of gesture sequences commonly encountered in our Mapping-by-Demonstration system [21] as the process of extracting distinct behaviors or semantically meaningful segments from continuous multi-dimensional time-series data. Human motions can be measured via either motion capture systems or ambulatory sensors such as inertial measurement units (IMUs). In this thesis, we focus on movement data from IMUs. Figure 3.1 illustrates the problem of temporal segmentation: given sensor data of a person performing a gesture sequence, we want to isolate distinct gestures by defining the boundaries between each gesture.

Gesture segmentation can be classified into two categories: textitsupervised algorithms that have access to known gesture templates in order to perform the segmentation and *unsupervised* algorithms that require no a priori knowledge of the gestures being observed. Given the constraints of our system, we consider only unsupervised techniques. Work in supervised gesture segmentation in musical contexts can be found in [22] and [12]. However, those methods assume prior knowledge on a set of motion primitives and therefore is outside the scope of this paper.

To our knowledge, research in unsupervised gesture segmentation in the context of sound computing or NIME communities is still quite nascent. Most of these methods take a simplistic approach such as using acceleration change as the threshold to segment a motion sequence [2]. In this example, the method interprets gesture transitions as points where the movement has stopped, started, or significantly changed directions. However, this approach performs segmentation at the signal level rather than in a feature space that captures the temporal

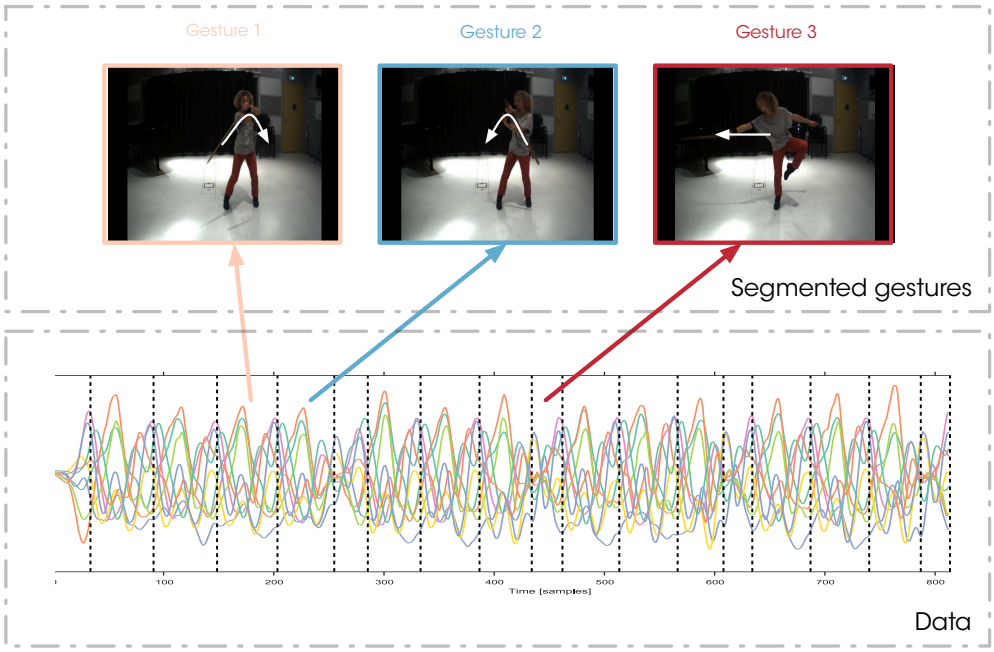


Figure 3.1: Example gesture segmentation of a Tai-Chi movement sequence from sensor data.

and spatial properties of the signal, making it limited in discovering higher-level gestural information.

We therefore turn to other communities such as robotics and vision for literature treatment of unsupervised movement segmentation. These techniques can be further categorized by whether or not motion templates are used in the segmentation. We consider two approaches: template-free and probabilistic template based approaches.

3.1.1 Template-free approaches

Motion templates are pre-determined prototypes of motion against which observed data is checked in order to determine segment candidates. In unsupervised techniques, motions to be identified are not known a priori, eliminating the need for templates. In this section, we examine methods that do not require templates for motion segmentation. The algorithms can be categorized by the assumptions they make about the underlying structure of the data at segmentation points.

3.1.1.1 Segmentation based on velocity properties

Several algorithms have been developed to use velocity properties as the basis for segmenting motions. Pomplun et al. [49] assumes that there exists a pause

between motion transitions. They propose a method that declares a segmentation point when the root mean square (RMS) of the joint velocities fall below a certain threshold. While this technique provides an efficient way of performing segmentation, it is restrictive in that fluidity in natural human movement is neither assumed nor represented.

Fod et al. [20], on the other hand, assumes that a change of movement direction, measured by angular velocity, corresponds to possible transition to a different motion. They propose a method based on zero-velocity crossings (ZVCs) to detect those points. ZVCs are points whose angular velocity values have changed signs (from positive to negative or from negative to positive). Therefore, a segmentation point is recognized when a sufficient number of dimensions in the joint angle data exhibit ZVCs within a short time frame.

Lieberman et al. [42] builds on the ZVC-based method by taking into account other types of signals (such as tactile information and stereo vision) collected in a imitation learning setting in humanoid robots. They also define an adaptive velocity threshold for determining the significance of a movement in order to inform the segmentation.

Techniques based on velocity properties are fast but tend to produce over-segmentation because velocity changes that are unrelated to action change can occur. Noise in signal can also contribute to spurious segmentation. Although a post-processing step can be introduced to merge the over-segmented motions, it lacks the intelligence needed to inform which segments to merge or not to merge. Furthermore, ZVC-based methods are subject to inaccuracy as dimensions increase in observation data. For example, when multiple dimensions display ZVCs with slight offset from one another, it is unclear where to place the segmentation point. Finally, these methods are restricted to movement sequences that are well characterized by ZVCs. In sequences that contain smooth transitions between movement segments, velocities may not actually cross zero even though a segment should have been detected.

3.1.1.2 Segmentation based on variance in feature data

Koenig et al. [35] propose a segmentation algorithm based on signal variance. A sliding window is passed over the movement sequence, computing the variance of each window based on a cost function. Segmentation cuts are produced at points with maximum variance. This is based on the assumption that movements that are in transition to a new action display high signal variance. While this method is intuitive, it may also over-segment when the performed actions contain inherently large variance.

However, signal variance may not capture sufficient information about the observed data. Kohlmorgen and Lemm [36] uses probability density function (pdf) to represent the observation sequence in a sliding window. The pdfs are used to train a hidden Markov model (HMM). the Viterbi algorithm is used to generate the most likely state sequence representing segmentation points from the pdfs.

In graphics literature, Barbic et al. [1] proposes the use of simple methods such as Principal Component Analysis (PCA), probabilistic PCA (PPCA), and Gaussian Mixture Model (GMM) to perform segmentation. The PCA and PPCA methods detect sudden changes in the intrinsic dimensionality of the motion sequence, whereas GMM is used to detect changes in the distribution. These changes mark transitions to new segments. These approaches have been shown to provide good performance in motion segmentation problems.

3.1.2 Probabilistic template based approaches

Another approach to segmentation is to formulate the segmentation problem as identification of motion templates in current observation data. Motion templates that characterize basic actions in streaming motion data can be modeled in an unsupervised manner using probabilistic methods.

The algorithm proposed by Chiappa et al. [13] treats the observed motion sequence as a concatenation of hidden trajectories, namely motion templates, that are transformed according by time warping and additive noise. Bayesian likelihood is used to compute the probability that the observation sequence is derived from some hidden trajectory. Expectation-maximization (EM) algorithm is used to estimate the warping needed to transform the observed data to the hidden trajectory. Segmentation is performed through extracting motion primitives. However, this methods requires computationally intensive batch-processing, making it unsuitable for online applications.

3.1.3 Summary

Many techniques exist across disciplines to tackle the segmentation problem. ZVC-based methods, albeit fast and lightweight, are prone to over-segmentation. Sophisticated generative probabilistic models such as Bayesian produce good segmentation results but involve heavy computations and are offline.

We adopt an online, template-free approach to segmentation based on hidden Markov models (HMM). Our approach is adapted from the Kohlmorgen and Lemm [36] algorithm for general time-series segmentation to handle multimodal data sequences. The algorithm produces segmentation in an unsupervised manner without annotated gesture database.

After the observation sequences are segmented into primitives, the next step is to build a system that incrementally learns models of gesture-sound relationships as new data arrive. The system should also be able to update its existing models or incorporate any newly-built models into its knowledge base.

Most systems for motion primitive learning take place offline or require the motion segments to be labeled a priori. For instance, Jenkins and Mataric [33] propose a system for clustering the segmented data into groupings. They employ the spatiotemporal Isomap (ST-Isomap) algorithm to embed the data in lower dimensional space, and using the "sweep-and-prune" technique, they cluster these segments into groupings, thereby constructing the primitive motion model. While this system automatically clusters segmented data, it cannot be turned into an incremental algorithm as the the entire data sequence must be available for subspace embedding. Other systems, such as those proposed by Billard et al. [5], uses manually clustered (i.e. labeled) data to train their HMM-based motion primitive model.

We are interested in building a system that incrementally updates its models of gesture-sound relationships as new data arrive and incorporates any newly-built models into its knowledge base. In subsequent section, we draw from robotics literature in learning-by-demonstration for a review of incremental learning techniques.

3.2 Incremental Learning of Gestures from Demonstration

3.2.1 Online learning techniques

Calinon et al. [8] describe an approach for incremental learning of motions based on Gaussian mixture models (GMM). In this system, motion data are passed through principal components analysis (PCA) to determine a relevant subspace. Next, the reduced dataset is abstracted into a set of Gaussian mixtures, and the structure of the GMM is learned through incremental training.

Kadone and Nakamura [34] develop a system based on associative neural networks with non-monotonic sigmoid functions to perform online learning of human motion primitives. The learned primitives are automatically clustered into a hierarchical tree structure. However, the built models are used to recognize motions rather than to generate motions.

Kulic et al. [39] have presented an incremental system for learning motion pattern primitives. In this system, motion primitives are represented by HMM that can be used for motion recognition and generation. New motion primitives are

incrementally clustered based on their relative distance to existing motion models. If the distance is high, indicating that the newly observed data is dissimilar to existing models, an HMM is created to represent the new data. If the distance is low, the observed data is merged with the existing model to which it is most similar. A hierarchical tree structure is formed as a result of clustering.

3.2.2 Summary

In order for a technique to be suitable for integration into our target gesture-sound mapping system, it requires the following properties:

1. The system's embedded mechanisms to recognize previously learned motions as well generation of exemplary motion prototypes.

2. The system's ability to automatically cluster and learn each newly introduced motion

3. The system's organization of learned motion models for easy retrieval later.

We employ an approach similar to Kulic et al. [38] for incremental learning of gesture-sound segments. As the observed data are segmented, they are abstracted into HMMs, where the algorithm incrementally clusters the segments into temporally coherent groups of actions and stores them into a storage system.

CHAPTER 4

PROPOSED APPROACH

Our goal in this thesis is to address the problem of extracting gesture-sound primitives during online observation. Two sub-problems are involved in addressing this problem: detection of boundaries between distinct musical actions, and grouping of segment models based on similarity measure and subsequent storage of these models. Our algorithm takes a combined segmentation and incremental clustering approach in learning the mappings between gesture and sound. We have set the following constraints for our proposed system:

1. Knowledge constraint: We do not presume a priori knowledge about the data to be received by the system. This indicates an unsupervised approach to the segmentation of motion-sound sequences and the subsequent learning of multimodal mappings. Rather than having a pre-existing database of gestures to learn from, we build the gesture database from examples in real-time.

2. Workflow constraint: We require that the system receives gesture-sound data sequentially over time rather than through a batch processing method. This suggests that we must perform online segmentation of the data inputs as well as incremental learning of the segments.

3. Time constraint: We wish to bring the entire training pipeline online to run in real-time, from segmentation to labeling and to the eventual model training. As such, our segmentation and clustering algorithms must be computationally efficient.

In order to meet these constraints, we propose a probabilistic segmentation algorithm based on work by Kohlmorgen et al. [36] on segmentation of time-series data. We adapted the algorithm to work with multimodal gesture-sound data. We then employ an incremental clustering algorithm for the labeling and learning of the segmented gestures. The incremental learning framework takes inspiration from work by Kulic et al. [39] on the learning of full-body movements in a human-robot interaction setting. Our approach differs in that we do not implement a hierarchical storage structure but rather a linear one.

In the following sections, we first detail the unsupervised online segmentation algorithm with a presentation of the multimodal extension. We subsequently detail the incremental clustering technique that combines the learning of the HMMs.

4.1 Unsupervised Segmentation of Multimodal Sequences

We apply and adapt the segmentation algorithm proposed by Kohlmorgen et al. [36] for unsupervised segmentation of gesture-sound sequences. The algorithm segments multivariate data probabilistically by defining an HMM over the observed data sequence, associating each hidden state with a window in the observation sequence. The Viterbi algorithm is then used to find the optimal *state* sequence that best represents the observed sequence, where optimality is defined based on the distance between neighboring data windows.

A general overview of the algorithm is as follows:

1. Feature extraction. For this component, features are extracted from motion and sound data. We first find the most correlated motion and sound components using Canonical Correlation Analysis (CCA) and then we map the joint data to a probability density function (pdf). The use of pdf provides convenient mathematical tools for tracking changes in the distribution and for capturing more complex distributions than raw signal data could.

2. Similarity measure. A distance metric is defined for computing the divergence between two distributions. In this case, we use the standard Euclidean distance function.

3. Online segmentation algorithm. The online segmentation algorithm utilizes inter-distribution distances to discover prototypes within an input sequence.

4.1.1 Feature extraction

Given rich input sensor and audio data, it is typically helpful to select a set of features that facilitate discrimination between classes. This also serves to remove noisy signals that will confuse the algorithm. While it is possible to manually select optimal features given a priori knowledge about application-specific signals, it is usually more realistic and less labor-intensive to automatically find the right set of features as inputs to the algorithm to help it achieve high accuracy. In this step, we first find correlations between the two types of signals we are dealing with – gesture sensor data and audio sample. We then map the correlated streaming gesture-sound data into some feature space to facilitate the algorithm's computation of the similarity matrix.

4.1.1.1 Gesture representation

Gesture sensor data is represented as a multi-dimensional vector, with each dimension corresponding to a particular observation on a gesture feature. The features we obtain are extracted from motion capture data (specifically inertial measurement units) – the velocity coordinates are v_x, v_y, v_z and the acceleration coordinates are a_x, a_y, a_z, where x, y, z correspond to the 3 axes reported by the gyroscope and the accelerometer. These features capture the geometric and dynamic information in movements.

4.1.1.2 Audio representation

To represent the audio signal, we extract Mel-Frequency Cepstral Coefficients (MFCCs) [50] as features, which serve to capture the short-term spectral-based features of the signal.

4.1.1.3 Canonical Correlation Analysis (CCA)

Since our input stream is multimodal (gesture and sound), gesture data alone does not describe the complete gesture-sound properties we wish to capture. Segmentation on gesture-only data can lose out on the rich audio information that is recorded synchronously with the listener's gestures as there is a two which the listener performs a gesture that describes the sound. In order to address the possible interdependency between gesture and sound data, we propose the use of CCA to perform analysis on both modalities [10]. CCA is used to find correlation between multimodal gesture and sound data based on the assumption that motion and sound samples are closely correlated with each other. In our target applica-

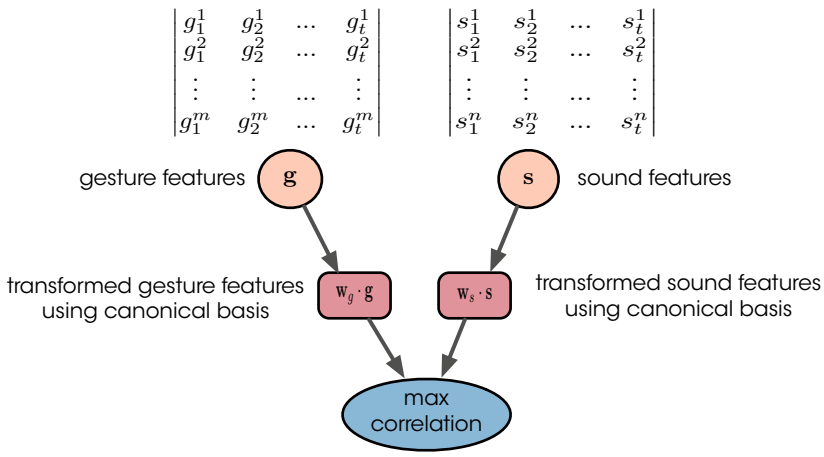


Figure 4.1: Given an m-dimensional vector of gesture features and n-dimensional vector of sound features, CCA finds their linear relationships using canonical bases $\mathbf{w}_g$ and $\mathbf{w}_s$

tions, such as vocalization of gestures, changes in sound correspond to changes in gestures.

Given gesture data received from the motion sensors and sound features extracted from the sound processes, we aim to identify the gesture components that are most correlated with the audio components. Let $\mathbf{s}$ be sound features, our problem can be formulated as finding a dimension in gesture feature space $\mathbf{g}$ that contributes the most to the maximization of correlation with $\mathbf{s}$. A simple correlation is not suitable for our problem because it is highly sensitive to the coordinate systems that describe $\mathbf{s}$ and $\mathbf{g}$, and in our case, gesture features $\mathbf{g}$ and sound features $\mathbf{s}$ share entirely different coordinate systems. Furthermore, a simple correlation does not estimate the contribution toward the correlation result of components along the gesture feature space $\mathbf{g}$ and sound feature space $\mathbf{s}$. Therefore, we need a method that will project both the sound and gesture features onto a common coordinate system and also simultaneously estimates their corresponding correlations.

Canonical Correlation Analysis (CCA) proposed by Hotelling [31] provides such a method by finding the linear transformation of the first variable that is most correlated to the some linear transformation of the second variable such that the correlation between two multi-dimensional random variables can be determined. Figure 4.1 shows the transformation of the random variables $\mathbf{g}$ and $\mathbf{s}$ to a common source.

CCA is used to find the canonical bases, $\mathbf{w}_s$ and $\mathbf{w}_g$, that maximize the correlation between the projections $\mathbf{g}' = \mathbf{w}_g^\top \mathbf{g}$ and $\mathbf{s}' = \mathbf{w}_s^\top \mathbf{s}$. The canonical correlation

between **s** and **g** is formulated in terms of the covariance matrices for **g** and **a**: $C_v v \in^{m \times m}$ and $C_s s \in^{n \times n}$, as well as cross-covaraince matrix of the vectors **g** and **s**: $C_g s \in^{m \times n}$. The covariance matrices are estimated by the total covariance matrix $C(\mathbf{G}, \mathbf{S})$ as follows:

$$C(\mathbf{G},\mathbf{S}) = \begin{bmatrix} C_{gg} & C_{gs} \\ C_{sg} & C_{ss} \end{bmatrix} = \mathbb{E}\left[\binom{\mathbf{g}}{\mathbf{s}} \quad \binom{\mathbf{g}}{\mathbf{s}}^{\top} \right] \tag{4.1}$$

As such, we formally define the canonical correlation, ρ, as,

$$\begin{aligned} \rho &= \max_{\mathbf{w}_g, \mathbf{w}_s} \frac{\mathbb{E}[\mathbf{g}'\mathbf{s}']}{\sqrt{\mathbb{E}[\mathbf{g}'\mathbf{g}'^{\top}]\mathbb{E}\mathbf{s}'\mathbf{s}'^{\top}}}, \\ &= \max_{\mathbf{w}_g, \mathbf{w}_s} \frac{\mathbb{E}[\mathbf{w}_g^{\top}\mathbf{g}\mathbf{s}^{\top}\mathbf{w}_s]}{\sqrt{\mathbb{E}[\mathbf{w}_g^{\top}\mathbf{g}\mathbf{g}^{\top}\mathbf{w}_g]\mathbb{E}[\mathbf{w}_s^{\top}\mathbf{s}\mathbf{s}^{\top}\mathbf{w}_s]}}, \\ &= \max_{\mathbf{w}_g, \mathbf{w}_s} \frac{\mathbf{w}_g^{\top}\mathbb{E}[\mathbf{g}\mathbf{s}^{\top}]\mathbf{w}_s}{\sqrt{\mathbf{w}_g^{\top}\mathbb{E}[\mathbf{g}\mathbf{g}^{\top}]\mathbf{w}_g\mathbf{w}_s^{\top}\mathbb{E}[\mathbf{s}\mathbf{s}^{\top}]\mathbf{w}_s}}, \\ &= \max_{\mathbf{w}_g, \mathbf{w}_s} \frac{\mathbf{w}_v^{\top}C_{gs}\mathbf{w}_s}{\sqrt{\mathbf{w}_g^{\top}C_{gg}\mathbf{w}_g\mathbf{w}_s^{\top}C_{ss}\mathbf{w}_s}} \end{aligned} \tag{4.2}$$

In the above equations, $\mathbb{E}[.]$ denotes empirical expectation. This equation has a closed-form solution using Lagrange multipliers, which results in an eigenproblem as,

$$\begin{aligned} C_{gg}^{-1}C_{gs}C_{ss}^{-1}C_{sg}\mathbf{w}_v &= \lambda^2\mathbf{w}_g \\ C_{ss}^{-1}CagCgg^{-1}C_{gs}\mathbf{s} &= \lambda^2\mathbf{w}_s \end{aligned} \tag{4.3}$$

Here, $\mathbf{w}_g$ and $\mathbf{w}_s$ are canonical bases of **g** and . The eigenvectors w_{g^1} and w_{s^1} correspond to the largest eigenvalue λ^2 and they maximize the correlation between canonical variates, $v_1' = w_{g_1}^{\top}\mathbf{g}$ and $s_1' = w_{s^1}^{\top}\mathbf{s}$. More details about CCA can be found in [30].

4.1.1.4 Projecting gesture and sound data to feature space

In order to uncover underlying structures, the incoming data stream is first embedded into a higher dimensional space. Consider an incoming data stream, $\vec{y}_1, \vec{y}_2, \vec{y}_3, ...$ with $\vec{y}_t \in^n$, we embed the data into higher-dimensional space,

$$\vec{x}_t = (\vec{y}_t, \vec{y}_{t-\tau}, ..., \vec{y}_{t-(m-1)\tau}) \tag{4.4}$$

where parameter m is the embedding dimension and τ is the delay parameter of the embedding. The dimension of the vector $\vec{x}$ is hence $d = mn$.

Next, the density distribution of the embedded data is estimated over a sliding window of length W using a standard density estimator with multivariate Gaussina kernels [6], which is centered on the data points in the window $\{\vec{x}_{t-w}\}_{w=0}^{W-1}$, and is given by

$$p_t(\mathbf{x}) = \frac{1}{W}\sum_{w=1}^{W-1}\frac{1}{(2\pi\sigma^2)^{d/2}}\exp\Big(-\frac{(\mathbf{x}-\vec{x}_{t-w})^2}{2\sigma^2}\Big). \tag{4.5}$$

where $p_t(x)$ is the density distribution of the window t, and σ the variance and can be estimated from distribution by choosing σ to be proportional to the mean distance of each $\vec{x}_t$ to its first d nearest neighbors, averaged over $\{\vec{x}_t\}$. σ acts as a smoothing parameter, also known as the bandwidth of the kernel. It controls the degree of smoothing. Large σ produces smooth curves but also does not pick up the details, whereas small σ risks picking up too much detail, resulting in a noisy fit. W denotes one observation unit and controls the size of variation we wish to detect in the distribution. Small W (< 5) allows for detection of minute variations in the distribution while large W (> 20) allow the algorithm to overlook these variations and detect larger changes in the distribution. Usually W should be large enough to capture the full density distribution of a single gesture, but small enough so as to avoid expensive computations.

4.1.2 Similarity measure

After enough sample points are collected to form the first pdf, a new pdf can be formed with each new subsequent point. The distance between two pdf's,

$$d(p_t, p_s) = \int (p_t(x) - p_s(x))^2 dx \tag{4.6}$$

can be calculated using integrated square error (ISE). We first consider the case of two general mixtures $f = \sum_{i=1}^{F}\alpha_i f_i$ and $g = \sum_{j=1}^{G}\beta_i g_j$.

$$\begin{aligned} d(f,g) &= \int \big(f-g\big)^2 d\mathbf{x} \\ &= \int (\sum_{i=1}^{F}\alpha_i f_i - \sum_{j=1}^{G}\beta_j g_j)^2 d\mathbf{x} \\ &= \int (\sum_{i=1}^{F}\alpha_i f_i)^2 - 2(\sum_{i=1}^{F}\alpha_i f_i)^2(\sum_{j=1}^{G}\beta_j g_j) + (\sum_{j=1}^{G}\beta_j g_j)^2 d\mathbf{x} \\ &= \sum_{i,k}^{F}\alpha_i\alpha_k\int f_i f_k d\mathbf{x} - 2\sum_{i=1}^{F}\sum_{j=1}^{G}\alpha_i\beta_j\int f_i g_j d\mathbf{x} + \sum_{j,l}^{G}\beta_j\beta_l\int g_i g_l d\mathbf{x} \end{aligned} \tag{4.7}$$

The integral of the two multivariate Gaussian distributions $f_i \sim \mathcal{N}(\vec{\mu}_i, \sigma_i^2 I_d)$ and $f_j \sim \mathcal{N}(\vec{\mu}_j, \sigma_j^2 I_d)$ can be found using,

$$\int f_i f_j d\mathbf{x} = \frac{1}{(2\pi(\sigma_i^2+\sigma_j^2))^{\frac{d}{2}}} \exp\left(\frac{-(\vec{\mu}_i-\vec{\mu}_j)^2}{2(\sigma_i^2+\sigma_j^2)}\right) \tag{4.8}$$

Finally, plugging Eq. 4.5 into Eq. 4.7 via Eq. 4.8 produces an analytical distance function for our windowed pdfs as follows,

$$\begin{aligned} d(p_t(\mathbf{x}), p_s(\mathbf{x})) = \frac{1}{W^2(4\pi\sigma^2)^{d/2}} \sum_{w,v=0}^{W-1} \Big[& \exp\Big(-\frac{(\vec{x}_{t-w}-\vec{x}_{t-v})^2}{4\sigma}^2\Big) \\ & - 2\exp\Big(-\frac{(\vec{x_{t-w}}-\vec{x_{s-v}})^2}{4\sigma}^2\Big) \\ & + \exp\Big(-\frac{(\vec{x_{s-w}}-\vec{x_{s-v}})^2}{4\sigma^2}\Big)\Big] \end{aligned}$$

4.1.3 HMM construction

The idea behind unsupervised segmentation is to represent the data sequence in terms of a smaller set of prototype pdf's found from within the sequence itself. To do that, we define an HMM over a set S of sliding windows. Each state in the HMM corresponds to one window W, which is represented by a pdf. The continuous observation probability distribution, or the probability to observe pdf p_t at state s is given by,

$$p(p_t(\mathbf{x})|s) = \frac{1}{\sqrt{2\pi\varsigma}} \exp\Big(-\frac{d(p_t(\mathbf{x}), p_s(\mathbf{x})}{2\varsigma^2}\Big) \tag{4.9}$$

The initial state distribution $\{\pi_s\}_{s\in S}$ is given by uniform distribution, i.e. $\pi_s = \frac{1}{N}$ where N is the number of states in the model. The transition probability between states is defined by,

$$a_{ij} = \begin{cases} \frac{k}{k+N-1}, & \text{if } i = j \\ \frac{1}{k+N-1}, & \text{if } i \neq j \end{cases} \tag{4.10}$$

The transition probability is designed such that the probability to stay in the same state is k times more likely than transitions to any other states. k is the ratio $\frac{a_{ii}}{a_{ij}}$ where $i \neq j$. It determines a current state's resistance to change.

The Viterbi algorithm [51] is then applied to find the optimal state sequence given the HMM. The resulting state sequence represents the sequence of prototype pdf's that have the maximum probability of generating the observed sequence of pdf's. Segment points are generated by cutting the sequence at points where the state changes.

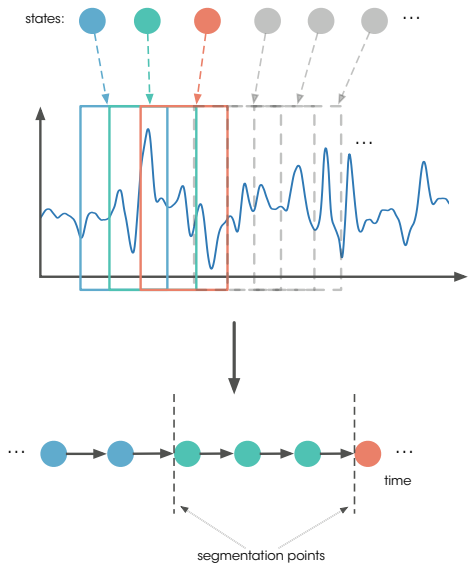


Figure 4.2: HMM definition.

4.1.4 Segmentation algorithm

4.1.4.1 Offline version

The traditional Viterbi algorithm computes the optimal state sequence using maximum likelihood function L. However, we can reformulate the algorithm to compute minimum of the cost function $-log(L)$ instead. This way, we can avoid numerical problems caused by products by replacing them with sums [51]. Using this formulation, we define the offline segmentation algorithm below, which takes as inputs the distance matrix $D = (d_s, t)_{s,t \in S}$ and C which is the regularizer that encodes the transition cost:

In the algorithm, $S = T$ since we constrain states to be the pdfs that make up the time-series. $c[s,t]$ represents the cost of the optimal sequence ending at state s at time t. $D[s,t]$ is the distance between two pdf's z_T is the termination point from which the algorithm backtracks through the minimum costs logged at each time step in order to find the minimum cost sequence. C is a regularization constant given by $C = 2\varsigma^2 log(k)$. It embeds the transition probability defined previous and determines the cost to switch from current state to a new state.

Algorithm 1 Kohlmorgen-Lemm offline segmentation algorithm pseudocode

procedure OFFLINEVITERBI(D, C)
  for all $s \in S$ **do** Initialization
    $c_1[s,1] = D[s,1]$
    $c_2[s,1] = 0$
  for $t = 2,3,...,T$ **do** Recursion
    for all $s \in S$ **do**
      $c[t,s] = D[s,t] + min_{k \in S}(c[k,t-1] + C \times (1 - \delta_{s,k}))$
      $c_2[t,s] = argmin_{k \in S}(c[k,t-1] + C \times (1 - \delta_{s,k}))$
  $z_T = argmin_{k \in S}(c_1[k,T])$ Termination
  $X_T = s_{z_T}$
  for $t = T, T-1,...,2$ **do** Backtracking
    $z_{t-1} = T_2[z_t, t]$
    $X_{t-1} = s_{z_{t-1}}$

4.1.4.2 Online version

The offline segmentation can be turned into an online algorithm by incrementally building the state path matrix as new state is observed from streaming data. cost at the new time step is computed by reusing the likelihood and optimal state sequence from the previous time step.

Algorithm 2 Kohlmorgen-Lemm online segmentation algorithm pseudocode

procedure ONLINEVITERBI(D,C)
  $o[1] = 0$
  for $t = 1,2,...,T$ **do**
    Generate new state k in the HMM corresponding to teh window of data at the current time step
    for $t = 1,2,...,T-1$ **do**
      $c[k,t] = D[k,t] +$

$$\begin{cases} 0 & \text{if } n \equiv 0 \\ min(c[k,t-1], o[t-1] + C) & \text{else} \end{cases}$$

    if $c[k,t] < o_t$ **then** $o_t = c[k,t]$
  for all $\forall s \in S$ **do**
    $c[s,T] = D[s,T] + min(c[s,T-1], o[T-1] + C)$
  $o[T] = min_s(c[s,T])$

Similar to the offline case, the minimum state path must be tracked simultaneously during the computation. This can be done by storing the sequence of points that have switched states in each path.

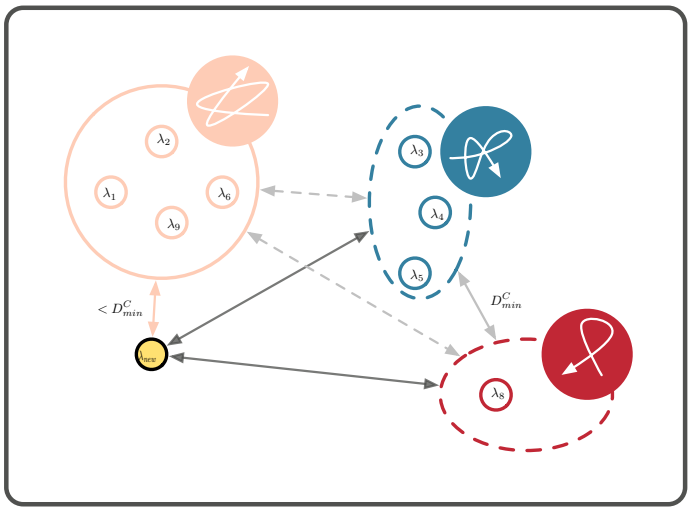


Figure 4.3: Illustration of the clustering algorithm. Each time a new data arrives, it is encoded into an HMM and compared with existing exemplary models. If the distance is smaller than a threshold, it is grouped into that model cluster.

4.2 Incremental Clustering and Learning of Gesture-Sound Primitives

After the streaming data has been segmented into primitives using the segmentation algorithm described above, an incremental clustering algorithm is applied to label the segments while simultaneously learning the mapping between gesture and sound data (Figure 4.3. The main task in the incremental clustering module is to group similar segments together based on their similarity and learn an HMM for each group of similar segments. Each of these groups are referred to as "primitives". They represent a group of similar gestures whose model is used to generate representative sequence of that particular motion.

We adapt an incremental algorithm proposed by Kulic et al. [39] to cluster and learn gesture-sound segments. Figure ?? shows a schematic of the clustering algorithm and the adapted algorithm pseudocode is shown in Algorithm 3.

4.2.1 Observation sequence encoding

The first step in the algorithm is to encode the newly observed segment into HMM. We use the multimodal HMM detailed in [24] to map gesture data to sound data.

Algorithm 3 Incremental clustering algorithm pseudocode (adapted from Kulic's original work [39]

```
1: procedure INCREMENTALCLUSTERING
2:     Learn HMM λ_i for observed segment O_i
3:     for Each existing cluster C_j do
4:         Calculate distance, D_ij between λ_i and HMM λ_{C_j}
5:         Keep track of the minimum distance between new observation and an
   existing cluster.
6:     if D_ij < threshold then
7:         Place λ_i into cluster C_j and retrain the cluster
8:     else
9:         Form a new cluster C_i, containing λ_i
```

4.2.2 HMM distance calculation

Once the segment is encoded into an HMM, it can be compared to the HMMs in existing clusters. If no cluster has been formed yet, the segment forms a new cluster. Otherwise, the distance between two HMMs is calculated using Kullback-Leibler divergence [51]:

$$D(\lambda_1, \lambda_2) = \frac{1}{T}(\log P(O^2|\lambda_1) - \log P(O^2|\lambda_2)) \tag{4.11}$$

In the equation, λ_1 and λ_2 are two HMM's, O^2 is the observation sequence generated by λ_2, and T is the length of the observation sequence O^2. This distance measure is non-symmetric. The symmetric version is defined by taking the average of two intra HMM distances:

$$D_s = \frac{D(\lambda_1, \lambda_2) + D(\lambda_2, \lambda_1)}{2} \tag{4.12}$$

Furthermore, as this distance measure does not satisfy the triangular equation, it measures the pseuo-distance between two models rather than the actual distance.

4.2.3 Clustering

The newly built model is compared against all existing clusters using the distance measure defined above. We employ an adaptive threshold for determining if a node should be merged into one of the existing clusters or create a new cluster on its own. The adaptive threshold is defined as:

$$thresh = \alpha D^C_{min} \tag{4.13}$$

where D^C_{min} is minimum intra distance between all existing models in the cluster space and α is a multiplication factor.

If the distance D_ij between the new observation sequence O_i and its closest cluster C_j is smaller than $thresh$, the newly observed sequence will be included into C_j. Then, the HMM for C_j is retrained with all observation sequences from the HMMs in C_j and the new observation sequence. However, if D_ij is larger than $thresh$, cluster C_j will not be a possible candidate for generating the new observation sequence. And if the entire cluster space has been searched and no match has been found, a new cluster with the new observation is created.

CHAPTER 5

EXPERIMENTAL RESULTS

We have introduced an incremental learning system for segmenting and clustering multimodal observation data. The system is expected to be incorporated into the existing Mapping-by-Demonstration framework for gestural control of sound synthesis. Instead of having to manually segment and label the data before learning their intrinsic motion-sound relationships, our system handles this part of the work automatically. We expect our segmentation system to output segmentation points similar to those from manual segmentation. In addition, we expect our incremental clustering algorithm to perform accurate labeling of the segmented data from the automatic segmentation step. Finally, we expect the integrated system to learn the motion-sound mappings. In this section, we designed experiments to quantitatively evaluate the performance of our system according to these expectations.

5.1 Evaluation Method

Choosing a performance measure to evaluate a complex multimodal interaction system is not straightforward. There are three separate tasks we are concerned with in evaluating the system: segmentation, labeling, and learning of the mappings. Each of these tasks needs its own set of evaluation criteria to address the different challenges at hand.

For automatic segmentation, one of the challenges lies in choosing ground truth data. Segmentation points annotated by human subjects can be loosely defined or ambiguous. The start and end points of a gesture are often subjective and can

vary from one subject to another. Furthermore, the specificity of a segmentation can also be a point of contention, i.e., at what level of granularity should one segment a continuous sequence? For example, music-related gestures, such as those incurred during an instrumental performance, can oftentimes be interpreted at different time scales, "from the more extended gestures that shape rhythmical, textural, or melodic patterns, to the micro-gestures that create minute inflections of pitch, dynamics, and other features in the course of a single tone" [29]. In addition to granularity, degrees of freedom (DoF) can also add to the complexity of the segmentation problem. A complete gesture is often made up of activities in different DoF captured by sensing devices placed at different parts of the body, but not all DoF may be directly contributing to the overall perception of movement. Since we do not consider each DoF individually as data quickly become unwieldy as dimensions increase, filtering out activity along DoF that do not play a significant role in influencing the direction of a movement will help the segmentation algorithm perform more accurately. All these different factors are to be considered when designing a gesture segmentation system and its evaluation.

For the labeling task, the challenge lies in finding a good measure for comparing the similarity between segments from the segmentation step in order to cluster similar segments together. The algorithm should ensure that intra-distances within a cluster remains small while inter cluster distances remains big.

Finally, to assess the model of motion-sound mappings, we are concerned with two tasks: recognition and synthesis. For the recognition step, we first want to see if the system is able to recognize the same gesture in the test dataset when trained on a different dataset. For the synthesis step, we want to compare the generated motion and sound parameters with the original parameters in order to evaluate how well our model represents the relationship. In other words, we asses the algorithm's ability to model accurately the gesture primitives and their associated sound trajectories for synthesis.

To validate the performance of our proposed system, a gesture dataset involving performance of Tai-Chi movements is considered. Given the dataset, we designed two sets of experiments to evaluate, separately, our segmentation module, our incremental clustering module, and finally the combined incremental learning system.

5.1.1 Data collection

We used inertial measurement unit (IMU) as our primary motion capture system as the sensors are inexpensive and the capturing process unobtrusive.

An IMU measures the acceleration (using accelerometer) and the angular

velocity (using gyroscope) of the object or body part to which it is attached. For the Tai-Chi dataset, we recorded movements using the MO interface [52]. Each interface unit contains a 3-axis accelerometer and gyroscope, generating 6 DoF in total each.

5.1.1.1 Tai-Chi dataset

Movement data was collected from two participants performing Tai-Chi movement sequences, one a professional Tai-Chi instructor (henceforth referred to as **I**) while the other a training Tai-Chi student (henceforth referred to as **S**. They were asked to perform common Tai-Chi sequences that ranged in variety, from those with two repeated gestures (*huit-boucle)* to those with up to 15 unrepeated gestures (*sequence-debut*. For all sequences, there is no break between any of the gestures. Sound data was collected from **I** as she vocalizes to her gestures during performance. Movement and sound data were recorded synchronously using mini-MO [52] interfaces and DPA microphone headset, respectively. All performances were also videotaped. 3 MO units were used (totaling 18 DoF), attached at the wrist and arm of the participants as well as on the handle of the sword used during the performance (See Figure 5.1 for set-up detail). Both the motion capture and audio data were sampled at 100Hz.

5.1.2 Data preprocessing

In order to remove noise from the observation signals, we smooth both gesture features (motion sensor data) and audio features (MFCCs extracted from audio recordings) by taking the mean of every 10 frames, and we down-sample the signal by a factor of 5 using an 30-point FIR filter with hamming window. The resulting data points are multi-dimensional feature vectors, with each dimension representing values from the accelerometer and gyroscope sensor.

5.1.3 Analysis procedures: segmentation

To evaluate the segmentation results, we compare the algorithmic segmentation points to the manual segmentation points. Manual segmentation is obtained by a human observer watching the video of the performed gesture sequence and determining the start and end points of a gesture. Then the precision-recall metrics are applied to the algorithmic segmentation against the manual segmentation [1]. *Precision* is defined as the ratio of the number of correct segments reported to the

[1] TP: *true positive*, FP: *false positive*, FN: *false negative*

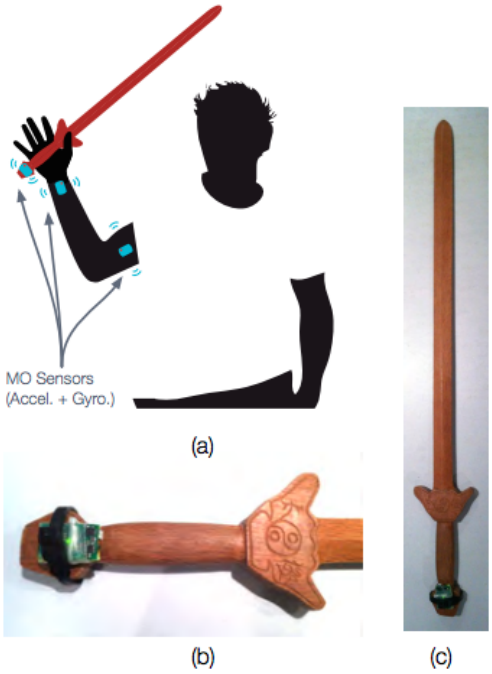


Figure 5.1: Sensor placements for Tai-Chi dataset. Source: [21]. © Copyright: J. Françoise, 2015

total number of correct and incorrect segments reported by the algorithm:

$$Precision = \frac{TP}{TP + FP} \tag{5.1}$$

It measures the algorithm's ability to not label an incorrect segment as correct. *Recall* on the other hand is defined as the ratio of the number of correct segments reported to the total number of correct segments in the manual segmentation:

$$Recall = \frac{TP}{TP + FN} \tag{5.2}$$

It measures the algorithm's ability to find all the correct segments. In addition to precision and recall, we also compute the *F-measure*, which is measures the algorithm's accuracy, given by:

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5.3}$$

We have taken steps in our experimental setup to address the segmentation challenges mentioned previously: *ambiguity*, *granularity*, *dimensionality*.

Ambiguity As all gesture sequences in our datasets are continuous, i.e., smooth transitions from one gesture to another, the exact frame of transition cannot be

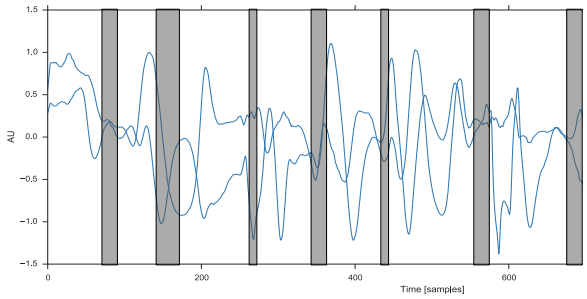


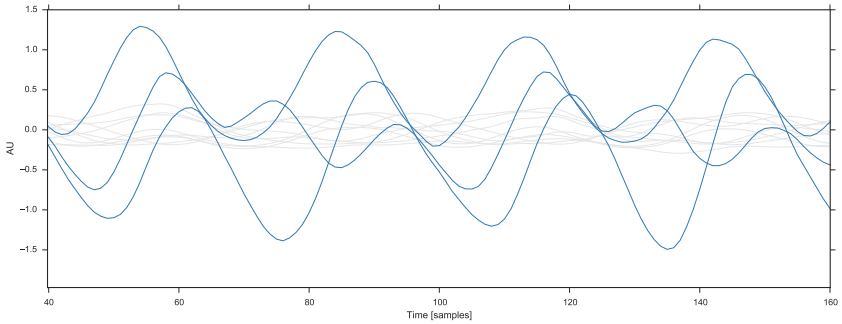
Figure 5.2: Manual segmentation variation example taken from the Tai-Chi dataset. The greyed areas indicate the range of variations between different human labelers.

accurately determined. Therefore, in order to account for this inherent *ambiguity* in manual segmentation, we allow a range of frames to be specified as the ground truth taken by human subjects. The ranges vary in size according to different judgments of segmentation points between human labelers as seen in Figure **??**. A cut generated by the algorithm is considered correct, or true positive, if it falls within its corresponding range, and if it falls outside of this range, the segment is considered false positive. Finally, a segment is false negative if it fails to generate a cut where there should be one.

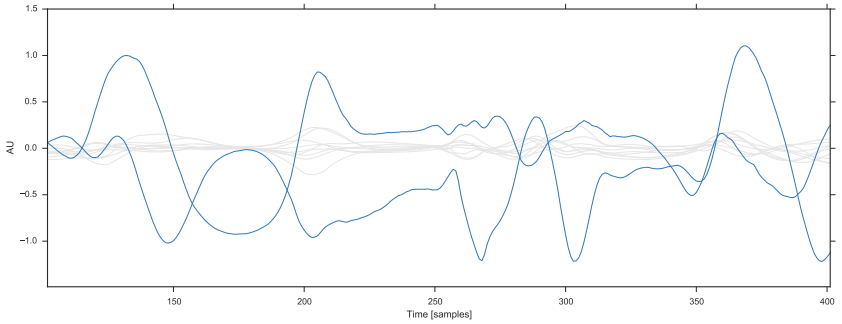
Granularity Next, to address the issue of possible differences in segmentation *granularity* due to varying subjective judgments, we choose the algorithm's parameter values that maximize the segmentation results on a few observation sequences in order to fix the level of granularity taken by the manual segmentation. Those values are then applied to the entire database without adjustments so as to achieve a consistent comparison between experiments.

Dimensionality In high-dimensional data, there often exists a high number of DoFs containing irrelevant activity. In order to improve the algorithm's performance in handling data with *high-dimensionality*, we follow a similar approach as Lin et al. [43] to extract the significant DoFs. DoFs are considered significant if they undertake large range of motions. These DoF are selected by first computing the standard deviation of each DoF in a given data sequence. The obtained standard deviations are then grouped via k-means clustering where $k = 2$. The cluster with the larger centroid contains DoFs that have the highest variations and hence are taken as significant features for the given data sequence. The feature space can be further reduced by calculating correlations between the significant DoFs and removing the redundant DoFs when correlations are found to be over 80%. Figure

?? shows the extracted significant DoFs (denoted by dark lines) of two sequences from the Tai-Chi dataset.



(a) huit-boucle



(b) sequence-debut

Figure 5.3: Extraction of significant DoFs from two example Tai-Chi sequences. The dark lines indicate the significant DoFs whereas the lighter lines indicate the insignificant DoFs.

5.1.4 Analysis procedures: incremental clustering

The purpose for this set of experiments is to determine if the clustering algorithm based on Kullback-Leibler divergence is able to produce accurate labels. In this part, we test the clustering algorithm on manually segmented but unlabeled data. Because our incremental clustering algorithm combines the learning of the motion-sound relationship, we cannot dissociate the evaluation of the correctly labeled segments from the evaluation of the learned HMM-based models. Therefore, in addition to labeling accuracy of the segments, we also evaluate the system's ability to generate corresponding sound and motion parameters.

5.1.5 Analysis procedures: combined incremental learning system

In this set of experiments, we evaluate the final synthesis results of the full combined algorithm. The incremental clustering module takes as inputs segmented data from the automatic segmentation module and performs simultaneous clustering and learning of these motion-sound pairs. We compare the synthesis results from our extension of the system to results from the original system proposed by Françoise et al. [21] in order to assess the feasibility of such an extension.

5.2 Segmentation of Continuous Data Sequences

The 1st experiment will motivate the use of multimodal approach to segmentation of coupled motion and audio sequences. The 2nd experiment validates the consistency of the algorithm by comparing segmentation results of same sequences performed by different subjects. Finally, the last experiment further validates algorithm's consistency by testing it on ambiguous musically-inspired gestures. It aims to determine if the algorithm produces the same segmentation cuts in repetitions of the same sequences.

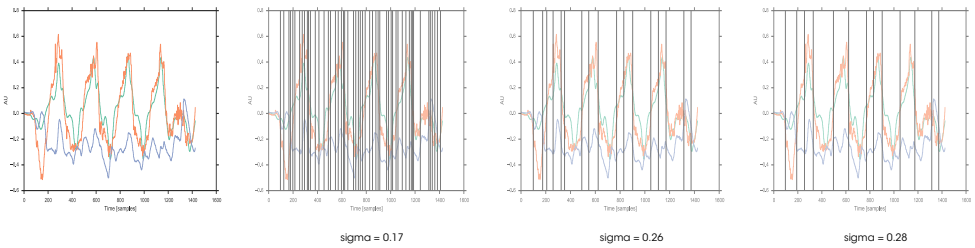
5.2.1 Parameter tuning

Before evaluating the segmentation results, we first study the behaviors of the small set of free parameters required by the algorithm. They are: σ (bandwidth of the Guassian kernel), C (transition cost to a new state), and W (window size). Although some parameters (σ) can be automatically estimated from the distribution using methods detailed in the proposed approach section, further tuning is necessary in order to achieve optimal results. For the sake of exposition, we evaluate the parameters using just one of the motion sequences from the Tai-Chi dataset. The same evaluation procedure for parameter selection is applied to all other motion sequences in subsequent experiments. In the experiments that follow, the exact values are not as important as the behavior when changing these values.

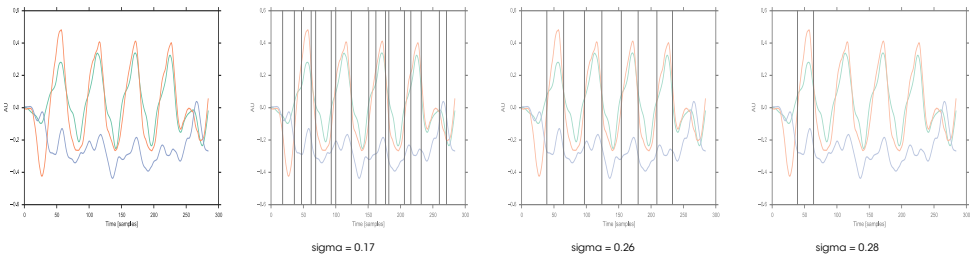
5.2.1.1 Influence of σ

σ is the bandwidth, or the smoothing parameter, of the Gaussian kernel. It controls how finely grained the pdf will represent the underlying data. To illustrate its effect, we performed segmentation on a short sub-sequence from the Tai-Chi dataset, varying the σ value at each trial. We also wanted to see the effect filtering and smoothing has on the segmentation, so we compared segmentation results

from un-preprocessed gyroscope data gathered from one sensor with results from preprocessed version of the same data. Figure 5.4 shows the results. We see that,



(a) Un-preprocessed data

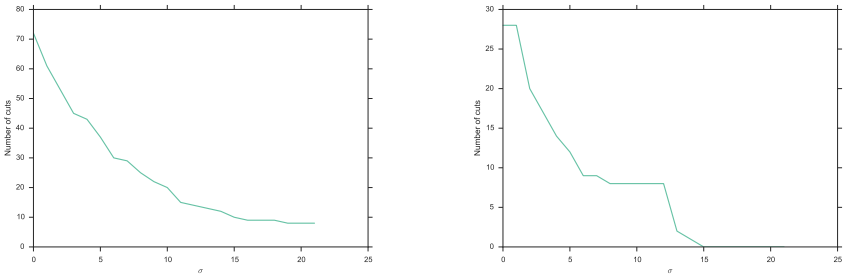


(b) Preprocessed data

Figure 5.4: Influence of σ on segmentation over time. The leftmost plot shows the original signals on which the segmentation is performed. (a) shows the segmentation results on raw data while (b) shows the segmentation results on smoothed data.

across both un-preprocessed and preprocessed data, as σ decreases, the number of segmentation points increase, which aligns with its expected behavior of overfitting the data when its values are small. σ therefore plays the role of determining the granularity of the segmentation. In addition to the general behavior of σ as it increases its value, we also want to investigate the effect smoothing has on the segmentation result. As we can see, un-preprocessed data contains more noise of which σ is sensitive to pick up. As a result, at the same σ values, there will be more cuts detected by the algorithm on un-preprocessed data than on preprocessed data with smoother curves.

Furthermore, we can see in Figure 5.5 the evolution of the number of segmentation cuts as σ changes values and their differences across noisy signals versus smooth signals. The figure shows that when there is less noise in the signal, the number of segmentation converges to zero quicker. This indicates that when we are dealing with noisy data such as those with large DoF, we will expect to choose a larger σ value than in cases of smooth signals.

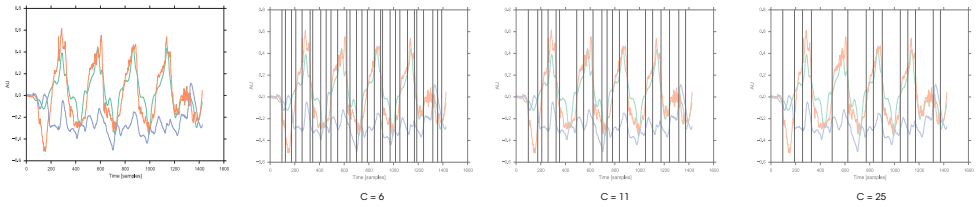


(a) Un-preprocessed data (b) Preprocessed data

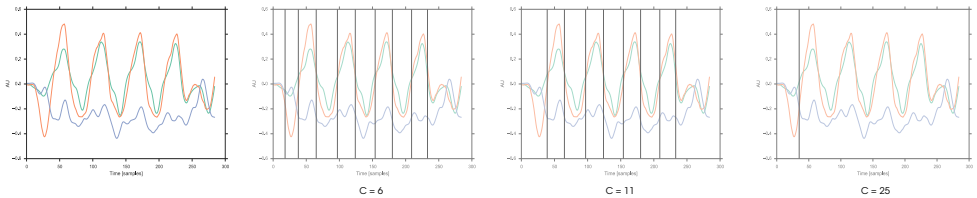
Figure 5.5: Number of cuts over time as σ increases. Two cases are examined here: segmentation on un-preprocessed data and segmentation on preprocessed data.

5.2.1.2 Influence of C

C encodes the cost in changing to a new HMM state in the segmentation algorithm. It can be interpreted as the resistance of a state to change from its current state. It has similar behaviors as σ in that as its values decrease, more segmentation cuts result (Figure 5.6). Figure 5.6



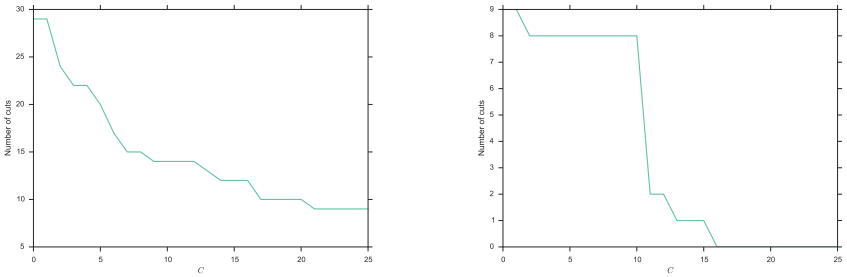
(a) Un-preprocessed data



(b) Preprocessed data

Figure 5.6: Influence of C on segmentation over time.

C is also subject to the same influence signal noise has on the number of cuts (Figure 5.7). However, it is worth noting that C is much less sensitive to value changes than σ. The range in which σ goes from over-segmentation to no segmentation is very small – [.15, .36], whereas for C, its range is much larger – [5, 30]. This indicates that C will be chosen at larger intervals than σ.



(a) Un-preprocessed data (b) Preprocessed data

Figure 5.7: Number of cuts over time as C increases.

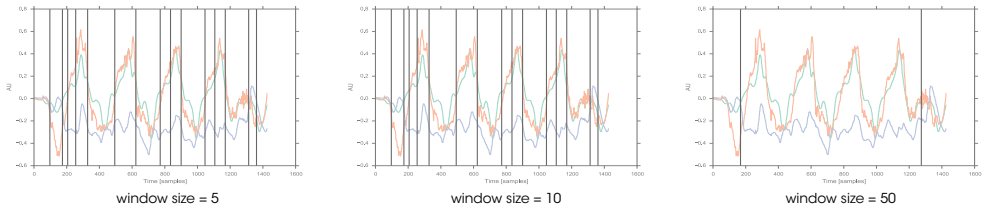
5.2.1.3 Influence of window size

Window size controls the amount of data represented by one pdf. In other words, it determines the size of variations we wish the algorithm to detect. A small window size such as 3 would allow the algorithm to pick up small variations as change points in the signal and hence represent segmentation locations. On the other hand, a large window size would allow the algorithm to treat small variations in the signal as generated by the same pdf and hence no cuts will be given at those points. We notice in Figure 5.8 that the noisier a signal, the more inertia (provided by larger window sizes) is needed for the algorithm to disregard variations. However, when the signal is smooth, the algorithm is able to reach similar results in much shorter amount of time as W contributes the most to the algorithm's computational cost. So when considering a real-time application, small window sizes are desired, hence motivating a preprocessing step involving filtering and smoothing.

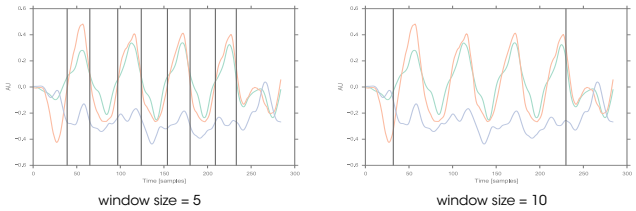
Summary The selection of parameter values is highly dependent on the type of signals received by the algorithm. It also controls the performance of the overall segmentation. While there are no exact values for these parameters that can be applied across all signal types, this section gives an intuition for tuning these parameters in order to optimize the algorithm's performance.

5.2.2 Segmentation results on gesture-only data

In this experiment, we compare the segmentation results on gesture data from 2 types of motion sequences from the Tai-Chi dataset. One sequence, *huit-boucle* contains repetitions of the figure-"8" motion with a different gesture interspersed in between the repetition. Another sequence, *sequence-debut* contains continuous performance of a series of non-repetitive gestures commonly found in Tai-Chi



(a) Un-preprocessed data



(b) Preprocessed data

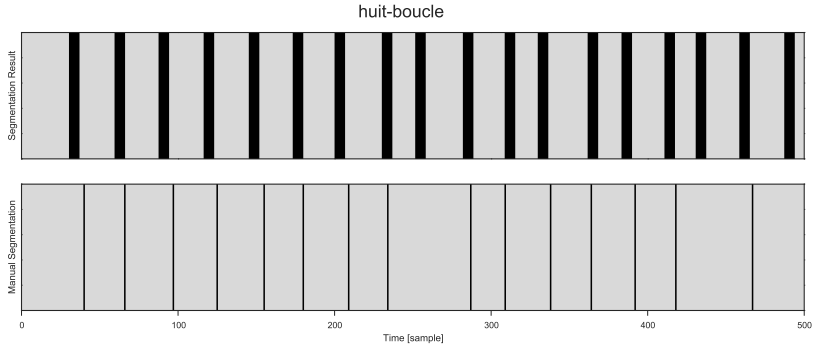
Figure 5.8: Influence of window size on segmentation over time.

movements. The purpose of the experiment is to evaluate the types of gesture sequences the algorithm excels at and identify the weak points of the algorithm. For this experiment, we consider gesture-only data (audio is not yet introduced at this point) in order to evaluate the algorithm's performance on the motion segmentation task. Results from this section will be used to motivate the use of multimodal data. As mentioned previously, data from all sensors are preprocessed (filtered and smoothed) before being sent to the algorithm. For the algorithm parameter values, please refer to Table 5.1.

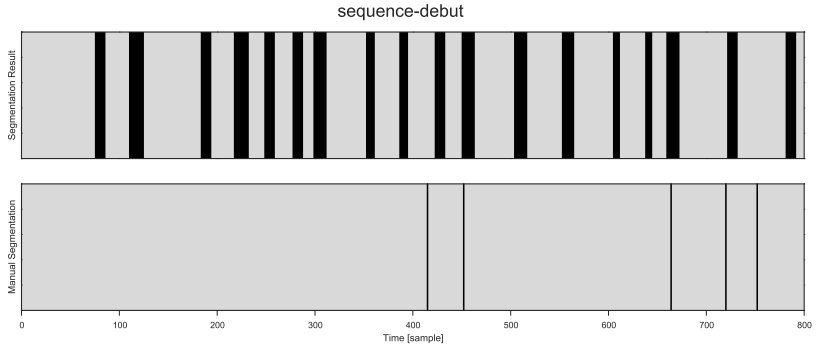
Parameter	σ	C	W
Value	.265	11	1

Table 5.1: Gesture-only segmentation algorithm parameters

Figure 5.9 shows segmentation points generated by the algorithm compared against the manual segmentation points. Figure 5.10 shows the details of the algorithm segmentation with original waveforms. We can see from these two plots that *huit-boucle* performs significantly better than *sequence-debut* given the same parameter settings. In Figure 5.10, we notice the extracted axes in *huit-boucle* are more correlated (move roughly in the same directions) than extracted axes in *sequence-debut*. As a result, the algorithm is able to detect instances where sharp transitions occur. This contributes to the lower accuracy, as represented by the F-measure in Table 5.3, of *sequence-debut*. For both sequences, recall is higher than



(a) Segmentation results of *huit-boucle* sequence



(b) Segmentaiton results of *sequence-debut* sequence

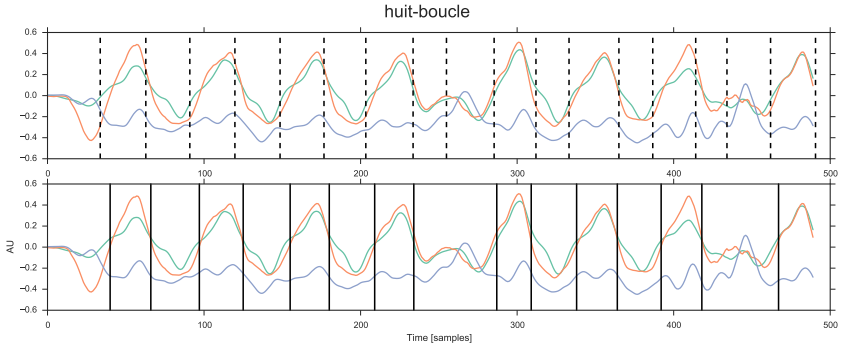
Figure 5.9: Gesture-only segmentation results on Tai-Chi data. It compares motion segmentation points assigned by a human observer with those by the algorithm.

precision, implying that while the algorithm has ability to report segmentation that matches well with the manual segmentation, it is likely to produce noise in the segmentation.

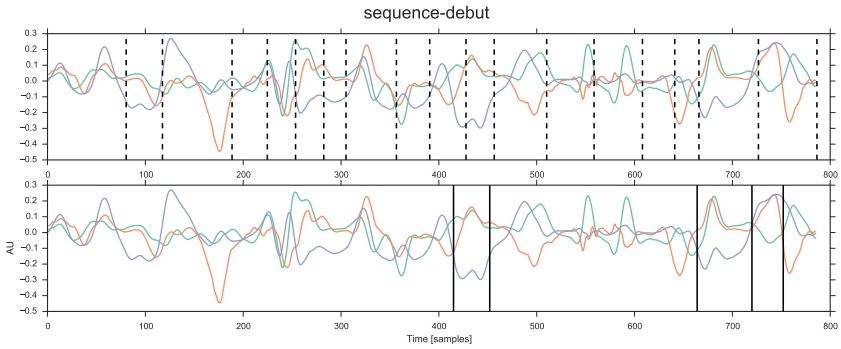
Summary With gesture-only data, the algorithm performs well with sequences that have clear visual pattern. In *huit-boucle*, it is clear that the sequence contains a repetitive gesture with 2 different gestures laced in between. However for *sequence-debut*, it is unclear at the signal level, to even human observers, where segmentation cuts should be placed due to the lack of correlation between extracted gesture features.

5.2.3 Improving segmentation using multimodal features

In this experiment, Canonical Correlation Analysis (CCA) is used to extract the most correlated motion and sound features. As a reminder, the sound features are represented by MFCCs extracted from audio recordings of the vocalizations made by the participant while performing Tai-Chi movements. The gesture and



(a) Segmentation results of *huit-boucle* sequence



(b) Segmentaiton results of *sequence-debut* sequence

Figure 5.10: Gesture-only segmentation results on Tai-Chi data (detailed). It compares motion segmentation points assigned by a human observer with those by the algorithm.

sound features are normalized to have zero mean and one variance. The hybrid data then gets passed to the segmentation algorithm. In order to motivate the use of multimodal data, we compare the multimodal segmentation results to the results of gesture-only segmentation from previous section. Table 5.2 shows the parameters used to analyze all datasets in this experiment.

Parameter	σ	C	W
Value	.28	13.5	2

Table 5.2: Hybrid segmentation algorithm parameters

The algorithm is tested on the same sequences from the previous experiment that contain accompanying audio with parameters shown in Table 5.2. For both sequences, the multimodal approach shows comparable result or improvement. 5.11.

The precision-recall table is shown in 5.3.

Sequence Type	Sequence	Precision	Recall	F-measure
huit-boucle	Gesture-only	**100**	88	**94**
	Hybrid	94	**94**	**94**
sequence-debut	Gesture-only	80	25	38
	Hybrid	**81**	**87**	**84**

Table 5.3: Precision-Recall table of gesture-only and hybrid segmentation results.

Summary Segmentation results improve significantly for sequence-debut while keeping the parameters the same. Segmentation based on multimodal data thus demonstrates more consistent and stable results across different observation sequences. This is because the audio information can provide better representation of the direction of the movement, hence improving the segmentation results.

5.2.4 Discussion

The algorithm has shown promising results in segmenting two types of motion sequences: one with repetitive gestures and another with continuous, non-repetitive gestures. The algorithm is able to arrive at similar cuts as manual segmentation.

The algorithm's performance is highly dependent on a good preprocessing step that extracts pertinent information about the gestures performed and the audio that accompanies the gestures. Therefore, the use of CCA has been shown to improve the segmentation result when compared to gesture-only segmentation. We also note that higher-dimensional vectors perform worse as they have increased probability to contain irrelevant data, contributing to the overall noise of the signal. So dimensionality reduction methods are applied as a preprocessing step.

5.3 Incremental Clustering of Motion-Sound Segments

In this section, we will test the performance of the incremental clustering algorithm on manual segmentation data. For the 1st experiment, we will evaluate the accuracy of the labels produced by the clustering algorithm. Since motion-sound relationship is simultaneously modeled during the clustering step, we will also evaluate how well the models have learned the relationship by re-generating motion trajectories.

The incremental clustering algorithm is tested on the two Tai-Chi sequences used in earlier experiments.

5.3.1 Labeling accuracy

The incremental learning algorithm outputs a set of labels associated with each sequence according to clustering results. These labeling results from the algorithm are compared against the original labels show in Figure 5.12. The algorithm is able to reproduce the original labels for *sequence-debut*. For *huit-boucle*, the algorithmically generated labels differ from the manual labels in sections where transitions occur. In those transitional gestures, the gestures are divided into two smaller gestures by the human observer. However, at the signal level, those two gestures are combined as one gesture by the algorithm.

5.3.2 Resynthesis of movement parameters

With the labels generated from the clustering algorithm, we can resynthesize the movement trajectories from three axes of the gyroscope data. Results are shown in Figure 5.13. Trajectories from both sequences can be re-synthesized with high fidelity. In *huit-boucle*, the synthesized trajectories lose precision in sections where labeling results from the previous section differ.

Summary Incremental clustering has been shown to generate labels that are consistent with manual labels, provided that the manual labels correspond directly to changes in the gesture at the signal level. We have also shown in this section that given algorithmically generated labels, the system is able to recover the original motion trajectories with high fidelity.

5.3.3 Discussion

We have shown that the automatic segmentation algorithm works well with signals that have been preprocessed. The performance of the algorithm is highly dependent on the parameter values, thus making parameter tuning a necessity in ensuring good performance. However, the process of manual tuning can be labor-intensive. We have supplied intuition for the setting of the three free parameters used by the model.

For incremental clustering, the algorithm has displayed almost identical labeling results as manual labels. Because models of motion and sound are learned simultaneously during clustering, movement trajectories can be generated in real-time. The resynthesis results are shown to be comparable to results from batch learning. This shows promise in generating the corresponding sound trajectories which will be covered in the next section.

5.4 Combining Online Segmentation and Incremental Clustering: An Incremental Learning System

The incremental clustering algorithm is able to incrementally label and learn the gesture primitives. In this section, we perform experiments with segments generated from the segmentation algorithm.

5.4.1 Labeling accuracy

To test labeling accuracy, the incremental clustering algorithm is run on both manual segments and algorithmic segments. Their labeling results are compared to the ground truth labels assigned by a human subject. In Figure 5.14, we have labeling from the repetitive sequence – namely *huit-boucle*. We see that the incremental labeling of the algorithmic segments outputs accurate labels except in the middle sections where the 2 separate gestures are treated as one complete gesture by the segmentation algorithm. This reflects inaccuracy on the part of the segmentation algorithm rather than the incremental clustering algorithm.

For non-repetitive sequences, we ignore labeling errors that are results of errors in algorithmically-produced segments. Instead, we focus on what is consistent here across the 2 incremental clustering result, which is that all labels are different, indicating that no repetition is found by the incremental clustering algorithm, which agrees with the ground truth labels.

Summary We can see from this section that the labeling results from the clustering algorithm can help us better understand the meaning of the segments. For example, in the non-repetitive sequence where the segmentation algorithm performs less well, we can see from the labeling results that the reason is because there is no repeating pattern in the sequence.

5.4.2 Movement trajectory resynthesis

Our goal of the incremental system is to resynthesize motion trajectory from learned models, which can then be used to generate the corresponding sound parameters for audio synthesis.

Figure 5.16 shows the motion trajectory of the repetitive *huit-boucle* sequence for one axis of the gyroscope data resynthesized from the trained segment models built during the incremental learning phase. The solid lines are the trajectories generated using algorithmically determined segmentation and labels. The dotted lines are generated using manual segmentation and labels from incremental clustering. Finally, the dashed lines are the original motion trajectory. We can see

that the combined algorithm is able to reproduce the original motion trajectory quite well. Overshooting or undershooting happens when there's a slight offset in the algorithmic segmentation.

Figure 5.17 shows resynthesis results from the non-repetitive sequence. Again, the motion trajectory is recovered quite well. Inaccuracies result again from errors in the segmentation. We can see that the problematic section in the resynthesis corresponds to the problematic segmentation produced by the segmentation algorithm.

5.4.3 Sound trajectory synthesis

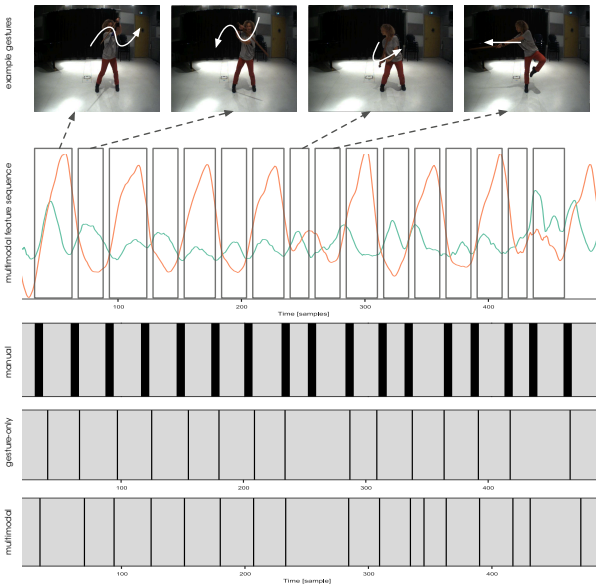
Figure 5.18 shows an example sound trajectory synthesized from the movement trajectory by learning a multimodal model on the repetitive sequence. Trajectory from the combined algorithm are shown in solid line, from incremental-only in small dashes, and from ground truth in large dashes. The sound parameter used is loudness.

Considering the inaccuracies that have potentially accumulated along the way from the segmentation algorithm, the clustering algorithm, and finally the motion resynthesis, the sound parameter generation results are good and follow quite well the directions of the original trajectory.

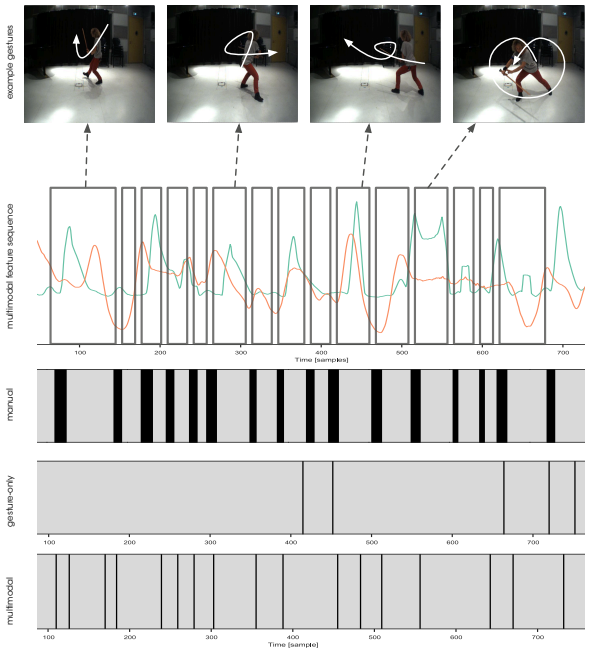
5.4.4 Discussion

We have succeeded in performing the complete workflow of the incremental learning system, from unsupervised segmentation, to incremental clustering, to motion trajectory resynthesis, and finally to the produced sound trajectory synthesis. There are many complex components involved and many details to look at and compare at each step.

We conducted a set of experiments to evaluate the performance of joint segmentation and clustering, which is a challenging task. For segmentation, the algorithm is able to arrive at similar cuts as manual segmentation. For incremental clustering, the labeling performance is highly dependent on good segmentation results. If good segmentation is given, the combined incremental clustering and learning algorithm is able to arrive at comparable results as results from non-incremental learning. This shows promise toward developing an incremental learning system for interactive design of gesture-sound mappings.

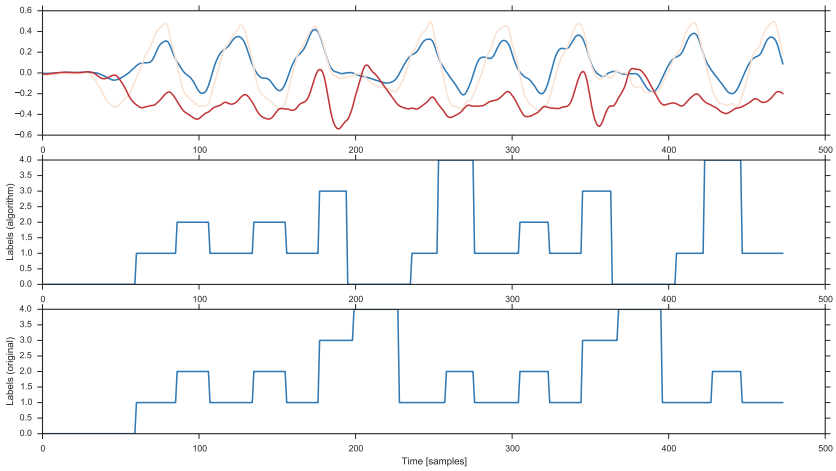


(a) *huit-boucle*

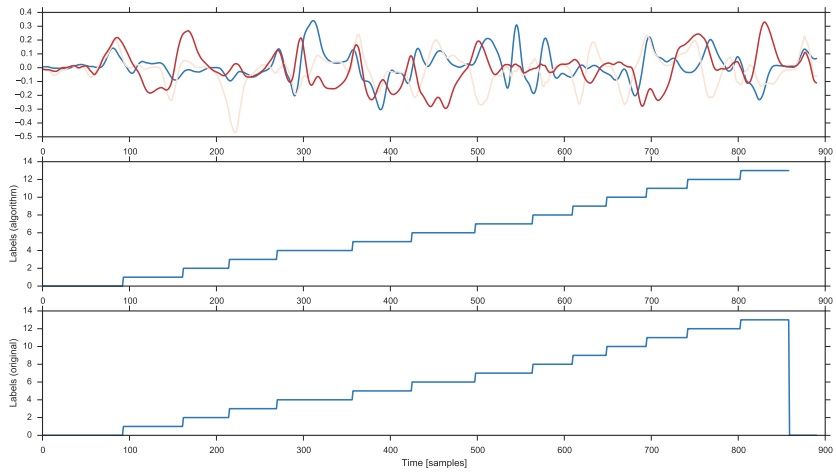


(b) *sequence-debut*

Figure 5.11: Comparison of segmentation based on gesture-only data and multimodal data. The waveform is comprised of two most correlated sound (corresponding to loudness) and motion features (corresponding to gyroscope feature) as found by CCA. The green line is the sound feature whereas the orange line is the motion feature.

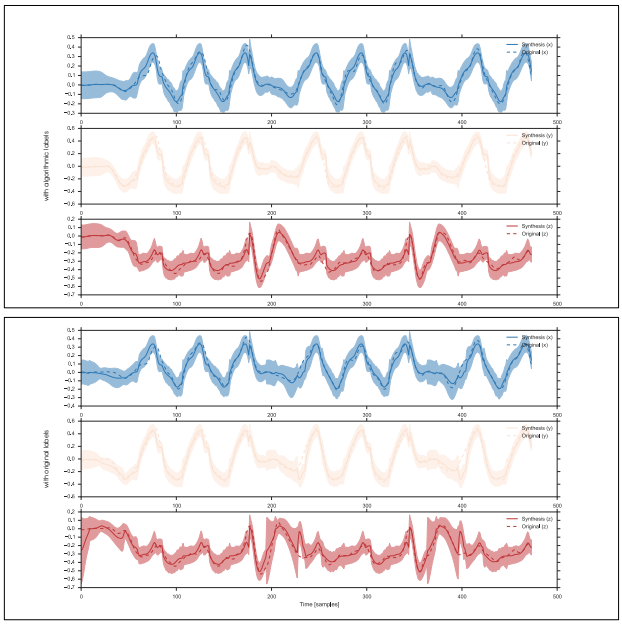


(a) Labeling results of *huit-boucle* sequence

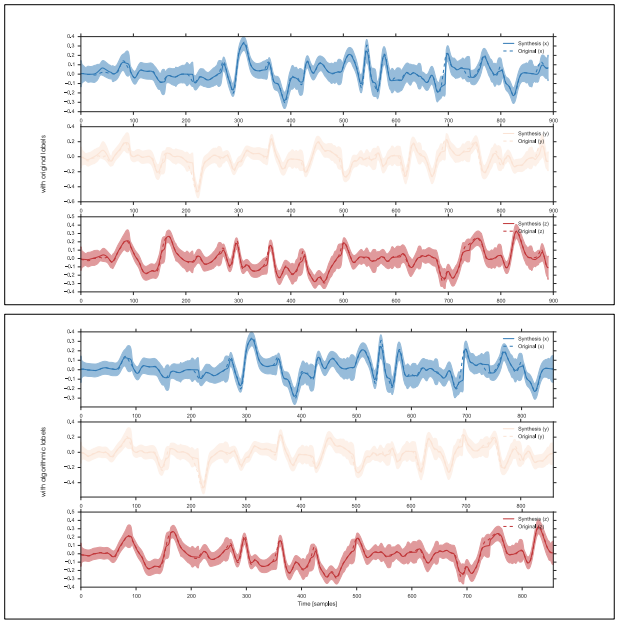


(b) Labeling results of *sequence-debut* sequence

Figure 5.12: Labeling results from incremental clustering, compared with original labels.



(a) Motion trajectory resynthesis comparison for *huit-boucle* sequence



(b) Motion trajectory resynthesis comparison for *sequence-debut* sequence

Figure 5.13: Motion trajectory resynthesis of three axes from gyroscope data. Shaded regions indicate the variance between algorithm-generated trajectory and original trajectory.

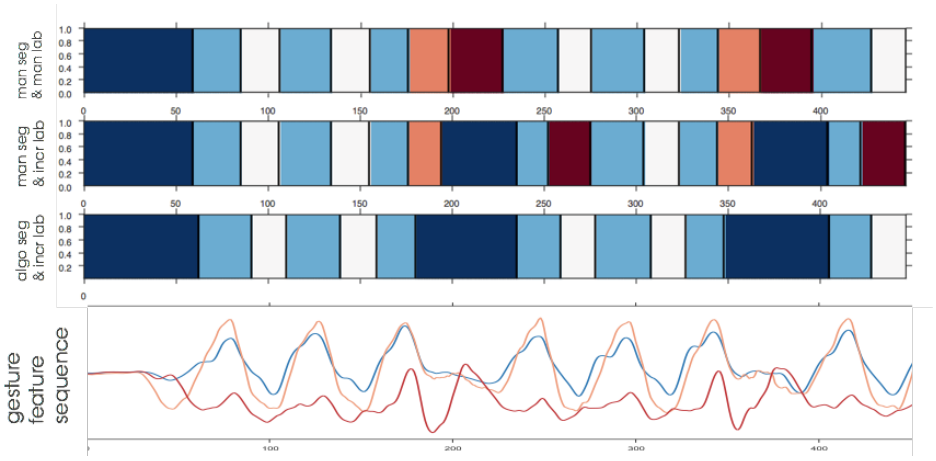


Figure 5.14: Labeling results of repetitive sequence (*huit-boucle*). Colors represent labels.

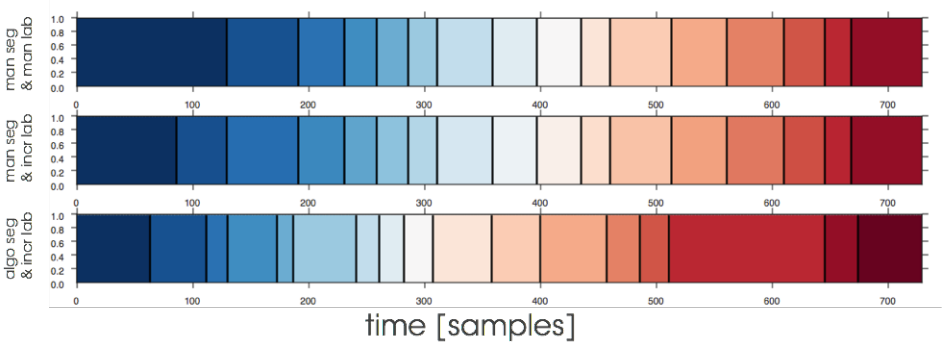


Figure 5.15: Labeling results of non-repetitive sequence (*sequence-debut*). Again colors represent labels.

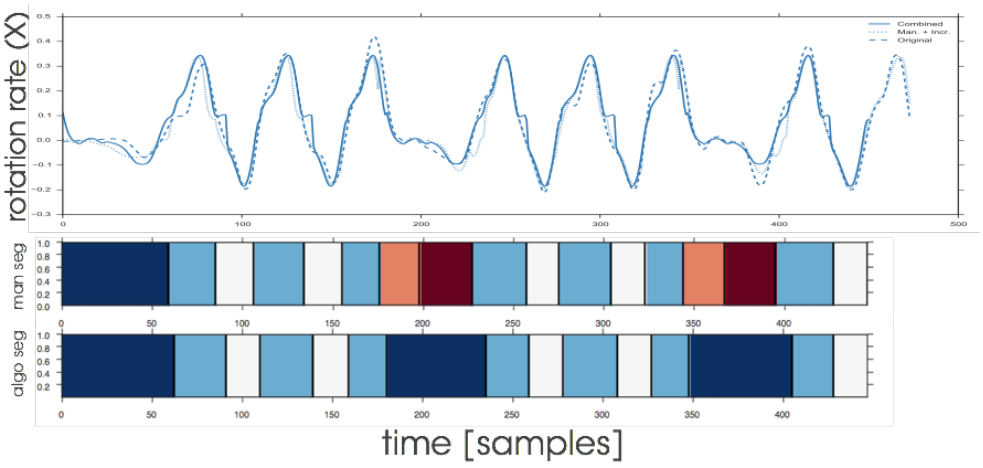


Figure 5.16: Motion resynthesis results for repetitive sequence (*huit-boucle*): combined (solid line), incremental-only (small dashes), ground truth (large dashes).

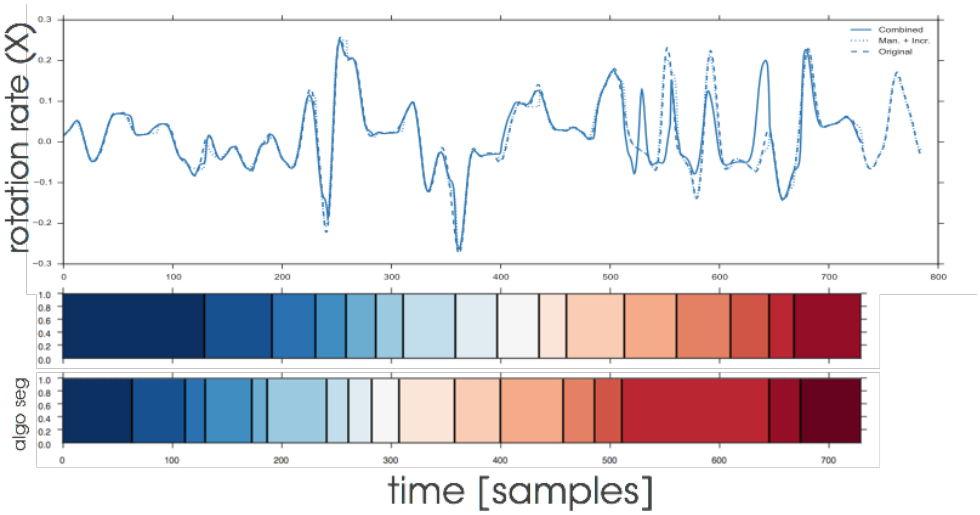


Figure 5.17: Motion resynthesis results for non-repetitive sequence (*sequence-debut*): combined (solid line), incremental-only (small dashes), ground truth (large dashes).

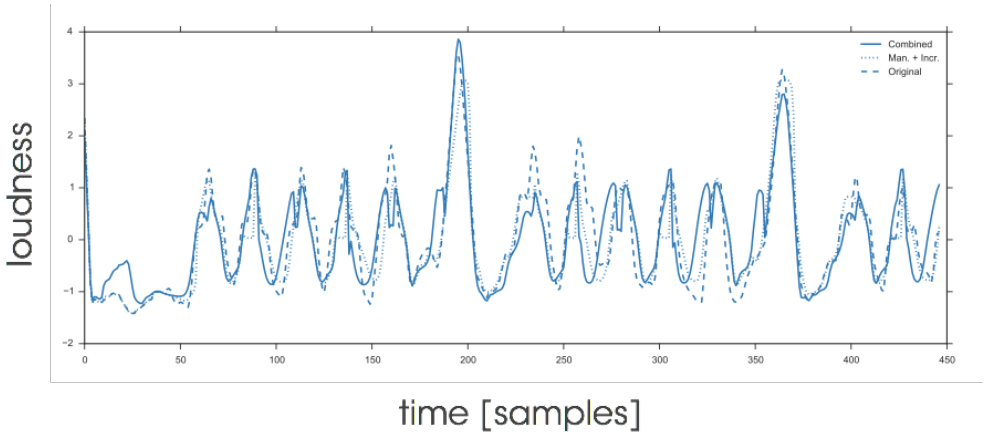


Figure 5.18: Comparison of synthesized sound trajectory: combined (solid line), incremental-only (small dashes), ground truth (large dashes)

CHAPTER 6

CONCLUSION

This thesis has introduced an incremental learning system for building models of gesture and sound relationships. It serves as an extension to the Mapping-by-Demonstration system proposed by Françoise et al. [21]. It consists of two modules – automatic segmentation and incremental clustering. The automatic segmentation module breaks up continuous streaming motion-sound data into distinct segments, whereas the incremental clustering module performs the labeling of the algorithmic segments and simultaneously learns the mapping between gesture and sound components. These two modules combine to construct an integrated online training phase for learning gesture-sound mappings.

We tested our algorithm on a dataset consisted of Tai-Chi movement sequences and vocalizations of the gestures. We have shown that it achieves performance comparable to the original batch learning system. However, the overall performance is highly dependent on the performance of the segmentation algorithm. If the segmentation algorithm produces semantically meaningful segments, the incremental clustering module will be able to group similar segments together and build models that accurate represent the data. The system has demonstrated the promise of an incremental algorithm for enhancing user experience and augmenting learning capabilities in an interactive gestural sound control system. The implementation of autonomous segmentation and clustering algorithms supports a more continuous interaction with the mapping design system.

6.0.1 Summary of Findings

We tested our algorithm on a dataset consisted of Tai-Chi movement sequences and vocalizations of the gestures. We have shown that it achieves performance comparable to the original batch learning system. However, the overall performance is highly dependent on the performance of the segmentation algorithm. If the segmentation algorithm produces semantically meaningful segments, the incremental clustering module will be able to group similar segments together and build models that accurately represent the data. The system has demonstrated the promise of an incremental algorithm for enhancing user experience and augmenting learning capabilities in an interactive gestural sound control system. The implementation of autonomous segmentation and clustering algorithms supports a more continuous interaction with the mapping design system. This work demonstrates a promising move toward a real-time incremental learning system for sound design using gestures.

6.0.2 Future Work

For this round, we tested our data on Tai-Chi datasets. For future work, We hope to complete a more comprehensive study using more complex motion-sound data involving musical gestures. We also hope to develop a complete evaluation methodology for the combined system.

BIBLIOGRAPHY

[1] J. BARBIČ, A. SAFONOVA, J. PAN, C. FALOUTSOS, J. HODGINS, AND N. POLLARD, *Segmenting motion capture data into distinct behaviors*, Graphics Interface, Proceedings of, (2004), p. 185–194.

[2] F. BEVILACQUA, J. RIDENOUR, AND D. J. CUCCIA, *3d motion capture data: motion analysis and mapping to music*, in Proceedings of the workshop/symposium on sensing and input for media-centric systems, Citeseer, 2002.

[3] F. BEVILACQUA, N. SCHNELL, N. RASAMIMANANA, B. ZAMBORLIN, AND F. GUÉDY, *Online gesture analysis and control of audio processing*, in Musical Robots and Interactive Multimodal Systems, Springer, 2011, pp. 127–142.

[4] F. BEVILACQUA, B. ZAMBORLIN, A. SYPNIEWSKI, N. SCHNELL, F. GUÉDY, AND N. RASAMIMANANA, *Continuous realtime gesture following and recognition*, in Gesture in embodied communication and human-computer interaction, Springer, 2010, pp. 73–84.

[5] A. G. BILLARD, S. CALINON, AND F. GUENTER, *Discriminative and adaptive imitation in uni-manual and bi-manual tasks*, Robotics and Autonomous Systems, 54 (2006), pp. 370–384.

[6] C. M. BISHOP, *Neural networks for pattern recognition*, Oxford university press, 1995.

[7] E. BOYER, B. CARAMIAUX, S. HANNETON, A. ROBY-BRAMI, O. HOUIX, P. SUSINI, AND F. BEVILACQUA, *Legos project - state of the art*, tech. rep., 2014.

[8] S. CALINON AND A. BILLARD, *Incremental learning of gestures by imitation in a humanoid robot*, in Proceedings of the ACM/IEEE international conference on Human-robot interaction, ACM, 2007, pp. 255–262.

[9] B. CARAMIAUX, *Studies on the relationship between gesture and sound in musical performance*, University of Paris VI, (2012).

[10] B. CARAMIAUX, F. BEVILACQUA, AND N. SCHNELL, *Towards a gesture-sound cross-modal analysis*, in International Gesture Workshop, Springer, 2009, pp. 158–170.

[11] B. CARAMIAUX, J. FRANÇOISE, N. SCHNELL, AND F. BEVILACQUA, *Mapping through listening*, Computer Music Journal, 38 (2014), pp. 34–48.

[12] B. CARAMIAUX, M. M. WANDERLEY, AND F. BEVILACQUA, *Segmenting and parsing instrumentalists' gestures*, Journal of New Music Research, 41 (2012), pp. 13–29.

[13] S. CHIAPPA AND J. R. PETERS, *Movement extraction by detecting dynamics switches and repetitions*, in Advances in neural information processing systems, 2010, pp. 388–396.

[14] A. DE RITIS, *Senses of interaction: What does interactivity in music mean anyway? focus on the computer game industry*, in Proc. of the Society for Artificial Intelligence and the Simulation of Behavior Conference, 2001.

[15] S. S. FELS AND G. E. HINTON, *Glove-talk: A neural network interface between a data-glove and a speech synthesizer*, Neural Networks, IEEE Transactions on, 4 (1993), pp. 2–8.

[16] R. FIEBRINK AND P. R. COOK, *The wekinator: a system for real-time, interactive machine learning in music*, in Proceedings of The Eleventh International Society for Music Information Retrieval Conference (ISMIR 2010). Utrecht, 2010.

[17] R. FIEBRINK, P. R. COOK, AND D. TRUEMAN, *Play-along mapping of musical controllers*, Citeseer, 2009.

[18] R. FIEBRINK, D. TRUEMAN, AND P. R. COOK, *A metainstrument for interactive, on-the-fly machine learning*, in Proc. NIME, vol. 2, 2009, p. 3.

[19] R. A. FIEBRINK AND P. R. COOK, *Real-time human interaction with supervised learning algorithms for music composition and performance*, Citeseer, 2011.

[20] A. FOD, M. J. MATARIĆ, AND O. C. JENKINS, *Automated derivation of primitives for movement classification*, Autonomous robots, 12 (2002), pp. 39–54.

[21] J. FRANÇOISE, *Motion-Sound Mapping by Demonstration*, phd dissertation, Université Pierre et Marie Curie, 2015.

[22] J. FRANÇOISE, B. CARAMIAUX, AND F. BEVILACQUA, *Realtime segmentation and recognition of gestures using hierarchical markov models*, Mémoire de Master, Université Pierre et Marie Curie–Ircam, (2011).

[23] J. FRANÇOISE, B. CARAMIAUX, AND F. BEVILACQUA, *A hierarchical approach for the design of gesture-to-sound mappings*, in 9th Sound and Music Computing Conference, 2012, pp. 233–240.

[24] J. FRANÇOISE, N. SCHNELL, AND F. BEVILACQUA, *A multimodal probabilistic model for gesture–based control of sound synthesis*, in Proceedings of the 21st ACM international conference on Multimedia, ACM, 2013, pp. 705–708.

[25] ——, *Mad: mapping by demonstration for continuous sonification*, in ACM SIGGRAPH 2014 Studio, ACM, 2014, p. 38.

[26] J. FRANÇOISE, N. SCHNELL, R. BORGHESI, AND F. BEVILACQUA, *Probabilistic models for designing motion and sound relationships*, in Proceedings of the 2014 International Conference on New Interfaces for Musical Expression, 2014, pp. 287–292.

[27] N. GILLIAN, R. KNAPP, AND S. O'MODHRAIN, *A machine learning toolbox for musician computer interaction*, in Proc. of NIME, vol. 11, 2011.

[28] R. I. GODØY AND M. LEMAN, *Musical gestures: Sound, movement, and meaning*, Routledge, 2010.

[29] T. HALMRAST, K. GUETTLER, R. BADER, AND R. I. GODØY, *Gesture and timbre*, Musical gestures: Sound, movement, and meaning, (2010), p. 183.

[30] D. R. HARDOON, S. SZEDMAK, AND J. SHAWE-TAYLOR, *Canonical correlation analysis: An overview with application to learning methods*, Neural computation, 16 (2004), pp. 2639–2664.

[31] H. HOTELLING, *Relations between two sets of variates*, Biometrika, (1936), pp. 321–377.

[32] A. HUNT, M. WANDERLEY, AND R. KIRK, *Towards a model for instrumental mapping in expert musical interaction*, in Proceedings of the 2000 International Computer Music Conference, 2000, pp. 209–212.

[33] O. C. JENKINS AND M. J. MATARIĆ, *Performance-derived behavior vocabularies: Data-driven acquisition of skills from motion*, International Journal of Humanoid Robotics, 1 (2004), pp. 237–288.

[34] H. KADONE AND Y. NAKAMURA, *Segmentation, memorization, recognition and abstraction of humanoid motions based on correlations and associative memory*, in Humanoid Robots, 2006 6th IEEE-RAS International Conference on, IEEE, 2006, pp. 1–6.

[35] N. KOENIG AND M. J. MATARIC, *Behavior-based segmentation of demonstrated tasks*, in Proceedings of the International Conference on Development and Learning, Citeseer, 2006.

[36] J. KOHLMORGEN AND S. LEMM, *A dynamic hmm for on-line segmentation of sequential data*, in Advances in neural information processing systems, 2001, pp. 793–800.

[37] P. KOLESNIK AND M. M. WANDERLEY, *Implementation of the discrete hidden markov model in max/msp environment.*, in FLAIRS Conference, 2005, pp. 68–73.

[38] D. KULIC, C. OTT, D. LEE, J. ISHIKAWA, AND Y. NAKAMURA, *Incremental learning of full body motion primitives and their sequencing through human motion observation*, The International Journal of Robotics Research, 31 (2012), pp. 330–345.

[39] D. KULIĆ, W. TAKANO, AND Y. NAKAMURA, *Combining automated on-line segmentation and incremental clustering for whole body motions*, Proceedings - IEEE International Conference on Robotics and Automation, (2008), pp. 2591–2598.

[40] C. A. KURBY AND J. M. ZACKS, *Segmentation in the perception and memory of events*, Trends in cognitive sciences, 12 (2008), pp. 72–79.

[41] M. LEE, A. FREED, AND D. WESSEL, *Neural networks for simultaneous classification and parameter estimation in musical instrument control*, in Aerospace Sensing, International Society for Optics and Photonics, 1992, pp. 244–255.

[42] J. LIEBERMAN AND C. BREAZEAL, *Improvements on action parsing and action interpolation for learning through demonstration*, in Humanoid Robots, 2004 4th IEEE/RAS International Conference on, vol. 1, IEEE, 2004, pp. 342–365.

[43] J. F.-S. LIN AND D. KULIC, *Online segmentation of human motion for automated rehabilitation exercise analysis*, Neural Systems and Rehabilitation Engineering, IEEE Transactions on, 22 (2014), pp. 168–180.

[44] D. MERRILL AND J. A. PARADISO, *Personalization, expressivity, and learnability of an implicit mapping strategy for physical interfaces*, in Proceedings of the CHI Conference on Human Factors in Computing Systems, Extended Abstracts, 2005, pp. 2152–2161.

[45] E. R. MIRANDA AND M. M. WANDERLEY, *New digital musical instruments: control and interaction beyond the keyboard*, vol. 21, AR Editions, Inc., 2006.

[46] P. MODLER, *Neural networks for mapping hand gestures to sound synthesis parameters*, Trends in Gestural Control of Music, 18 (2000).

[47] A. MOMENI AND C. HENRY, *Dynamic independent mapping layers for concurrent control of audio and video synthesis*, Computer Music Journal, 30 (2006), pp. 49–66.

[48] J. A. PARADISO, *The brain opera technology: New instruments and gestural sensors for musical interaction and performance*, Journal of New Music Research, 28 (1999), pp. 130–149.

[49] M. POMPLUN AND M. J. MATARIC, *Evaluation metrics and results of human arm movement imitation*, in Proceedings, First IEEE-RAS International Conference on Humanoid Robotics (Humanoids, 2000.

[50] L. RABINER AND B.-H. JUANG, *Fundamentals of speech recognition*, (1993).

[51] L. R. RABINER, *A tutorial on hidden markov models and selected applications in speech recognition*, Proceedings of the IEEE, 77 (1989), pp. 257–286.

[52] N. RASAMIMANANA, F. BEVILACQUA, N. SCHNELL, F. GUÉDY, C. MAESTRACCI, E. FLÉTY, B. ZAMBORLIN, U. PETREVSKY, AND J.-L. FRECHIN, *Modular musical objects towards embodied control of digital music*, in Tangible Embedded and Embodied Interaction, 2011.

[53] D. ROCCHESSO, S. SERAFIN, F. BEHRENDT, N. BERNARDINI, R. BRESIN, G. ECKEL, K. FRANINOVIC, T. HERMANN, S. PAULETTO, P. SUSINI, ET AL., *Sonic interaction design: sound, information and experience*, in CHI'08 Extended Abstracts on Human Factors in Computing Systems, ACM, 2008, pp. 3969–3972.

[54] J. B. ROVAN, M. M. WANDERLEY, S. DUBNOV, AND P. DEPALLE, *Instrumental gestural mapping strategies as expressivity determinants in computer music performance*, in Kansei, The Technology of Emotion. Proceedings of the AIMI International Workshop, Citeseer, 1997, pp. 68–73.

[55] D. VAN NORT, M. M. WANDERLEY, AND P. DEPALLE, *On the choice of mappings based on geometric properties*, in Proceedings of the 2004 conference on New interfaces for musical expression, National University of Singapore, 2004, pp. 87–91.

[56] M. M. WANDERLEY, *Mapping strategies in real-time computer music*, Organised Sound, 7 (2002), pp. 83–84.

[57] M. M. WANDERLEY AND P. DEPALLE, *Gestural control of sound synthesis*, Proceedings of the IEEE, 92 (2004), pp. 632–644.