Human Action Recognition from Active Acoustics: Physics Modelling for Representation Learning and Inference Using Generative Probabilistic Graphical Models

by

Thomas S. Murray

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

February, 2015

Abstract

This dissertation explores computational methods to address the problem of physics-based modeling and ultimately doing inference from data in multiple modalities where there exists large amounts of low dimensional data complementary to a much smaller set of high dimensional data. In this instance the low dimensional time-series data are active acoustics from a micro-Doppler sensor that include no or very limited spatial information, and the high dimensional data is RGB-Depth skeleton data from a Microsoft Kinect sensor. The task is that of human action recognition from the active acoustic data. To accomplish this, statistical models, trained simultaneously on both the micro-Doppler modulations induced by human actions and symbolic representations of skeletal poses, are developed. This enables the model to learn correlations between the rich temporal structure of the micro-Doppler modulations and the high-dimensional motion sequences of human action. During runtime, the model then relies purely on the active acoustic data to infer the human action. In order to adapt this methodology to situations not observed in the training data, a physical model of the human body is combined with a physics-based simulation of the Doppler phenomenon to predict the acoustic data for a sequence of skeletal poses and a configurable sensor geometry. The physics model is then combined with a generative statistical model for human actions to create a generative physics-based model of micro-Doppler modulations for human action.

Primary Reader: Andreas G. Andreou

Secondary Reader: Ralph Etienne-Cummings

Committee Members: Mounya Elhilali, Philippe O. Pouliquen, Charbel G. Rizk

Acknowledgments

The work presented in this dissertation could never have happened in a vacuum. During my time at Johns Hopkins, I have learned a great deal from many people who have been generous enough to share their knowledge and time with me.

I want to thank my advisor, Andreas Andreou, for being an insightful and enthusiastic mentor throughout my graduate career. He has challenged me to think about important scientific questions in new ways. He has taught me to look across disciplines for inspiration and insightful connections. Thanks to him, my life as a researcher has been rarely dull and I only hope that I can emulate the devotion to interesting scientific research and mentorship that he has always shown his students.

A special thanks to my secondary reader, Ralph Etienne-Cummings, whose door has always been open, and conveniently located across the hall, during my time at Johns Hopkins. His guidance has been invaluable.

I want to thank the members of my committee, Mounya Elhilali, Philippe Pouliquen and Charbel Rizk, for their support throughout the dissertation process.

I want to thank Garrett Jenkinson for many thoughtful discussions during our

time together at Johns Hopkins, as well as for his comments on this manuscript.

I want to thank Andrew Cassidy, Joseph Lin, Recep Ozgun, Daniel Mendat, Tomas Figliolia, Sean McVeigh, Kayode Sanni, Gaspar Tognetti, Guillaume Garreau and Kate Fischl for the many engaging and insightful discussions, technical and otherwise, that have helped me to develop as both an engineer and a researcher during my tenure in the Andreou lab.

Daniel Mendat and Philippe Pouliquen have both been instrumental in various aspects of the research presented in this dissertation. I am grateful to Mike Carlin, Tomas Figliolia, Guillaume Garreau, Daniel Mendat, Jamal Molin, Kayode Sanni, Gaspar Tognetti and Jack Zhang, who all gave their time and energy as actors in the data collection experiments central to this dissertation.

My research was funded by an ONR MURI N000141010278, by NSF grant INSPIRE 1248056 and by NSF grant SCH-INT 1344772.

On a more personal note, I want to thank my family for their support. My parents, Marianne Simmons and Harry Murray, taught me to value knowledge and to think for myself. They have provided me with every opportunity to develop into an intelligent human being, of which they are fine examples themselves. My brother, Colin, has offered his own brand of wisdom throughout my life and is the quintessential calming influence. My uncle, John Murray, has always taken the time to give me thoughtful technical and professional advice. My grandparents, Helen and Richard Simmons, and Elizabeth and Harry Murray, instilled in me the importance of family and building a better world for future generations.

Finally, I want to thank Reina Chano for being a caring and supportive partner throughout this process. During the crunch, she kept me fed.

In the words of Douglas Adams,“So long and thanks for all the fish.”

Dedication

For my parents, Marianne Simmons and Harry Murray. Yes Mom, your progeny did, in fact, write this.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Human actions range from simple motions, such as a hand wave, to complex sequences composed of many intermediate actions, such as figure skating. Every day each of us performs many actions, even creating new actions to accomplish a novel task. Moreover, we are able to recognize and interact with other people because we can interpret their actions. Our brains enable all of this functionality, and they are unparalleled in their ability to process the world around us. By incorporating information from auditory, visual, olfactory and other sensory systems, humans are able to reliably identify the objects and actions they encounter. As engineers, studying the biological architecture and algorithms that have evolved in the brain to allow for these capabilities directly enhances our ability to design better machines capable of accomplishing tasks similar to those humans perform. As scientists, studying the challenges associated with sensing complex natural scenes and modeling interactions provides insights that are critical to understanding how biological organisms achieve these tasks so elegantly. While we allow biology to inspire our engineering, we are eager to develop a deeper understanding of biology through the process of engineering new systems.

Human actions occur in three-dimensional space and evolve over time. Most modern action recognition systems are based on visual data. Single RGB images capture a two-dimensional projection of the spatial arrangement of the human body in a scene. RGB video sequences capture the temporal evolution of those two-dimensional projections. Even more complete information can be gathered using RGB-Depth videos that can provide the temporal evolution of a human body in three dimensions.

However, it is also possible to sense motion by analyzing scattered acoustic waveforms. The micro-Doppler effect is a physical phenomenon that explains how moving objects induce frequency modulations when they scatter waves. The magnitudes of the modulations are proportional to the radial velocities of the objects scattering the waves. This mechanism provides an opportunity to infer information about motion from acoustic waves emitted by continuous active sonar sensors. Unfortunately, the low-dimensional time-series recorded by these sensors provides little information about the spatial arrangement of any moving objects and is therefore an incomplete representation of the total movement in a scene. Nonetheless, it is a scientifically interesting problem to investigate how much can be inferred about human actions from impoverished, low-dimensional acoustic data, particularly if physical models are used to constrain the search space.

Although the space of possible human motions is quite large, there are a lot of constraints placed on actions by the physical limitations and structure of the human body. In theory, a model that captures the precise physical constraints of human joints and dimensions could be used to bias the decisions of an action recognizer that operates on impoverished acoustic signals. This approach leverages prior knowledge about the task and models the physics of the environment.

Additionally, the physics behind the Doppler effect are well understood. By incorporating prior knowledge about the interactions between the sensor and the environment, models can be developed that account for the interaction between the environment and the acoustics to extract as much information as possible from the data recorded by a given sensor. They can also take advantage of the geometry of the sensor array in the environment to combine information from multiple sensors.

Figure 1.1 shows the ideal relationship the variables in the generative probabilistic model that incorporates the Doppler-physics and the structure of the human body. Essentially, the action class label Z_n defines the distribution of the skeletal poses Y_t. The joint positions X_t are defined by these poses and the properties of individual body segments are computed from the joint positions. Once the motion of the human has been defined, the Doppler modulations due to the human model can be determined deterministically and a noise model accounts for artifacts in the modulations that are either not captured by the human model or due to other moving objects in the

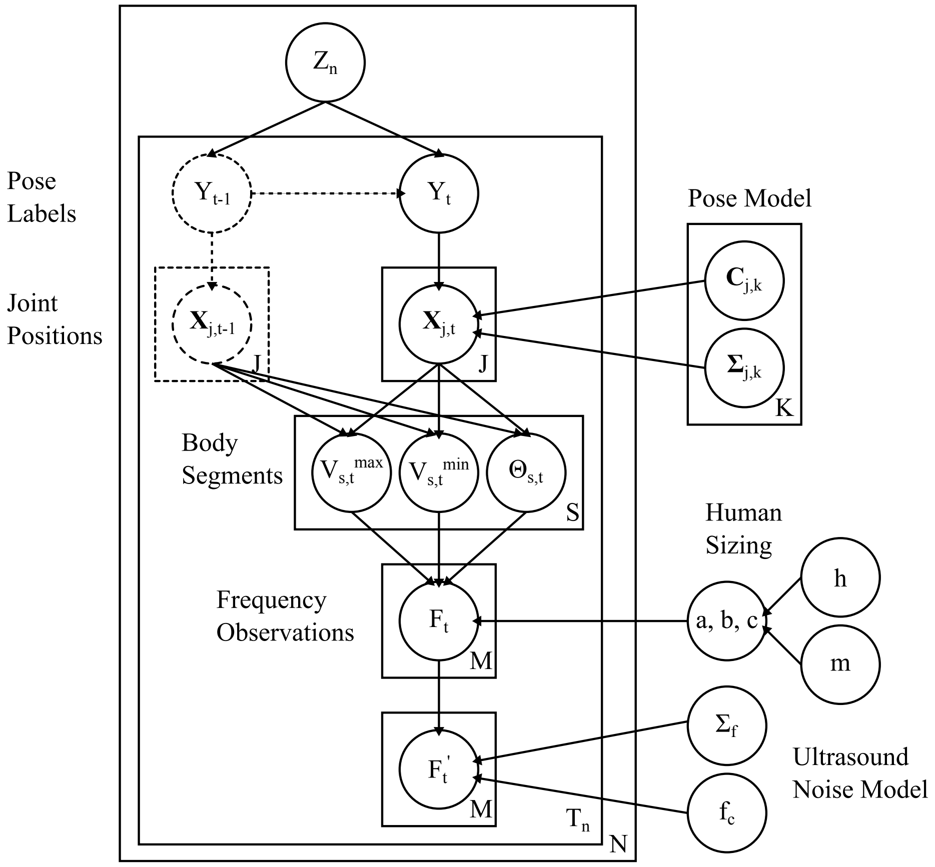


Figure 1.1: Graphical model illustrating the relationship between a moving human and acoustic micro-Doppler modulations.

environment. There are several global parameters used to tune the human model and adapt it to particular individuals.

Unfortunately, performing inference in this model is difficult for two reasons. First of all, computing the inverse to transform the frequency modulations back into joint positions is an ill-posed problem with multiple solutions. Also, there is no efficient closed form solution. Secondly, the temporal slices are not conditionally independent of each other, which prevents the use of inference techniques that decompose the computations efficiently.

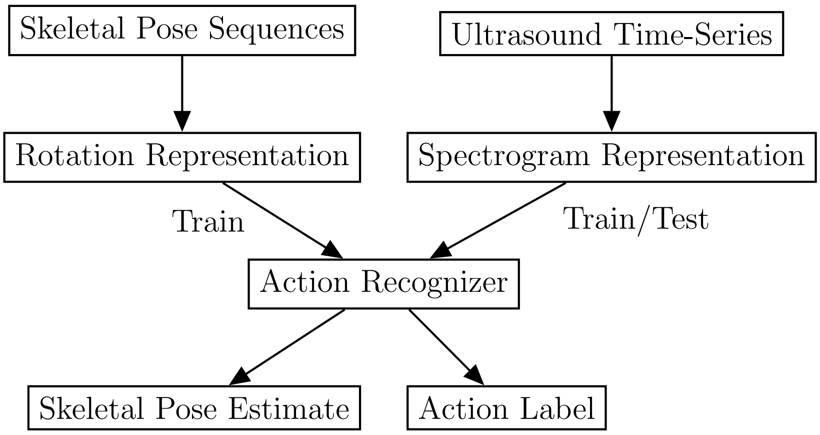


Figure 1.2: Schematic of the proposed data flow in the action recognizer.

Therefore, several simpler models that each specialize in a particular aspect of relating human action and active acoustics are developed. Figure 1.2 illustrates an approach to acoustic action recognition that builds on the hidden Markov model framework developed for traditional statistical speech recognition. This approach utilizes a simpler graphical model to capture the temporal sequences of skeletal poses and acoustic modulations and allows for the use of efficient inference algorithms. Unfortunately, integrating the Doppler physics into the model is still a challenge and the model instead learns purely statistical correlations between the data modalities.

Figure 1.3 illustrates an approach to incorporate physical models for both the environment and the interactions between the environment and the sensor with a generative probabilistic model to predict the acoustic modulations generated by a particular environment configuration. This framework focuses on a more realistic prediction of the acoustics and generating novel action sequences.

Given these challenges, is it possible to reliably recognize actions when only the

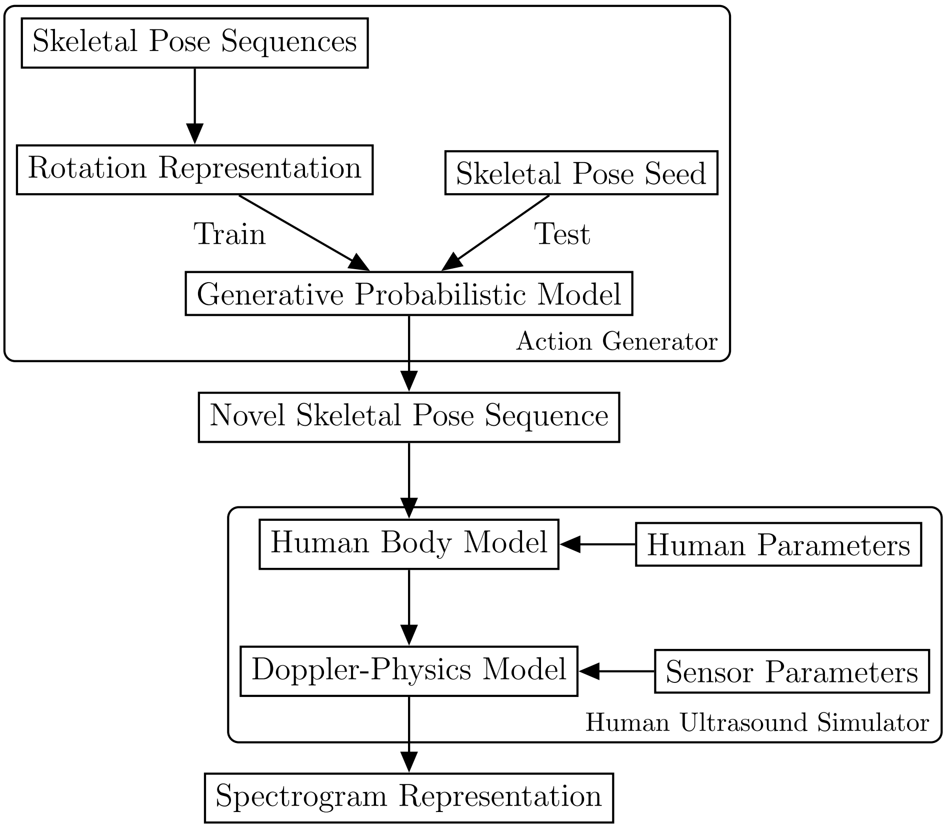


Figure 1.3: Schematic of the proposed data flow in the human ultrasound simulator.

micro-Doppler modulations are observed? How fine-grained of an action can be recognized? Can knowledge of the human body and its physical constraints, along with the physics behind the Doppler phenomenon, be used to bootstrap the impoverished active acoustic data and leveraged to construct models capable of inferring actions accurately? This scientific question is at the core of the modeling approach taken throughout this work. The models are uniquely tailored to leverage both the Kinect skeletal data and multiple active acoustic sensors.

The dissertation is organized as follows. Chapter 2 presents the underlying physics behind the Doppler effect. This phenomenon is the physical mechanism by which information about the velocity of moving objects can be inferred from the frequency modulations in scattered acoustic signals. The micro-Doppler effect is an extension of the Doppler effect for objects that have multiple components that each move with separate velocities.

Chapter 3 describes the properties of the active acoustic sensors, the Microsoft Kinect sensor and the data acquisition system used to continuously capture synchronized data from multiple sensors. The data acquisition system includes a frequency modulated beacon that allows sensors to be distributed across a wide area and connected to separate data collection computers. As long as the sensors are within range of the synchronization pulses emitted by the beacon, the data collected from the sensors is augmented with embedded timestamps that preserve the temporal coherence of the independent data streams. This allows the use of three active sonar sensors, operating in non-overlapping frequency bandwidths, to observe the same scene from multiple angles without interfering with each other.

Chapter 4 presents the Johns Hopkins University multimodal action (JHUMMA) dataset. The JHUMMA dataset is a collection of thirteen independent trials of actors performing twenty-one classes of actions. The dataset includes the recordings from three independent active sonar sensors and the RGB-Depth imagery recorded by the Kinect sensor. The actions recorded in the JHUMMA dataset strongly indicate that the micro-Doppler modulation patterns are an intriguing modality for sensing action. When a sonar unit is stationary, the Doppler phenomenon is uniquely selective for sensing motion in the environment. On the flip side, there is very little information about the spatial arrangement of moving objects present in the modulation patterns. In fact, the modulations in a single ultrasound time-series don't even contain complete information about the velocities of moving objects in the scene. Only the radial velocity components induce a Doppler shift.

Chapter 5 leverages machinery developed in the statistical speech recognition community and applies it to the task of recognizing the actions in the JHUMMA dataset from only the active ultrasound observations. Hidden Markov models (HMMs) are used to capture the sequential nature of the data and are trained such that the series of skeletal poses derived from the Kinect sensor form the latent states and the Doppler modulations recorded by the active ultrasound sensors form the visible states. The trained HMMs are then used to identify the most likely human actions when only the acoustic Doppler modulations of unseen sequences are observed.

Chapter 6 develops a physics based forward model that leverages the skeletal representation tracked by the Kinect to predict the Doppler modulations induced by human actions and observed by an active ultrasound sensor at any location. This physics simulation builds upon a simple sound scattering model of the human body.

Chapter 7 develops a generative conditional deep belief network (CDBN) capable of hallucinating the micro-Doppler modulations induced by novel human actions. The CDBN can be trained on either the acoustics themselves or the skeletal poses tracked by the Kinect. Combining the generative model for the skeleton with the Doppler-physics model developed in the previous chapter results in a configurable model that can be adapted to produce acoustic modulations for previously unobserved sensor configurations.

Chapter 2

The Micro-Doppler Effect

The velocity of an object moving in the direction of an observer can be estimated by measuring the frequency shift of any wave radiated or scattered by the object. This frequency shift is known as the Doppler effect. If the moving object has additional moving components, each moving part will result in a modulation of the base Doppler frequency shift. The scattered wave will then exhibit a spectrum with multiple frequency modulations, as the frequency shifts induced by the individual movements are superimposed. This frequency modulation is known as the micro-Doppler effect.

Throwing a rock into a relatively calm pond causes waves to travel radially out from the point of impact. The rock does not move in the water, simply sinking to the bottom, so the wavefronts are regularly spaced. A duck swimming in a pond also creates waves; however, they are not regularly spaced. This is illustrated in Figure 2.1. If you observe the duck swimming towards you, then the wavefronts ahead of the duck are squished together and will reach the shore with higher frequency. If you observe the duck swimming away from you, then the wavefronts behind the duck are spread out and will reach the shore with lower frequency. The observed shift in frequency due to the direction of motion is an example of the Doppler effect. The observed shift in frequency due to the motion of several ducks swimming together is an example of the micro-Doppler effect.

Figure 2.1: A swimming "Doppler"-Duck generating waves on the water. [a]

The Doppler phenomenon originated in the field of astronomy. It was first proposed in 1842, by Christian Doppler,[1] to explain the color of binary stars. Two

[a] Source: Daniel R. Mendat, March 21, 2009.

years later, the Doppler effect was demonstrated experimentally for acoustic waves by C.H.D. Buys Ballot.[2] He was able to detect the frequency shift of musicians playing a constant tone on a train as it passed through a station.

2.1 The Doppler Effect

Formally, the Doppler effect involves a source, which emits or scatters a wave, and an observer, which detects the wave. Although the Doppler effect is relevant to any wave phenomenon, I focus on developing machinery for acoustic waves. It is assumed that the scattering object is a simple point mass and that the wave emitted by the source is a constant tone, with a wavelength λ_{source} and frequency f_{source} related by

$$\lambda_{\text{source}} = \frac{c_s}{f_{\text{source}}}, \tag{2.1}$$

where c_s is the speed of sound.

A Doppler shift occurs when either the source or observer are moving. However, the Doppler shift contributed by the motion of the source and the motion of the observer is not symmetric. All motion is measured relative to the medium that the wave travels through. In this work, the medium is air.

First consider the case where the source and receiver are stationary. By definition, the distance between successive wavefronts is λ_{source} and a period of $\frac{1}{f_{\text{source}}}$ seconds elapses between their arrivals. In this case, the frequency is unchanged. Figure 2.2

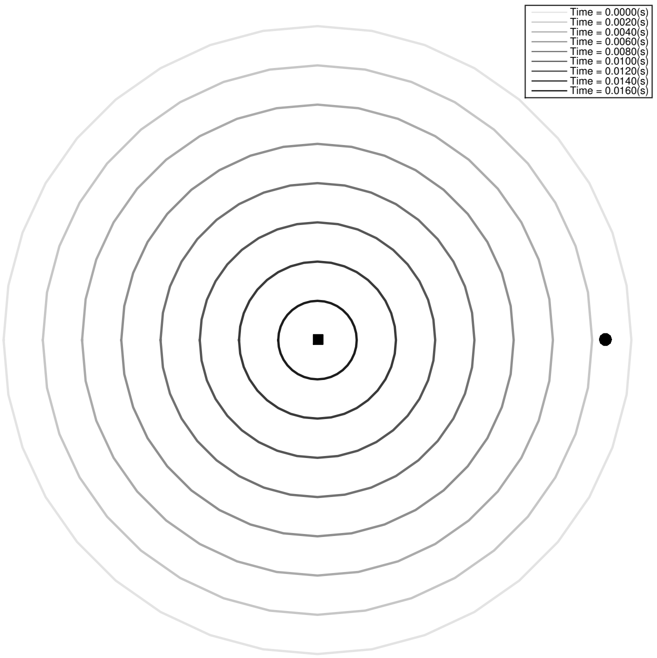


Figure 2.2: Simulation showing the wavelength of an acoustic wave when neither the source or the observer are moving.

simulates the acoustic wavefronts generated by the source where both the source and the observer are stationary. The source is the square and the observer is the circle. In this simulation, the transmitted frequency is $f_{\text{source}} = 500\text{Hz}$, the speed of sound is $c_s = 340\text{m/s}$ and the distance between the source and observer is 5m. The wavefronts are grayscale-coded based on the time that they were emitted.

Now consider the case where a stationary source emits a wave and the observer is moving with a velocity v_{observer}. Note that v_{observer} is only the radial component of the observer's velocity relative to the source. Figure 2.3 simulates the acoustic wavefronts generated by the source when the source is stationary and the observer is moving towards the source at $v_{\text{observer}} = \frac{c_s}{2}$. The source is the square and the observer

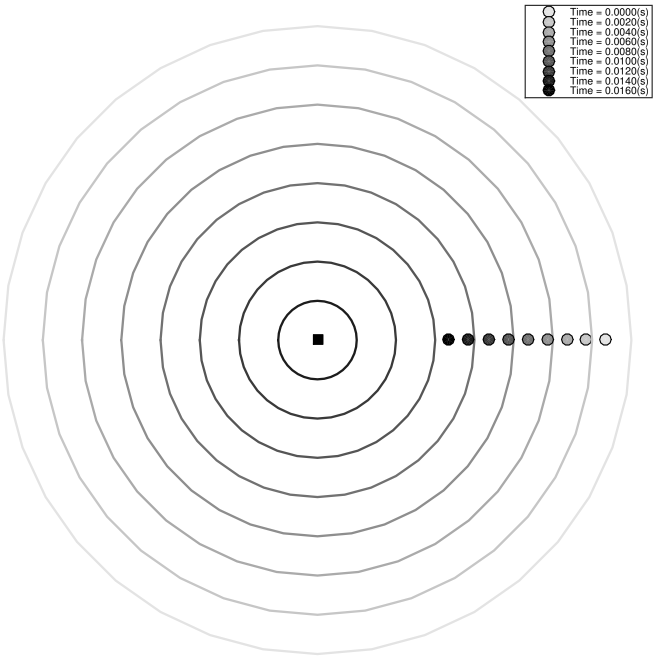


Figure 2.3: Simulation showing the relationship between the source wavelength and the observed wavelength of an acoustic wave when the source is stationary and the observer is moving.

is the circle. In this simulation, the transmitted frequency is $f_{\text{source}} = 500\text{Hz}$, the speed of sound is $c_s = 340\text{m/s}$ and the initial distance between the source and observer is 5m. The wavefronts are grayscale-coded based on the time that they were emitted.

The distance between consecutive wavefronts is no longer λ_{source} because v_{observer} moves the observer closer to or farther from the source. The second wavefront will have to travel a shorter or longer distance, respectively, before it arrives at the observer. This results in the apparent frequency shift of the observed signal. The distance between the wavefronts is

$$\lambda_{\text{observer}} = \lambda_{\text{source}} - \frac{v_{\text{observer}}}{f_{\text{observer}}}.$$

By definition, $\lambda_{\text{source}} = \frac{c_s}{f_{\text{source}}}$ and $\lambda_{\text{observer}} = \frac{c_s}{f_{\text{observer}}}$. Making these substitutions for the wavelengths yields,

$$\frac{c_s}{f_{\text{observer}}} = \frac{c_s}{f_{\text{source}}} - \frac{v_{\text{observer}}}{f_{\text{observer}}}.$$

Solving for the observed frequency,

$$f_{\text{observer}} = \frac{c_s + v_{\text{observer}}}{c_s} f_{\text{source}}, \tag{2.2}$$

results in an expression for the Doppler shift given a stationary source and moving observer. By convention, the velocity of the observer is considered positive if the observer is moving towards the source and negative if the observer is moving away from the source, relative to the medium.

Finally consider the case where the source is moving while it emits a wave and the observer is stationary. Figure 2.4 simulates the acoustic wavefronts generated by the source when the source is moving toward the observer at $v_{\text{source}} = \frac{c_s}{2}$ and the observer is stationary. The source is the square and the observer is the circle. In this simulation, the transmitted frequency is $f_{\text{source}} = 500$Hz, the speed of sound is $c_s = 340$m/s and the initial distance between the source and observer is 5m. The

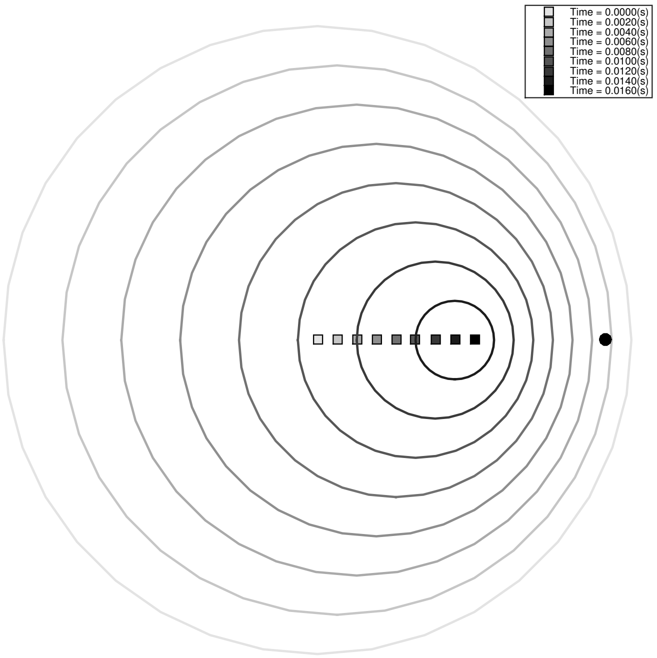


Figure 2.4: Simulation showing the relationship between the source wavelength and the observed wavelength of an acoustic wave when the source is moving and the observer is stationary.

wavefronts are grayscale-coded based on the time that they were emitted.

Let v_{source} be the radial component of the source's velocity relative to the observer. When a source is moving, the wavefronts will be more crowded together in front of the source and more spread out behind it. This occurs because after a wavefront is emitted, the source moves during the duration of the wave period before the next wavefront is emitted. The distance between two consecutive wavefronts is now,

$$\lambda_{\text{observer}} = \lambda_{\text{source}} - \frac{v_{\text{source}}}{f_{\text{source}}}.$$

Again, substituting for the wavelengths yields the following expression in terms of the source and observed frequencies,

$$\frac{c_s}{f_{\text{observer}}} = \frac{c_s}{f_{\text{source}}} - \frac{v_{\text{source}}}{f_{\text{source}}} = \frac{c_s - v_{\text{source}}}{f_{\text{source}}}.$$

Solving for the observed frequency,

$$f_{\text{observer}} = \frac{c_s}{c_s - v_{\text{source}}} f_{\text{source}}. \tag{2.3}$$

results in an expression for the Doppler shift given a moving source and stationary observer. By convention, the velocity of the source is considered positive if the source is moving towards the observer and negative if the observer is away from the observer, relative to the medium.

If both the source and the observer are in motion, then the effects are compounded. The observed frequency is shifted by,

$$f_{\text{observer}} = \frac{c_s + v_{\text{observer}}}{c_s - v_{\text{source}}} f_{\text{source}} = \frac{1 + \frac{v_{\text{observer}}}{c_s}}{1 - \frac{v_{\text{source}}}{c_s}} f_{\text{source}}. \tag{2.4}$$

The same sign conventions that applied to the source and observer velocities apply in this case also. For a more detailed development of the Doppler effect, see Chen[3,4] and Murray.[5]

2.2 The Doppler Effect for Monostatic Active Sonar

Active sonar is one type of sensor capable of exploiting the Doppler effect to measure the velocity of a target. By emitting a waveform that is Doppler-sensitive, such as a pure tone, the sonar can record the scattered waveforms and the difference between the transmitted and received frequencies can be measured. The radial velocity component of the target object that scattered the wave can then be estimated from any frequency difference between the transmitted and received signal.

In a monostatic sonar setup, the transmitter and receiver are co-located at the same coordinates. It is assumed that the target is a moving point mass and that the sonar equipment is stationary. Under these conditions, the Doppler phenomenon actually occurs twice; once as the transmitted signal arrives at the target and again as the target scatters the signal back to the receiver.

The first Doppler shift involves a stationary source, the sonar transmitter, and a moving observer, the target. Therefore, using Equation 2.2 yields,

$$f_{\text{target}} = \left(1 + \frac{v_{\text{target}}}{c_s}\right) f_{\text{transmitter}}. \tag{2.5}$$

The second Doppler shift involves a moving source, the target and a stationary observer, the sonar receiver. The final Doppler shift observed by the receiver is found by applying the first Doppler shift to Equation 2.3. This yields,

$$\begin{aligned} f_{\text{receiver}} &= \left(\frac{1}{1-\frac{v_{\text{target}}}{c_s}}\right) f_{\text{target}} \\ &= \left(\frac{1}{1-\frac{v_{\text{target}}}{c_s}}\right)\left(1+\frac{v_{\text{target}}}{c_s}\right) f_{\text{transmitter}} \\ &= \left(\frac{1+\frac{v_{\text{target}}}{c_s}}{1-\frac{v_{\text{target}}}{c_s}}\right) f_{\text{transmitter}}. \end{aligned} \tag{2.6}$$

This expression can be further simplified if it is assumed that the speed of sound is much faster than the speed of the source. For most natural environments, this is a very reasonable assumption as few things move at the speed of sound. The Taylor series for a function $f(x)$ evaluated at a point a is

$$f(x) \approx \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots . \tag{2.7}$$

For $f(x) = \frac{1+x}{1-x}$, evaluated at $a = 0$, the Taylor series is,

$$\frac{1+x}{1-x} \approx 1 + 2x + \frac{3}{2}x^2 + \cdots \tag{2.8}$$

Setting $x = \frac{v_{\text{source}}}{c_s}$, we can approximate the Doppler shift for the monostatic sonar as,

$$f_{\text{receiver}} \approx \left(1 + 2\frac{v_{\text{target}}}{c_s}\right) f_{\text{transmitter}}. \tag{2.9}$$

If the speed of sound is much faster than the speed of the target, that is

$$\frac{v_{\text{target}}}{c_s} \ll 1, \tag{2.10}$$

then it is appropriate to approximate the function with the first order Taylor series, ignoring the higher order terms.

2.3 The Micro-Doppler Effect

The micro-Doppler effect is an extension of the Doppler effect and occurs when multiple moving objects scatter a wave. Each object contributes its own Doppler shift related to the object's radial velocity component with respect to the receiver. All of the scattered waves are additive, and the resulting micro-Doppler modulation is a superposition of the individual components. Assuming that there are N moving point masses in a scene where a pure tone with frequency f_c is transmitted, then the scattered signal seen by the receiver is

$$s_{\text{receiver}}(t) = \sum_{i=1}^{N} A_i(t) \cdot \sin(2\pi f_c t + 2\pi f_i t + \phi_i(t)). \tag{2.11}$$

Each point mass scatters the pure tone and modulates the frequency by $f_i = 2\frac{v_i}{c_s} f_c$. This is precisely the Doppler shift contributed by the motion of a single point mass developed for monostatic sonar in Equation 2.9. There is also a phase shift $\phi_i(t)$

that depends on the range of the point mass. The ability to resolve this phase shift depends on the range of the point mass and the wavelength of the scattered wave. The amplitude of each component, A_i, depends on the scattering surface and the range of the point scatterer.

When the number of scattering targets is large, distinguishing the contribution of each component to the overall micro-Doppler modulation is difficult. Some approaches use significant prior knowledge to better estimate each piece, but the problem remains a challenge, particularly when the scene is not well defined a priori.

2.4 Using Micro-Doppler Modulations to Sense Action in Natural Environments

While the Doppler effect is very specific to sensing motion, there are still many challenges associated with exploiting it to sense and identify actions. At a fundamental level, real actions are sequences of motion that evolve in time and three-dimensional space. However, the micro-Doppler modulations recorded by a single active sonar sensor are one-dimensional time-series. The modulations of a pure tone used to sense a complex moving scene do not capture much in the way of range or spatial information. Over a given time window, the frequency modulations provide a histogram of the velocities present in the scene. Fortunately, due to the physical limitations of most multi-component objects, such as the human body, the realizable set of actions is heavily constrained. In the case of a human, the scattering components are linked rigid bodies, which constrain the space of human action and induce distinctive temporal structure in the modulations across a sequence of consecutive windows. This is a much more structured situation than the arbitrary sum of point masses expressed in Equation 2.11. One of the primary scientific questions addressed by this work is whether having a symbolic model of the physical objects of interest is sufficient to bootstrap low-dimensional, impoverished sensory data in order to make inferences about a complex, higher-dimensional, natural environment.

The micro-Doppler sonar technology complements cameras and visual surveillance in situations where the mere presence of life is relevant (for security or search and rescue reasons, for example), since although it depends upon a clear line of sight between the detector and the object of interest it does not rely on visibility per se. Preliminary work on human and animal species classification as well as action and behavior recognition has yielded highly promising recognition rates.[3,6–26]

Another challenge is that many moving objects have symmetry in their motion. For example, a pendulum may swing from side to side and a human body may move its right or left arm. Distinguishing between these actions can be very challenging for a single sonar sensor, located at the line of symmetry, due to paucity of spatial information in the micro-Doppler modulations. One way to overcome this limitation is to use multiple sensor units arranged so that no single line of symmetry is common to all the sensors. In the next chapter, a novel data acquisition system, that is designed to integrate multiple acoustic sensors with very accurate temporal resolution, is presented. Leveraging this system allows for synchronized data collection with multiple sonar units that will help alleviate ambiguities due to spatial symmetry.

Chapter 3

Data Acquisition and Processing

From an algorithmic perspective, some of the challenges associated with leveraging the micro-Doppler, discussed in Section 2.4, can be mitigated by judiciously acquiring a rich set of multimodal data on which to train models. Specifically, the use of multiple ultrasound sensors, arranged to maximize the power of the acoustics to discriminate between ambiguous orientational symmetry, and the collection of a limited amount of higher-dimensional visual data to provide a spatial ground truth for the scene.

Of course, this places additional requirements on the system used to acquire the experimental data, which must now be able to accommodate multiple sensors of different types that are distributed throughout the sensing environment. Moreover, maintaing precise temporal synchronicity between the data points recorded from each sensor is important, so that the data from individual sensors can be aligned unambiguously in time. This chapter describes the sensors, hardware, data collection software and basic signal processing algorithms used to acquire and visualize the acoustic and visual data that is presented throughout this work.

3.1 Acoustic Data Acquisition System

The acoustic data acquisition system began development under the SCANDLE project and was further improved during the MURI project.[27,28] It consists of a set of data acquisition (DACQ) units, and a frequency modulated (FM) transmitter designed to synchronize the data collection process across all of the DACQ units. The data acquisition units can be configured to interface with a wide range of acoustic sensors including several types of passive microphones, microphone arrays, ultrasound sonar transducers and seismic sensors. The data acquisition units have integrated FM receivers that allow for synchronization among all units within range of the FM transmitter. Additionally, other types of sensors can access the synchronization signal if they are connected to the same computer as the FM transmitter.

3.1.1 Sampling Acoustic Signals

A single DACQ unit can be configured to operate with one to four channels depending on the requirements of the target sensor. The total bandwidth of each DACQ unit is 100kHz, which is divided equally among the number of channels. The sampling frequency for each channel configuration is enumerated in Table 3.1.

Table 3.1: DACQ sampling frequencies for each channel configuration.

Number of Channels	Channel Sampling Frequency (F_s)
1	100kHz
2	50kHz
3	33kHz
4	25kHz

In each channel, the analog input signal $x_v(t)$ is digitized by a dedicated Analog Devices AD7686 analog to digital converter (ADC) with 16-bit resolution. The output codes of this ADC range from 0 to $2^{16} - 1 = 65535$ and encode the input voltage range, which is approximately $-2.5V$ to $2.5V$. The conversion from ADC counts $x_c[n]$ back into a voltage sample $x_v[n]$, is determined by scaling the counts to match the input voltage range. Thus,

$$x_v[n] = (x_c[n] - 2^{15})\Big(\frac{5V}{2^{16}}\Big). \tag{3.1}$$

To relate the sampled sequence to the analog channel input, note that continuous time t is related to the discrete sample index n by

$$t = \frac{n}{F_s}, \tag{3.2}$$

where F_s is the channel sampling frequency. Therefore,

$$x_v(t) = x_v(\frac{n}{F_s}) = x_v[n]. \tag{3.3}$$

3.1.2 Time-Frequency Representation of Acoustic Signals

The frequency content of the acoustic signals changes over time. In order to capture these changes, the acoustic time series presented throughout this work are often processed to produce a spectrogram. A spectrogram is a time-frequency representation generated by analyzing the frequency content of a signal in a series of time windows. For the acoustic data recorded by the data acquisition system, a Hanning function is generally used to window the signal in the time domain and the frequency content of each window is analyzed using a discrete Fourier transform (DFT). When visualizing the data, the convention is to plot the magnitude of the power spectrum with the frequency content on the vertical axis and subsequent time windows on the horizontal axis. One limitation of this representation is that phase information is discarded.

3.1.3 Noise Characterization

The noise level for a channel in the data acquisition system was experimentally measured to be approximately 2.5mV by grounding the input of the channel and measuring the amplitude of the analog signal at the input pin of the channel ADC with an oscilloscope. This measurement was taken with the programmable channel gain set to 1, the minimum value. When digitized, 2.5mV corresponds to 32 ADC counts. The channel ADCs have a resolution of 16 bits, so the signal to noise ratio (SNR) for a channel in the data acquisition system is approximately 12 bits.

However, it is less straightforward to determine the power of the noise spectrum after transforming these signals into frequency and time-frequency representations. When analyzing signals in the frequency domain, phase information is typically discarded and only the log magnitude of the power spectrum in decibels (dB), referenced to a 1V peak-to-peak sinusoid, is considered. The analysis of the noise level is further complicated by quantization and spectral leakage from frequency binning effects introduced by the fast Fourier transform (FFT), MATLAB's implementation of the DFT. Moreover, when processing the time-domain signals to produce time-frequency spectrograms, the effects of the time-domain windowing in the processing chain further obfuscates how the 2.5mV noise specification is affected. Therefore, to understand what level of power can be attributed to noise, the output of the signal processing chain was analyzed for known, simulated, test signals. By setting the amplitude of the sinusoid to the voltage level of a specified noise floor, 2.5mV for the DACQ system, the power level of a signal that should be considered noise can be determined empirically.

In order for the processed result to be independent of the length of the FFT, an appropriate scaling factor must be applied. The DFT definition used by MATLAB is

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot \exp(-2\pi j k \frac{n}{N}), \tag{3.4}$$

where N is the length of the DFT and $0 \leq k \leq N$. By convention, the FFT of a sinusoid with a peak-to-peak amplitude of 1V should have a magnitude of 1V. There is no normalization term to account for the length of the DFT, so longer DFTs can result in higher amplitude frequency content. Additionally, the signal energy is split between positive and negative frequencies. To account for these two effects, the DFT $X[k]$ is scaled by a factor of $\frac{2}{N}$, making the amplitude independent of the length of the DFT and doubling the energy. This normalization term is used in all of the subsequent signal processing so the signal power of various voltage levels is directly comparable regardless of the length of the DFT used for that particular processing algorithm.

The test signal is a discrete-time ideal sinusoid,

$$x[n] = A \cdot \sin(2\pi f \frac{n}{F_s}), \tag{3.5}$$

with amplitude A and frequency f. The sampling frequency F_s is set to 100kHz, the maximum bandwidth of a single data acquisition channel. The frequency of the test signal was placed in the center of a frequency bin to minimize spectral leakage. Table 3.2 reports the signal power attributed to test signals over a range of amplitudes and for a single FFT versus a spectrogram analysis.

According to the convention described above, the 1.0V test signal is the reference for the dB scale and should ideally have a power of 0dB. In practice the maximum

Table 3.2: Noise power for simulated sinusoidal signals.

Amplitude	FFT Signal Power (dB)	Spectrogram Signal Power (dB)
2.5 V	7.9274	2.6062
1.0 V	-0.0314	-5.3526
2.5 mV	-52.0726	-57.3938
0.25 mV	-72.0726	-77.3938

magnitude of the scaled FFT is only 0.9964V and there is a small amount of spectral leakage exhibited in the frequency domain. This accounts for the −0.0314 dB discrepancy.

3.1.4 Data Acquisition System Baseline Noise

To verify that the DACQ units fall within the simulated noise power specifications, three of the DACQ units were configured as active ultrasound sensors and left to record in a room with no moving objects. Using this setup, there should only be significant signal power at the carrier frequency emitted by the unit and 0Hz, which corresponds to direct current (DC). The ultrasound configuration is described in more detail in Section 3.2. These DACQ sensor configurations were used for the data collection described in Chapter 4.

The frequency spectrum, computed over an 80 second window of data recorded by the 40kHz ultrasound unit, is shown in Figure 3.1. Although several harmonics contain enough power to reach the decibel level corresponding to a 0.25mV amplitude signal, only the power at the carrier frequency and DC is sufficient to reach the decibel level corresponding to a 2.5mV amplitude signal. All of the analog components that

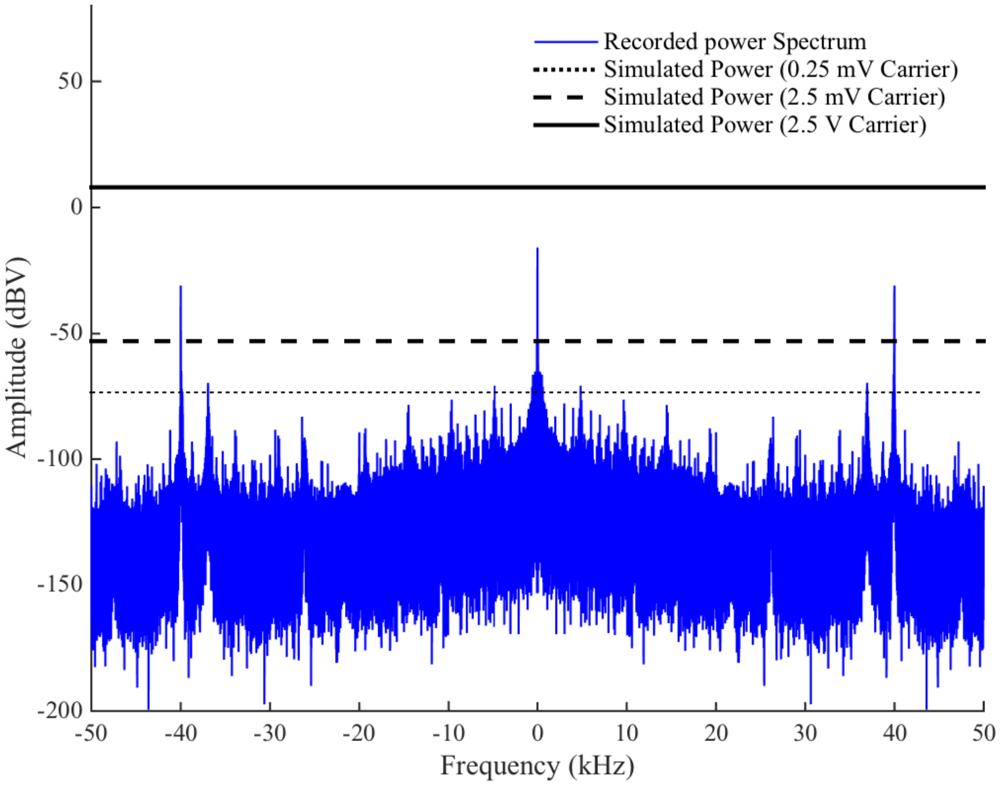


Figure 3.1: Measured noise power spectrum in the 40 kHz ultrasound unit.

are part of the data acquisition channels were selected to have noise specifications below 2.5mV.

The frequency spectrum, computed over an 80 second window of data recorded by the 33kHz ultrasound unit, is shown in Figure 3.2. The noise power spectrum responds similarly to the 40kHz case, although some of the harmonics are shifted.

The frequency spectrum, computed over an 80 second window of data recorded by the 25kHz ultrasound unit, is shown in Figure 3.3. Again, the noise power spectrum responds similarly to the 40kHz case, although some of the harmonics are shifted.

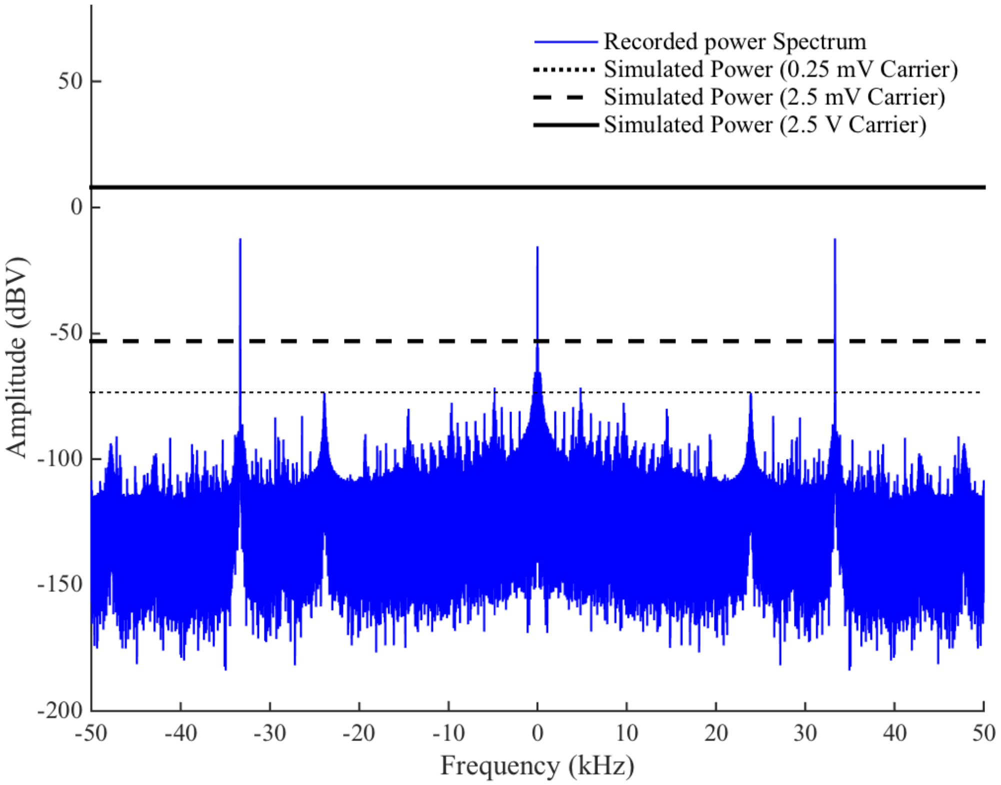


Figure 3.2: Measured noise power spectrum in the 33 kHz ultrasound unit.

The variation between DACQ units is due to mismatch in the analog components.

3.1.5 Setting the Channel Gain and Operating Mode

Each of the four channels in a data acquisition unit are equipped with a dedicated Texas Instruments PGA112 programmable gain amplifier (PGA). The PGA precedes the channel ADC in the analog processing chain, providing configurable gain control

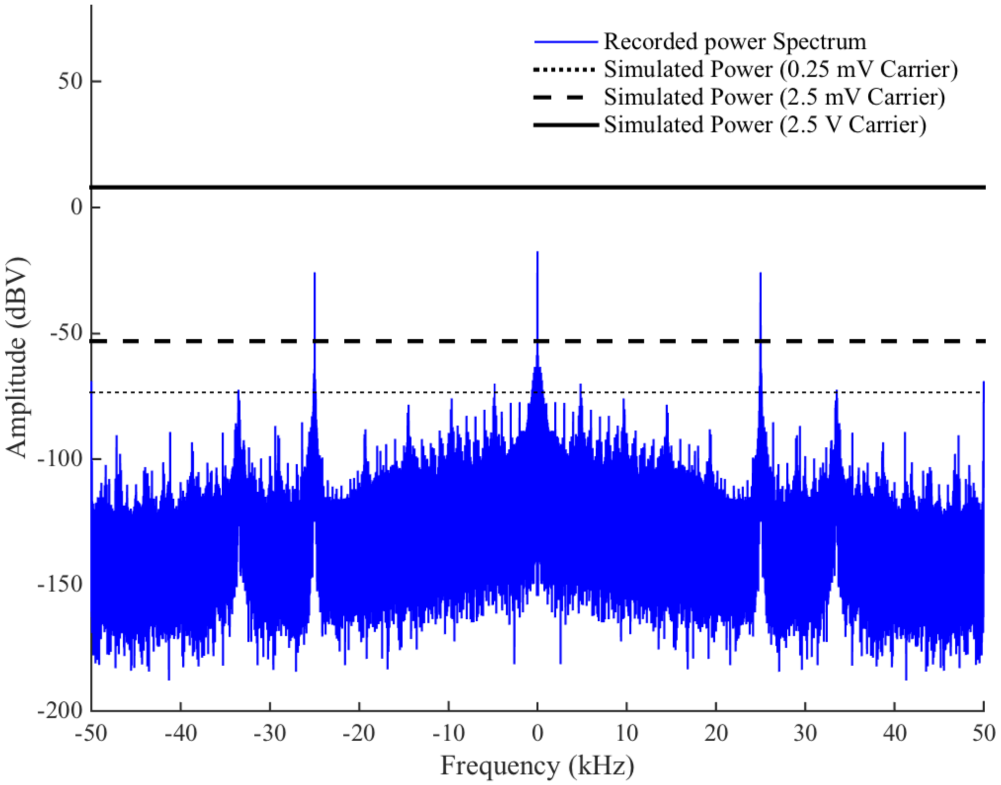


Figure 3.3: Measured noise power spectrum in the 25 kHz ultrasound unit.

for the analog signal as well as several diagnostic operating modes. These options are all configured by sending a control byte to the data acquisition unit. If the MSB is set to 0, then the following seven bits are interpreted to configure the PGA. If the MSB is set to 1 and the DACQ unit has been configured with a special debugging features, then the following seven bits are interpreted as several debugging protocols that are described in the next section. The upper nibble of the control byte encodes the gain setting, which can be any power of 2 ranging from 1 to 128. Table 3.3 enumerates the control nibble for each gain setting. Note that the MSB is always set to 0 in every nibble, so any gain setting is interpreted as a PGA configuration.

Table 3.3: Format of the control nibble for selecting different PGA gain multipliers.

Upper Configuration Nibble	Channel Gain
0000	1
0001	2
0010	4
0011	8
0100	16
0101	32
0100	64
0101	128

The lower nibble of the control byte encodes the operating mode. The PGA can be configured to output either the analog signal connected to its input or any of several constant voltage values derived from the power supply lines. Table 3.4 enumerates the control nibble for each mode setting. The constant input voltages provide a known input that can be used for diagnosing issues with the data acquisition process. They are also useful for investigating noise on the power supply.

Table 3.4: Format of the control nibble for selecting different PGA operating modes.

Lower Configuration Nibble	PGA Input Source
0000	5.0V (V_{dd})
0001	analog input (normal operation)
1100	0.0V (Gnd)
1101	4.5V ($\frac{9}{10} \times V_{dd}$)
1101	0.5V ($\frac{1}{10} \times V_{dd}$)
1101	2.5V ($\frac{1}{2} \times V_{dd}$)

The control byte for configuring the PGA can be programmed using the standalone `dacq_spga` application.

3.1.6 Setting Debugging Configurations

When the MSB of the control byte is set to 1, the following seven bits are interpreted as commands to effectively enable or disable several analog circuits on the data acquisition board. These circuits are each capable of coupling noise into the analog signal. These debugging configurations allow for the individual contributions from each sub-circuit to be analyzed. Table 3.5 enumerates the sub-circuit enabled by setting a particular debugging bit to 1. The MSB enables the debugging functionality. The two LSBs operate as a channel select, allowing the dedicated ADCs on each channel to be controlled independently.

Table 3.5: Format of the control byte for selecting different debugging modes.

Configuration Bit Index	Debug Function
7	Debug Select
6	Phase-Locked Loop (PLL) On
5	FM Local Oscillator (LO) On
4	FM ADC On
3	Output DAC On
2	Channel ADC On
1-0	Channel Select

The control byte for the debugging circuit configurations can be programmed using the stand-alone `dacq_sdbg` application.

3.1.7 Data Acquisition System Frame Format

Every frame is composed of a header, which has twelve 16-bit words, followed by 960 samples, which are 16-bits each. The overall sampling rate of a data acquisition unit is 100kHz, so each frame corresponds to 9.6 milliseconds of data. Table 3.6 enumerates the contents of each word in the header.

Table 3.6: Format for the header information in a DACQ frame.

<table>
<tr><th>Word Index</th><th colspan="2">Upper Byte</th><th>Lower Byte</th></tr>
<tr><td>0</td><td colspan="2">Synchronization, Byte 1</td><td>Synchronization, Byte 0</td></tr>
<tr><td>1</td><td colspan="2">Synchronization, Byte 3</td><td>Synchronization, Byte 1</td></tr>
<tr><td>2</td><td colspan="2">Module ID, Byte 2</td><td>Module ID, Byte 0</td></tr>
<tr><td>3</td><td colspan="2">Module ID, Byte 3</td><td>Module ID, Byte 1</td></tr>
<tr><td>4</td><td colspan="3">Frame Index</td></tr>
<tr><td>5</td><td>1</td><td colspan="2">Time-stamp, Bits 14 to 0</td></tr>
<tr><td>6</td><td>1</td><td colspan="2">Time-stamp, Bits 29 to 15</td></tr>
<tr><td>7</td><td>1</td><td colspan="2">Time-stamp, Bits 44 to 30</td></tr>
<tr><td>8</td><td>1</td><td colspan="2">Time-stamp, Bits 59 to 45</td></tr>
<tr><td>9</td><td colspan="2">PGA Gain/Mode</td><td>RSSI</td></tr>
<tr><td>10</td><td colspan="3">Minimum Value</td></tr>
<tr><td>11</td><td colspan="3">Maximum Value</td></tr>
</table>

The frame synchronization pattern is the four byte sequence $0, 0, 0, 255$. When observing the data stream, this pattern can be used to identify the beginning of a frame. All of the other fields in the frame have been modified where necessary to prevent this pattern from occurring elsewhere.

The module identification is a four character code that describes the configuration of the data acquisition unit used to collect the frame. For example, a unit configured as an ultrasound sensor with a transmitted frequency of 40kHz has a module identification code of US40. All of the module identification codes have been chosen so that they do not allow an erroneous synchronization pattern to appear in the data stream.

The frame index is an integer from 1 to 65535 that is used to mark each frame.

The time-stamp is a 60 bit number that encodes the time in milliseconds since Thursday, January 1st, 1970. The MSB in each word is set to one so that an erroneous synchronization pattern cannot appear in the data stream. Both the frame index and the time-stamp can be used to detect missing or dropped frames in the event that the data collection computer cannot keep up with the DACQ unit. However, it is possible for the timestamp to be erroneous if the reception of the FM synchronization signal is poor or the FM transmitter is not active. Conversely, the frame index is more robust, but rolls over after only 65535 frames, which corresponds to approximately 629 seconds. Therefore, a combination of the two should be used for aligning the data between multiple DACQ units.

The PGA gain setting and mode of operation that were programmed when the frame was recorded are encoded in a single byte as illustrated in Table 3.3 and Table 3.4. It should be noted that if a new gain or mode setting is programmed, it will take effect immediately, even in the middle of a frame. Therefore, it is possible for the data in a single frame to start with the settings listed in the header, but change partway through the frame. The new settings will be listed in the next frame, but there is no way to determine precisely at which sample the change occurred.

The relative signal strength indicator (RSSI) is an 8-bit integer between 1 and 255 that indicates the quality of the synchronization signal received from the FM transmitter. This information can be used to monitor the synchronization signal and is also used to tune the coil in the FM receiver circuit.

The minimum value and maximum value of the data samples in each frame is also reported in the header. This information can be used to automatically adapt the gain of the unit over time and also may speed up data normalization procedures that are part of many data post-processing steps.

Table 3.7: Format for the data samples in a DACQ frame.

Word Index	One Channel		Two Channels		Three Channels		Four Channels	
	Ch.	Sample	Ch.	Sample	Ch.	Sample	Ch.	Sample
12	0	0	0	0	0	0	0	0
13	0	1	1	0	1	0	1	0
14	0	2	0	1	2	0	2	0
15	0	3	1	1	0	1	3	0
16	0	4	0	2	1	1	0	1
17	0	5	1	2	2	1	1	1
18	0	6	0	3	0	2	2	1
19	0	7	1	3	1	2	3	1
⋮	⋮		⋮		⋮		⋮	
971	0	958	0	478	1	318	2	238
971	0	959	1	479	2	319	3	239

The header is immediately followed by 960 data samples. These samples are interleaved from each of the active channels in the data acquisition unit. Table 3.7 enumerates the order of the samples for each of the possible channel configurations. When the data is buffered, any ADC samples that are zero get changed to one so that an erroneous synchronization pattern cannot occur.

3.1.8 Data Acquisition Software

A data acquisition unit communicates with its host PC through a USB interface that emulates a serial port. Each unit has an on-board FPGA that buffers the data and controls the USB interface. The RAM blocks on the FPGA only have enough capacity to buffer three frames of data, so transferring them efficiently and without interruption is critical.

On the PC side, the stand-alone application, `dacq2pipe.exe`, is used to continuously transfer frames from the data acquisition unit to the hard disk of the host PC. The application takes the COM port number associated with the serial port of the data acquisition unit as an argument. As the data is transferred over the USB interface it is saved in two minute files. The total sampling rate of a data acquisition unit is 100kHz and there are 960 samples per frame. Therefore, each two minute file contains precisely $12,500$ frames. `dacq2pipe.exe` also opens a Windows pipe that allows other applications, such as MATLAB, to access the data as it is being streamed over the USB interface and saved.

3.2 Active Ultrasound Sensors

The velocity of a moving object relative to an observer can be estimated by measuring the frequency shift of a wave radiated or scattered by the object, known as the Doppler effect. If the object itself contains moving parts, each moving part will result in a modulation of the base Doppler frequency shift. This is the micro-Doppler effect that was described in Chapter 2.

In order to leverage this phenomenon for sensing action, a continuous sinusoidal carrier is transmitted and the modulated signal that is reflected off a moving object is observed. A sinusoid is the simplest Doppler sensitive waveform and is well suited for capturing frequency modulations due to motion. Continuous-wave (CW) sonar is always transmitting, so it has the advantage of maximizing the signal power reflected by a target. The downside of this setup is poor range resolution. To determine the range of a target, it must be possible to distinguish precisely when the the waveform that resulted in a reflection was transmitted so that the time-of-flight can be calculated. In the case of a continuous sinusoid, the transmitted waveform is repeated, so reflections from distances greater than the wavelength of the sinusoid could have come from ranges that are any multiple of the wavelength.

Table 3.8: The frequency and wavelength settings for each ultrasound carrier.

Carrier Frequency (kHz)	Wavelength (mm)
25	13.7
33	10.3
40	8.55

Our ultrasound units transmit one of three different ultrasonic frequencies. Table 3.8 enumerates the carrier frequency and its corresponding wavelength, assuming the speed of sound in air is $c = 343\text{m/s}$. The wavelengths for these frequencies are on the order of 10mm, so unless all of the energy was reflected from within this distance, determining the range of a target becomes ambiguous.

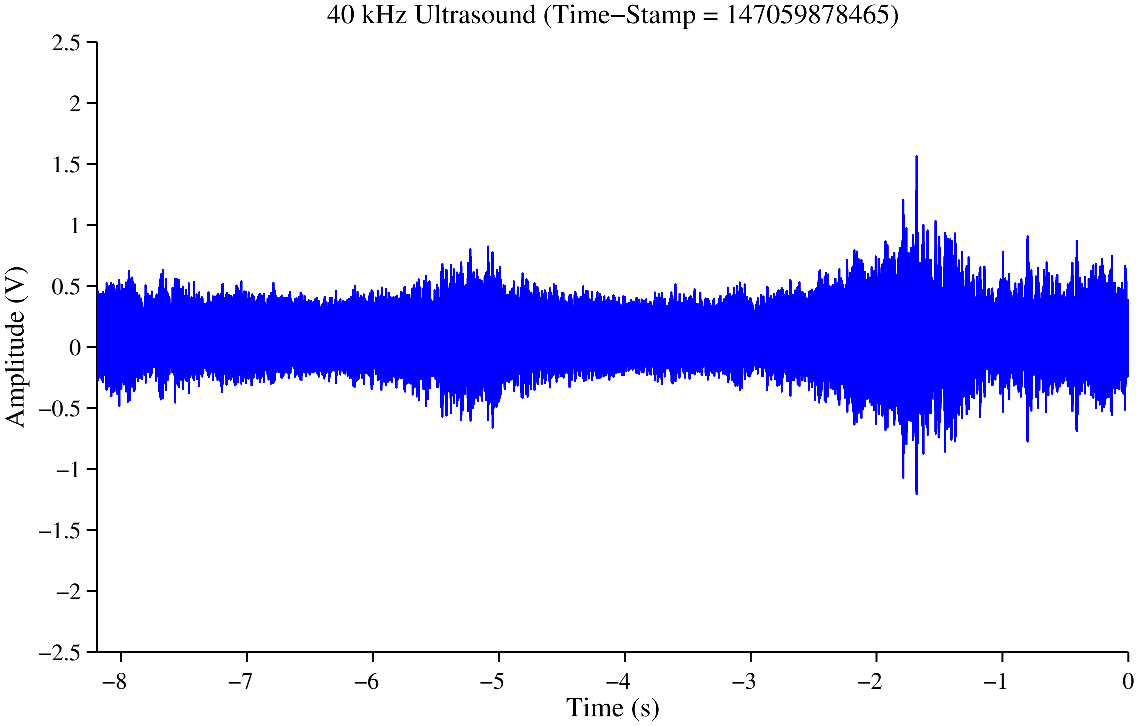


Figure 3.4: Raw time-series recorded by the ultrasound unit of a human walking.

A data acquisition unit can be configured as an ultrasound sensor by connecting the appropriate ultrasound receiver and transmitter transducers. Although the fourth channel is configured to output the transmitted waveform, this configuration is still considered single channel operation from the data acquisition perspective of recording the signal from the receiver, so the sampling rate is 100kHz. There are three sets of transducers centered at 25kHz, 33kHz and 40kHz. This allows separate data acquisition units to operate simultaneously without interference. Usually a single ultrasound unit is configured to use a transmitter and receiver in the same band. When the transmitter and receiver pair are co-located, the setup conforms to the monostatic sonar configuration described in Section 2.2. However, the system is flexible enough to support stand-alone transmitters and multiple receivers in a given band as well as multi-band combinations. For instance, with three data acquisition boxes, the system can be configured with one unit transmitting an ultrasonic sinusoid and two other units receiving the reflected signal.

Figure 3.4 shows an example of the raw time-series recorded by an ultrasound unit configured to transmit a 40kHz sinusoid signal and a co-located 40kHz receiver. The frequency modulations in the signal are induced by the micro-Doppler effect of the transmitted signal scattering off of a human walking in front of the sensor.

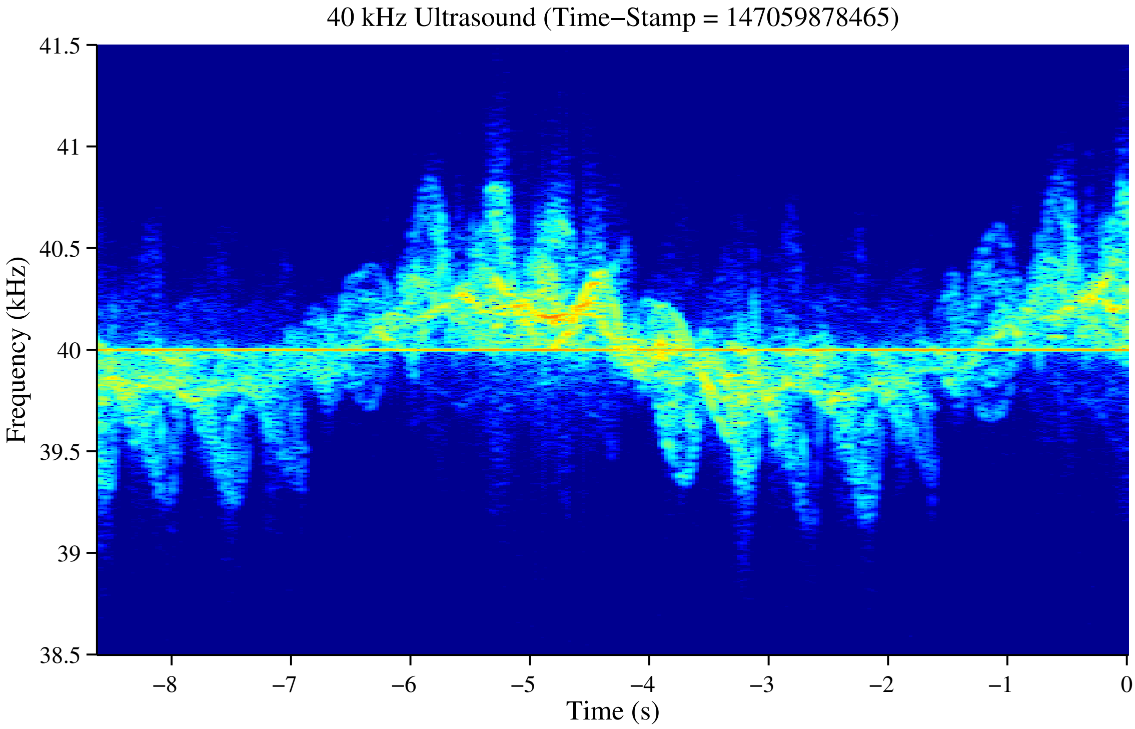


Figure 3.5: Spectrogram visualizing the micro-Doppler frequency modulations of a human walking.

Figure 3.5 shows an example spectrogram computed from the time-series shown in Figure 3.4. The time-frequency representation is more amenable to visualizing the micro-Doppler modulations in the signal. The overall translation of the human body as it walks back and forth is observed in the spectrogram modulations as the slower overall oscillation around the carrier frequency. The motion of the legs and arms is captured by the faster periodic modulations.

3.3 Microsoft Kinect Sensor

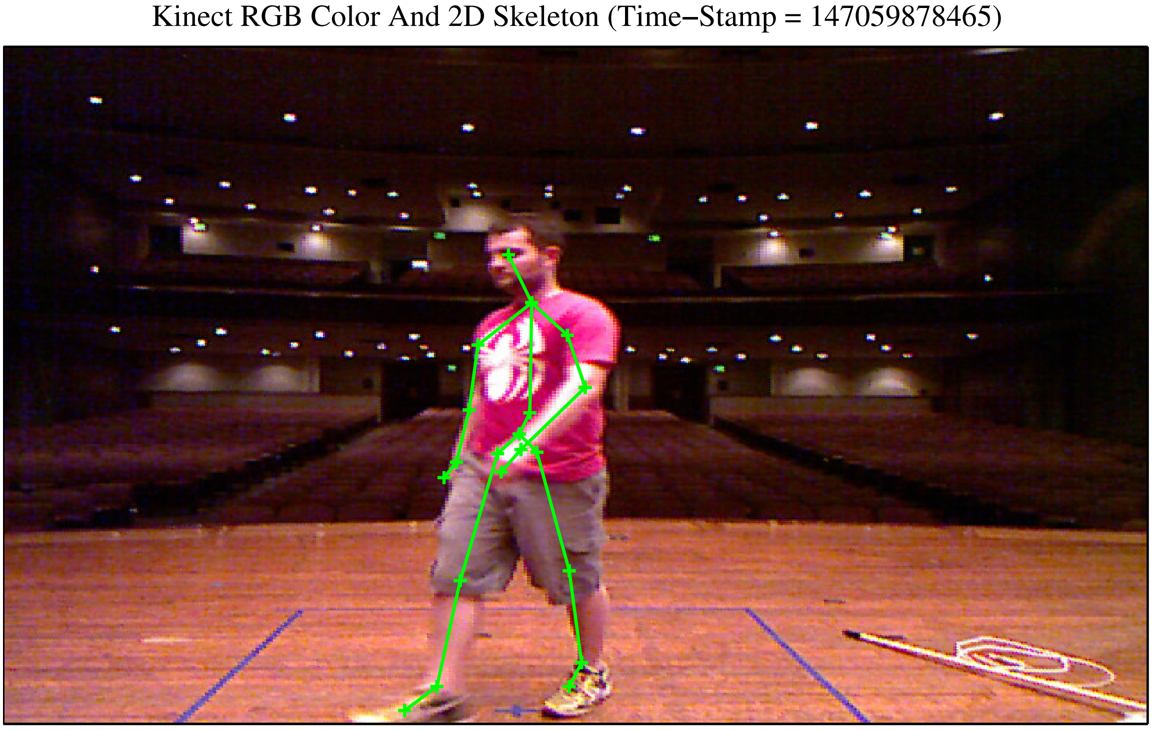

Kinect RGB Color And 2D Skeleton (Time−Stamp = 147059878465)

Figure 3.6: An example of the RGB color image captured by the Kinect for a human walking. The two-dimensional tracked skeleton is overlaid.

The Microsoft Kinect sensor actually combines two imaging sensors, an RGB imager and a grayscale depth imager, into a single package.[29] MATLAB interfaces with the Kinect through the image acquisition toolbox, which can capture the RGB image (640 × 480 pixels), the depth image (640 × 480 pixels), the two-dimensional skeleton tracking and the three-dimensional skeleton tracking. Figure 3.6 shows an example of an RGB image captured by the Kinect depth for a walking human and the associated two-dimensional skeleton.

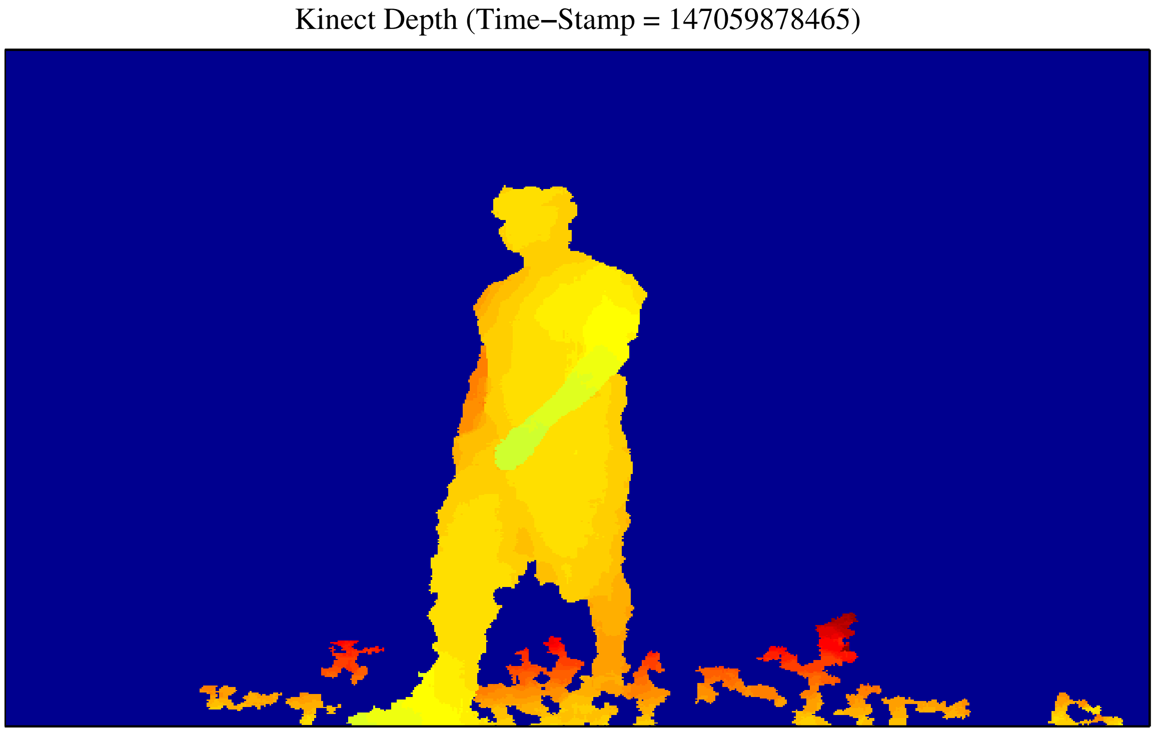


Figure 3.7: An example of the depth image captured by the Kinect for a human walking.

Based on the patent[30,31] filed by Prime Sense, a company that developed some of the intellectual property associated with the Kinect, the depth of each pixel in the image is computed using a combination of depth from focus and depth from stereo techniques. The Kinect sensor uses a pair of stereoscopic imagers to capture each frame. It also projects a known pattern of infrared light onto the scene using an astigmatic lens. The lens changes the orientation of the pattern depending on the depth of the object that it illuminates. The final grayscale depth map is computed by combining these techniques. Figure 3.7 shows an example of the Kinect depth map for the walking human portrayed in Figure 3.6.

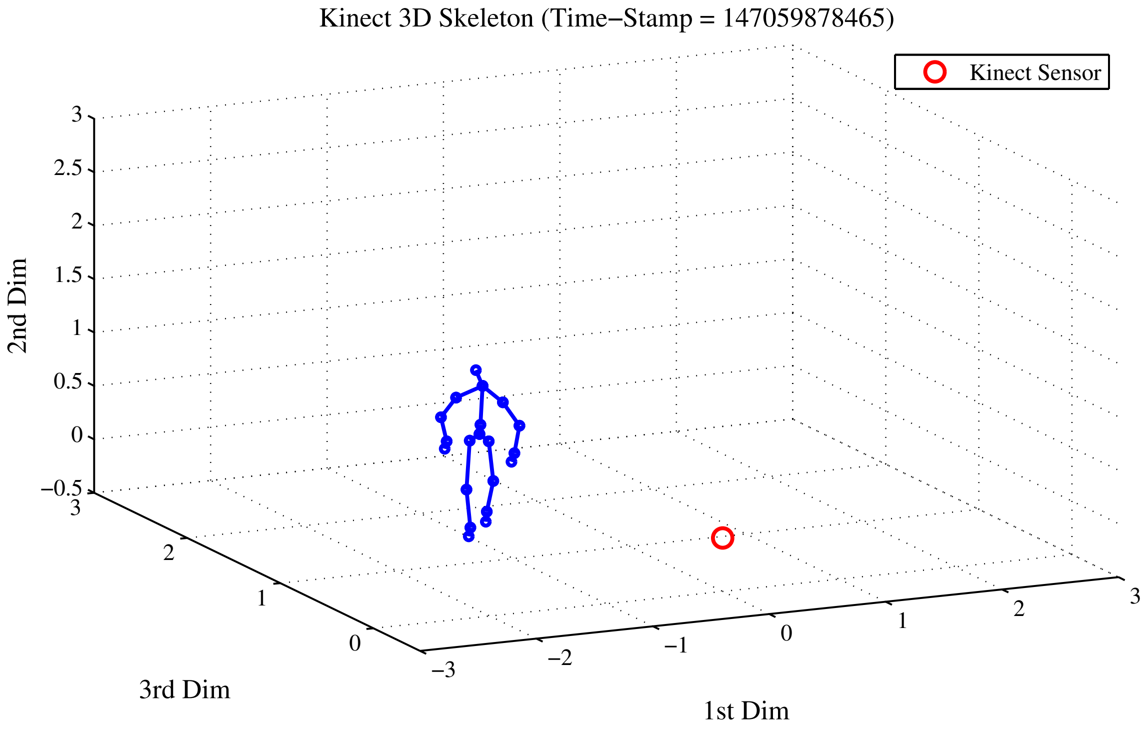


Figure 3.8: An example of three-dimensional skeleton tracked by the Kinect for a human walking.

The body position is tracked by applying randomized decision forests[32] to first tag each pixel in the depth map if it is a human body part and then inferring the position of the skeletal joints based on the tagged body parts. Figure 3.8 illustrates an example of the three-dimensional skeleton extracted from the Kinect depth map for a single frame. The global coordinate system used by the Kinect places the sensor at the origin and returns the skeletal joint positions in meters.

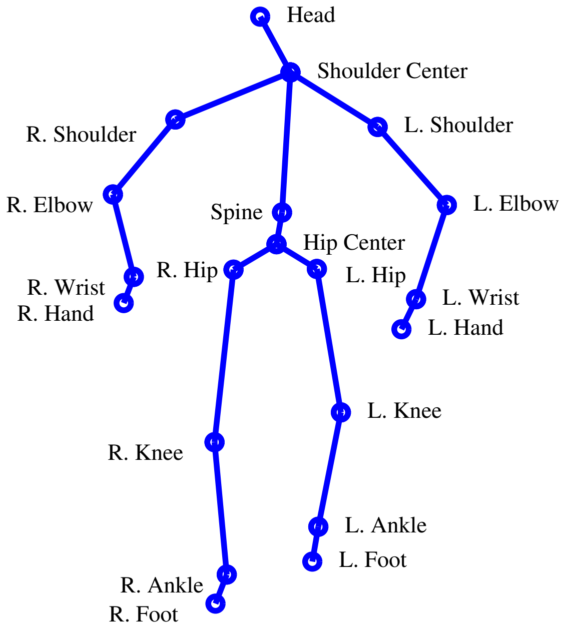


Figure 3.9: Joint labels for the Kinect skeletons.

Figure 3.9 shows the joint labels used to reference each of the joints tracked by the Kinect. The topology of the skeletal structure can by efficiently encoded as a tree, where the root is generally considered to be the hip-center joint. Although each joint is tracked in the global Kinect sensor space, there are several alternative representations that introduce spatial invariances that are useful for comparing different skeletal poses.

For human action recognition applications, it is preferable to develop algorithms capable of recognizing a particular action regardless of where it occurs in the global coordinate system. When training these algorithms it is advantageous to consider the hip-center as the origin for the skeleton at each frame. By referencing all of the other joints in a given frame to the position of the hip-center, the skeletal pose can be captured independently from the skeleton's global position. This skeletal pose representation provides translation invariance in the global coordinate system, which can greatly simplify the problem of recognizing a particular pose regardless of where a human is relative to the Kinect sensor. Storing the global position of the hip-center maintains all the necessary information to reconstruct the recorded scene exactly.

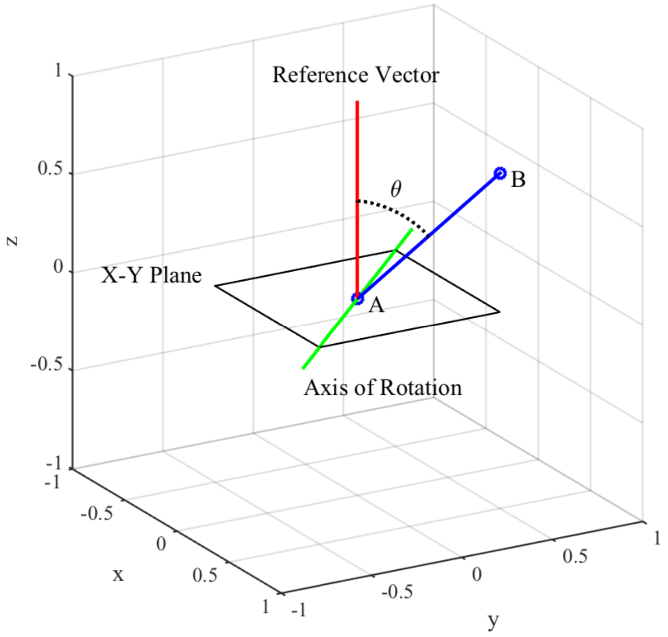


Figure 3.10: Components of the rotation representation for a single limb.

It is also beneficial if a human action recognition algorithm can be trained on skeletal poses collected from multiple subjects. One problem with the cartesian coordinates produced by the Kinect is their dependence on the height and limb lengths of the individual person. A very tall person and a very short person can perform the same action and generate very different cartesian joint coordinates even once the pose is adjusted to account for translation of the hip-center. However, the angle of the limbs as two people perform the same action is often much more consistent, even when their limbs are different lengths.

To leverage this invariance, the skeleton can be represented using the rotation of individual limbs instead of the cartesian coordinates of their constituent joints. The rotation representation is composed of two objects; an axis of rotation and an angle of rotation, θ. Figure 3.10 illustrates these components for a single limb. Each limb (blue line) is defined by two points, referred to as joint A and joint B. By convention, joint A is closer to the hip-center on the skeletal connection tree. The positive z-axis is used as a reference vector (red vector). The axis of rotation is the vector around which the limb must be rotated to match the reference. Due to the choice of reference, this axis is always constrained to the x-y plane.

One aspect of the rotation representation that makes them more complicated to use is the requirement that the precise limb lengths for each frame be stored independently in order to reconstruct the Kinect data exactly. Although limb lengths should be relatively consistent, errors in the skeletal tracking do not always result in consistent lengths. In this work, the limb lengths are extracted as the limb lengths averaged over each frame in the available data, for a particular individual.

Chapter 4

The Johns Hopkins University Multimodal Action Dataset

There are several standard datasets that have been developed over a period of years for action/activity analysis tasks. These are publicly available from various groups in the computer vision and machine learning communities and are aimed towards "action recognition" problems that have been an intense area of research in the respective research communities. All datasets are associated with publications that provide baseline performance scores and have "recognition" results reported in standard terms "recognition accuracy","precision", "precision/recall" curves[7,10,11,28,33–46]

The different datasets summarized in Table 4.1 have increasing "difficulty", starting with the KTH datasets that involve a static background and are actor staged and continuing with the UCF-101, HOHA-2 and HMDB51 that have more complex

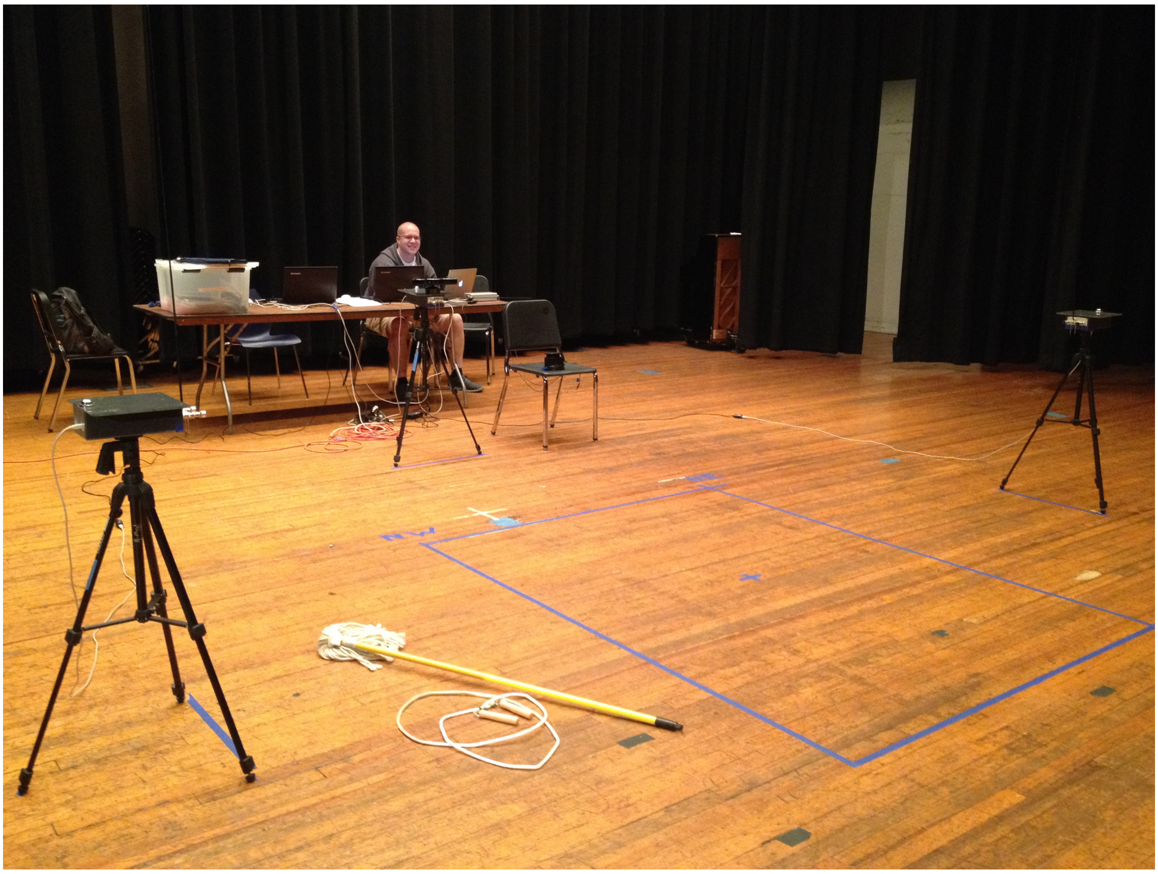


Figure 4.1: The JHUMMA data collection setup.

backgrounds from movies and videos. It should be noted that the HMDB51 dataset is selected in a way to accentuate action/activity categories that differ mainly in motion and hence especially suited for the development and evaluation of algorithms and an architecture of for a system that relies on spatial-temporal and dynamic cues as opposed to mostly shape cues, which most systems use today.

The Johns Hopkins University multimodal action (JHUMMA) dataset was collected on August 26th, 2014, in the auditorium of Shriver Hall, on Johns Hopkins University's Homewood campus. Three ultrasound sensors and a Kinect sensor were

Table 4.1: Action/Activity Recognition Datasets.

Dataset	Action Classes	Description
KTH[46]	6	static background, actor staged
Weizmann[45]	9	static background, actor staged
HOHA[44]	8	Hollywood movies
HOHA-2[43]	12	Hollywood movies
UCF-50[38]	50	online videos (Youtube)
HMDB-51[42]	51	online videos (Youtube, Google)
UCF-101[39]	101	online videos (Youtube)
SCANDAL[7,10,11,28]	7	active-passive acoustics, RGBD, static background actor staged
JHUMMA	21	static background, actor staged, RGBD, active acoustics

used to record joint multimodal data of ten unique actors performing a set of actions. The auditorium was chosen because it is a large open space and there are curtains on the stage where the data was collected. These features both reduce the number and strength of uninteresting reflections of the ultrasound carriers off static objects. The reduction in the total energy of unmodulated ultrasound reflections enables the use of higher gain settings on the ultrasound units, effectively increasing the dynamic range of the modulated ultrasound reflections that we are interested in. Figure 4.1 shows the JHUMMA data collection setup.

4.1 Data Collection Configuration

Figure 4.2 illustrates schematically the configuration of the various sensors used for the data collection. The bounding box, which corresponds to the area where the Kinect sensor reliably tracks a human, was marked on the auditorium stage to guide

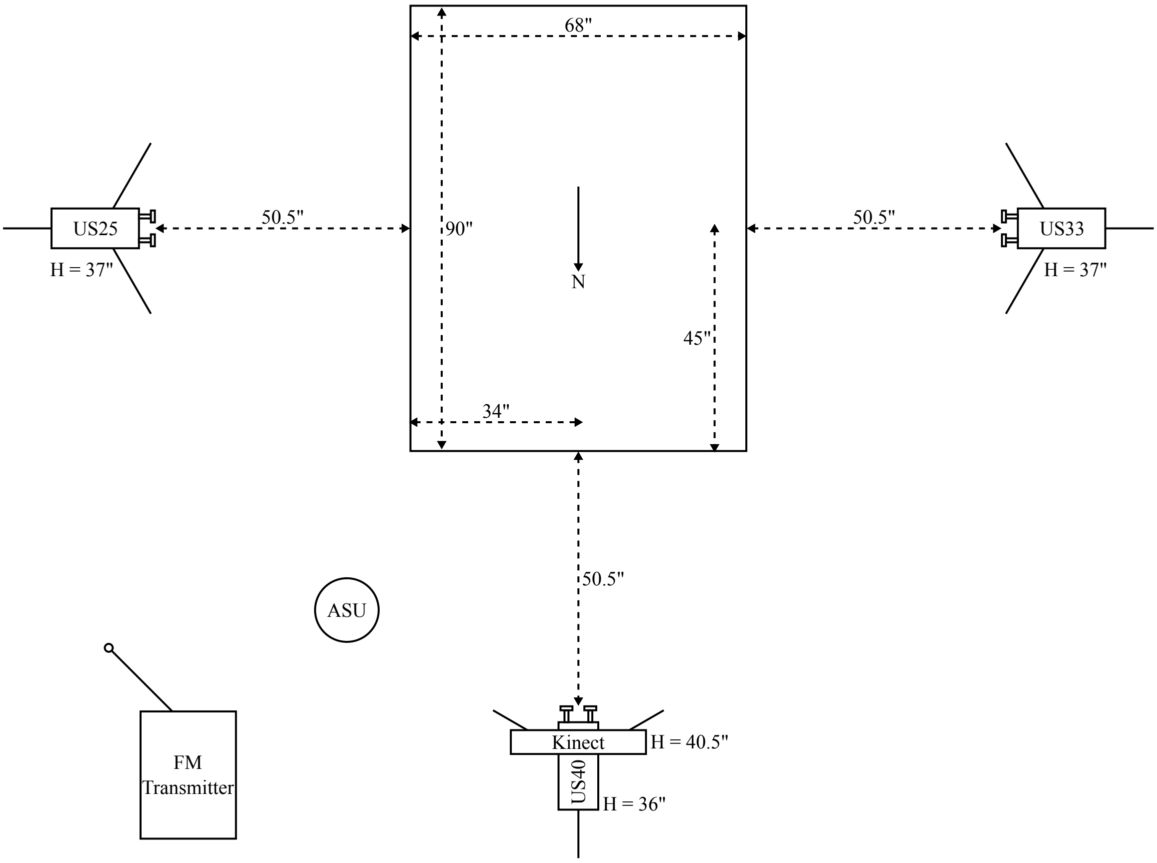


Figure 4.2: The experimental setup for the Shriver Hall data collection.

the actors. All actions were confined to this space and the orientation of the actions is referenced to the Kinect sensor, which was placed “north” of the bounding box. This cardinal direction convention was used to orient the recorded actions with respect to the sensor locations.

The Kinect sensor was placed directly on top of the 40kHz ultrasound sensor (US40). The 25kHz ultrasound sensor (US25) was placed to the east and the 33kHz ultrasound sensor (US33) was placed to the west.

4.2 JHUMMA Action Classes

Table 4.2: Script of actions nominally demonstrated by each actor.

Action	Orientation	Repetitions / Duration
Lunges	N	10 Per Leg (Alternating)
Lunges	NE	10 Per Leg (Alternating)
Lunges	NW	10 Per Leg (Alternating)
Left Leg Steps	N	10
Right Leg Steps	N	10
Left Arm Raise (Forward)	N	10
Left Arm Raise (Sideways)	N	10
Right Arm Raise (Forward)	N	10
Right Arm Raise (Sideways)	N	10
Walk in Place	N	20 Steps
Walk Facing Forward	N-S	10 cycles
Walk Facing Sideways	W-E	10 cycles
Walk and Pivot	NE-SW	10 cycles
Walk and Pivot	NW-SE	10 cycles
Jumping Jacks	N	10
Jump Rope	N	10
Body Squats	N	10
Jump Forward then Backward	N-S	10 sets
Jump Forward then Backward	NE-SW	10 sets
Jump Forward then Backward	NW-SE	10 sets
Punch Forward	N	10 Per Arm (Alternating)
Freestyle Mopping	NA	1 min
Freestyle Walking	NA	1 min

Table 4.2 enumerates the specific actions recorded for each actor and the nominal number of repetitions or duration of the action. This script was performed by ten unique actors and several performed it twice. A total of 14 runs through the script were recorded. Actions that involve the whole body and limit the occlusion of limbs maximize the potential of the Kinect sensor to accurately track the human skeleton. The majority of the actions included in the script were chosen based on this criteria.

Actions were also generally performed in an orientation relative to the Kinect sensor that mitigated the occlusion of body parts. Several actions were performed using multiple orientations so that the effects of recording ultrasound from several different angles is captured by the dataset. These orientations are also enumerated in Table 4.2.

4.3 Visualizing the Dataset

Figures 4.3 to 4.5 show a snapshots of the data recorded in the JHUMMA dataset during a single trial of each action. Figure 4.3 shows the first seven actions of the JHUMMA dataset according to the script in Table 4.2. Figure 4.4 shows the second seven actions and Figure 4.5 shows the final seven actions. The images in the first column were captured by the Kinect sensor's color imager and the two-dimensional skeleton track has been superimposed on top of the image. The spectrograms in the second column are generated from ultrasound data recorded by the 40kHz sensor. The spectrograms in the third column are generated from ultrasound data recorded by the 33kHz sensor. The spectrograms in the fourth column are generated from ultrasound data recorded by the 25kHz sensor. The time window used to choose the ultrasound data for each actions is the same for each sensor and the Kinect frames are all from within these windows. The window duration is fixed at just under nine seconds and is independent of the action. The displayed frequency content is a bandwidth of 2kHz centered on the respective carrier frequency of each ultrasound unit. The time and frequency markings are omitted for clarity.

By comparing the Doppler modulations recorded by each ultrasound sensor it is possible to gain a significant amount of information based on knowledge of the sensor positions and the orientation of the actions. For example, the right arm raise action only produces a modulation in the 25kHz ultrasound, which was on the right side of the actor's body. The 33kHz ultrasound, which was on the left side, exhibits almost no response as the body blocked almost all of the acoustic energy from reaching it. Conversely, the situation is reversed for the left arm raise action. The 40kHz ultrasound unit sees modulations for arm raise actions regardless of side because it is positioned in front of the body. However, the modulations are less pronounced as most of the motion is in the East to West plane.

There are also several other general characteristics of various actions that are easily observed. For example, the actions where the torso translates, such as lunges, walking and jumping, all exhibit a larger, relatively slow modulation component. Actions where only the limbs move much tend to simply oscillate around the carrier frequency because most of the actor's body is stationary. Additionally, based on the 40kHz ultrasound modulations, the jump rope action appears to produce the largest frequency shifts. This makes sense because the jump rope must move at a high velocity in order for the actor not to trip. The punching action generally produced the greatest frequency shifts generated purely by human motion.

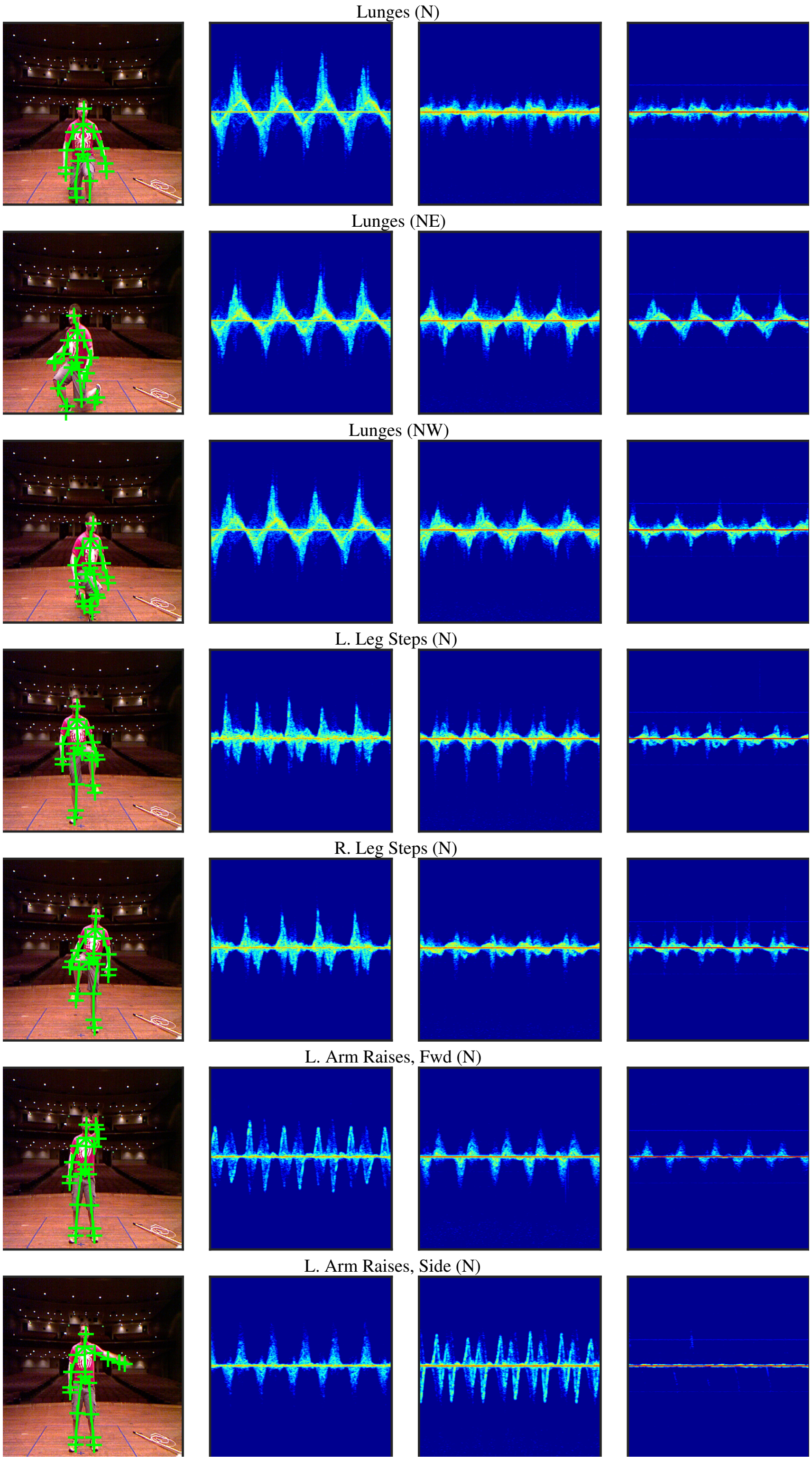


Figure 4.3: Examples of the first seven scripted actions in the JHUMMA dataset.

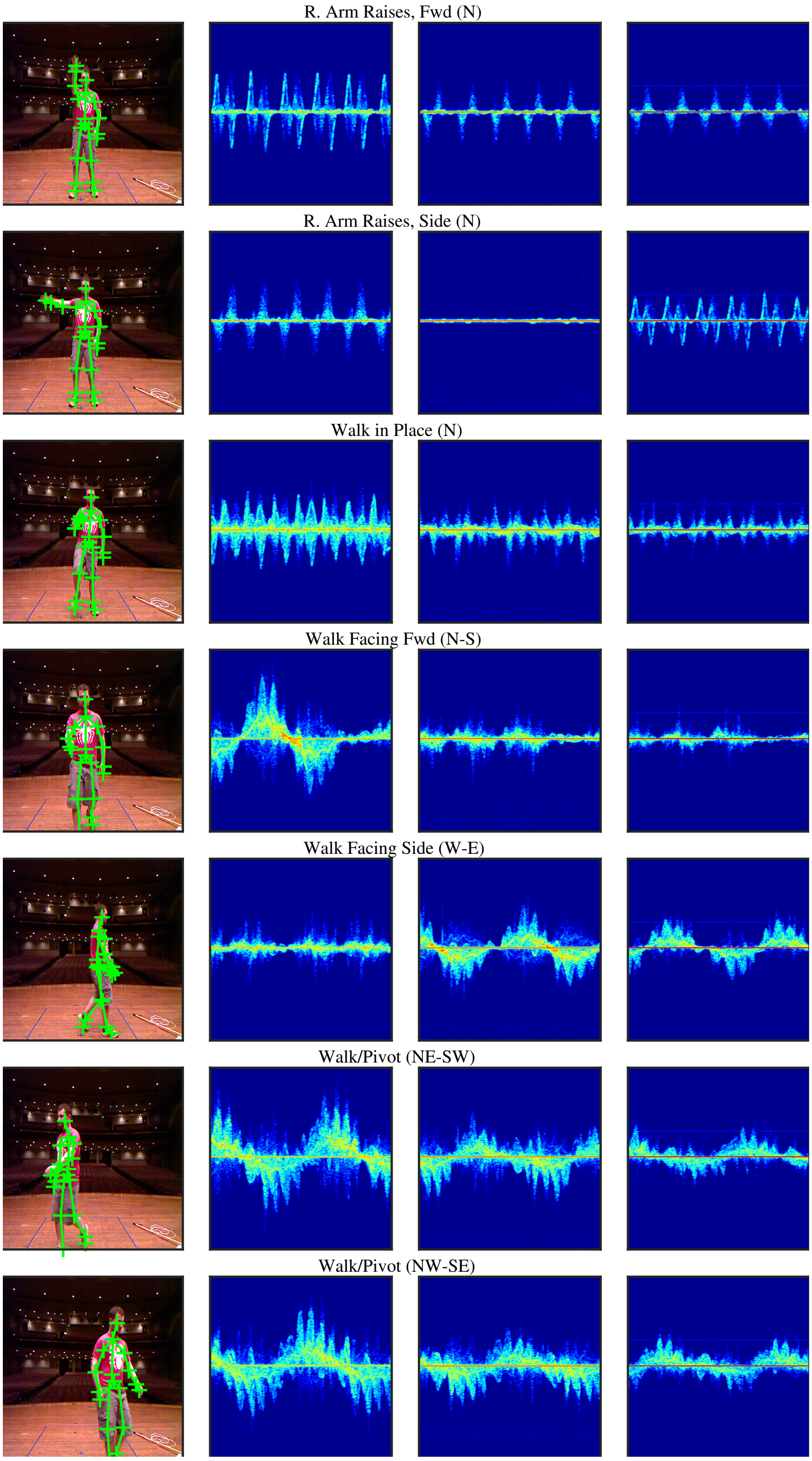


Figure 4.4: Examples of the second seven scripted actions in the JHUMMA dataset.

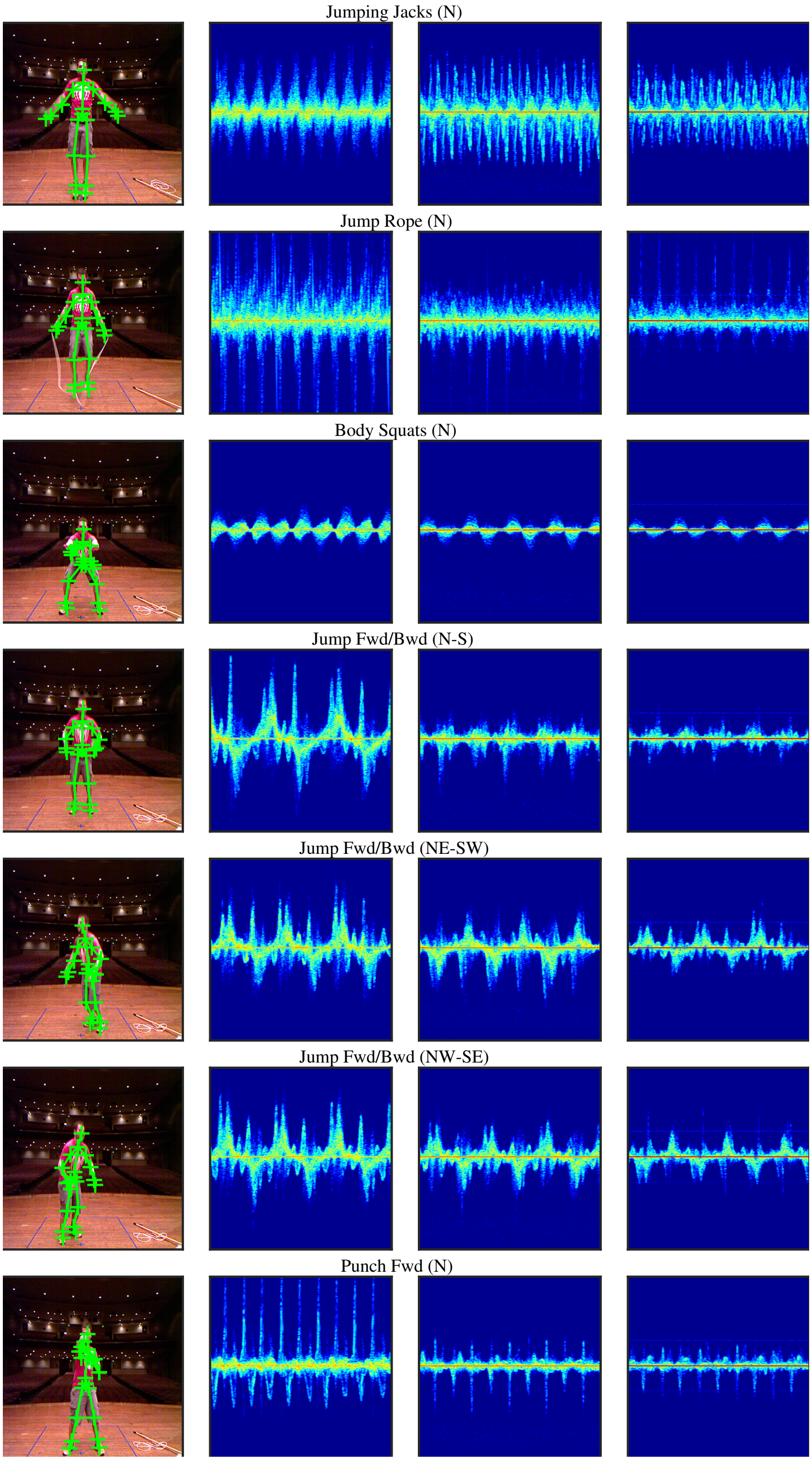
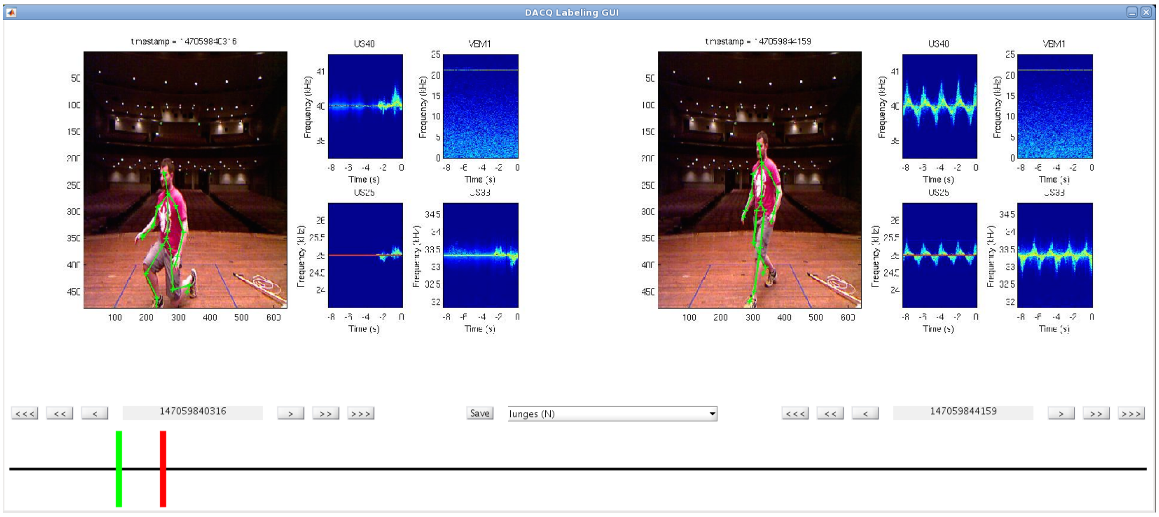


Figure 4.5: Examples of the final seven scripted actions in the JHUMMA dataset.

4.4 Processing the Dataset for Action Recognition Tasks



DACQ Labeling GUI

timestamp = 147059840316

timestamp = 147059844159

147059840316

lunges (N)

147059844159

Figure 4.6: Screenshot of the JHUMMA labeling GUI.

In order to facilitate the construction of supervised models built on the JHUMMA dataset, each action performed during each trial is coarsely labeled with a set of timestamps marking the beginning and end of the action sequence. Each coarsely labeled action sequence still contains several repetitions of the particular action. The coarse sequences are coherent temporal data streams, so more fine-grained labeling that would split them up further is deferred to the tasks that requires those labels. The graphical user interface (GUI), shown in Figure 4.6, was designed to facilitate labeling of the dataset. The left side of the GUI displays data from the timestamp tracked by the green bar. The data on the right side of the GUI displays data from the timestamp tracked by the red bar. All of the data within the timeframe between these two timestamps can be labeled as a single action.

Given the time-stamps that mark the beginning and end of each action sequence, the time-series data from each of the three ultrasound sensors is processed into a sequence of spectrogram slices. Using the temporal synchronization afforded by the DACQ system, the same series of overlapping time windows are used to produce the spectrograms for each ultrasound data stream. Throughout this work, the FFT window is set to 2^{14} samples, which corresponds to just over 0.1638 seconds. However, there is an 80% overlap between successive FFT windows, so the time window advances by approximately 0.0328 seconds between consecutive spectrogram slices. The spectrogram slices have also been limited to a bandwidth of 2kHz centered on the carrier frequency of the respective ultrasound sensor. These frequency ranges generally contain the micro-Doppler modulations and limit the dimensionality of the spectrogram modulations to reasonable range. Table 4.3 lists the velocity ranges that correspond to these frequency ranges.

Table 4.3: Velocity ranges of spectrogram slices.

Carrier Frequency (kHz)	Bandwidth Range (kHz)	Velocity Range (m/s)
40.00	39.00 to 41.00	-4.25 to +4.25
33.33	32.33 to 34.33	-5.10 to +5.10
25.00	24.00 to 26.00	-6.80 to +6.80

The timestamps corresponding to these windows are also used to produce a series of corresponding skeletal frames, one for every spectrogram slice. The average frame rate from the Kinect is around 30 frames per second, which corresponds to approximately 0.0333 seconds. This happens to match the rate of spectrogram slices very closely.

Now that a dataset has been developed that incorporates both visual data, which facilitates the accurate tracking of human movement, and active acoustic data, which captures the micro-Doppler modulations induced by the motion, the next chapter begins to develop models that capture the interaction between these modalities. The immediate goal is to accurately classify the actions in the JHUMMA dataset from their acoustic modulations. Later, more sophisticated models are developed that can accurately predict the acoustic modulations from the configuration of the human motions and even hallucinate new modulations with configurable characteristics.

Chapter 5

A Simple Generative Model for Human Action Recognition and Active Acoustics

For years, the speech recognition community has had great success modeling utterances with statistical models. In the traditional setup for speech recognition,[47] the data is a sequence of acoustic evidence, $\mathbf{A} = a_1, ..., a_m$. Each element a_i, in the sequence is drawn from some alphabet $\mathcal{A}$, such that $a_i \in \mathcal{A}$. This alphabet is often a discrete and finite set of phonemes. The acoustic data is then used to predict the sequence of words $\mathbf{W} = w_1, ..., w_n$ that constitute the utterance. The individual words, w_i, are drawn from some vocabulary $\mathcal{W}$, such that $w_i \in \mathcal{W}$. The goal of speech recognition is to find the best estimate for the uttered sequence,

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} P(\mathbf{W}|\mathbf{A}), \tag{5.1}$$

where $P(\mathbf{W}|\mathbf{A})$ is the probability of the word sequence $\mathbf{W}$ being uttered given the acoustic evidence $\mathbf{A}$. Unfortunately, the probability that a particular word sequence is generated given a set of acoustic data is not easy to model. To get around this, Bayes rule is often applied to split the expression up into subproblems. Thus,

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} P(\mathbf{A}|\mathbf{W})P(\mathbf{W}). \tag{5.2}$$

Here, $P(\mathbf{W})$, often referred to as the language model, captures the relative frequency of different words within the context that is relevant for the speech recognition system. In general, these statistics are reliably culled from transcriptions or other text documents. The likelihood that a word generates a particular sequence of acoustic data is captured by $P(\mathbf{A}|\mathbf{W})$, often referred to as the acoustic model. Estimating $P(\mathbf{A}|\mathbf{W})$ tends to be a much simpler problem because it is reasonable to develop compositional models that are built on components that model the acoustics for single words and pairs of words.

There are many similarities between human speech and human activity. Both are temporal sequences and Table 5.1 enumerates the basic components that compose each sequence. In speech, the features extracted from an acoustic time-series are

Table 5.1: Comparison of speech and action objects.

Speech	Activity
MFCCs	Spectrogram Frequency Bins
Phonemes	Skeletal Poses
Words	Actions
Sentences	Activity

traditionally Mel-scale cepstral coefficients (MFCCs). These features are extracted by a set of logarithmic spaced filters in the frequency domain over the audible range. Conversely, for the acoustic data in the JHUMMA dataset, the spectrogram features are linearly spaced in the frequency domain and centered in a narrow bandwidth around the carrier frequency of interest. In both cases, these features are computed for short time windows on the order of a few milliseconds each.

At the most granular level, the latent state space used for speech recognition is a set of symbolic phoneme labels. An analogous latent state for action recognition would be a set of symbolic skeletal poses. Sequences of phonemes are strung together to form words just as actions are formed by stringing together sequences of skeletal poses over time. At the highest level of granularity, sentences are formed by sequences of words that are governed by the grammar of a given language. Similarly, human activities are formed by sequences of actions that are governed by the task at hand. In fact, language models for human activity have even been proposed.[48–50]

Complex utterances are composed from sequences of words just as complex actions are formed from sequences of motions. In turn words are composed of sequences of phonemes classified from a speech signal. Drawing upon this methodology and developing a set of patterns classified from active acoustic signals is a reasonable approach to begin investigating the potential of micro-Doppler modulations to recognize and classify human actions. In this chapter, the action recognition task is roughly analogous to an isolated word recognition task in speech.

One major difference between speech and action is that human language is built around a core symbolic representation that is comparatively easy to define as a finite vocabulary of words. Human action on the other hand is composed of motions that are much more ambiguous to define. In lieu of trying to develop a well-defined dictionary of sub-motions, a set of representative skeletal pose prototypes are learned directly from the Kinect skeletal tracks. Similarly, much work has gone into developing the alphabet of phoneme classes that discretize the speech time-series into a sequence of symbols. No such symbolic representation exists for micro-Doppler modulations in active acoustics. Therefore, a set of representative spectrogram slice prototypes are learned directly from the active acoustics modulations.

The JHUMMA dataset is critical to training and testing models built using this approach. The availability of multimodal data that supports building symbolic representations for both human sub-motions and acoustic modulations for labeled actions provides the foundation for training supervised models. Moreover, the temporal alignment capabilities provided by the synchronization beacon integrated into the DACQ system enables the models to learn statistical correlations between the modalities accurately.

CHAPTER 5. A SIMPLE GENERATIVE MODEL FOR HUMAN ACTION RECOGNITION AND ACTIVE ACOUSTICS

In Section 5.1 the speech recognition approach is adapted more formally to the action recognition task of classifying the samples in the JHUMMA dataset. Section 5.2 describes the structure of the hidden Markov model (HMM) used to model the actions and their acoustic observations. Section 5.3 details the parameters of the models and develops the procedure for estimating them from training data. Section 5.4 develops the Viterbi algorithm for HMMs that is used to find the most likely skeletal pose sequence given a test sequence of spectrogram slices. Section 5.5 describes how the JHUMMA dataset is divided into cross-validation datasets as well as the division of training and testing examples. Section 5.6 describes the K-means algorithm used to learn prototype clusters from the training data for both the skeletal poses and the spectrogram slices. Section 5.7 examines the skeletal pose prototypes and provides additional details on their construction. Section 5.8 examines the spectrogram slice prototypes for each of the three ultrasound bands. Section 5.9 presents the classification procedure and results on the JHUMMA dataset. Section 5.11 illustrates example test sequences of each action and their prototype representations. It also suggests the limitations of a simplistic model like the HMM for capturing longterm correlations required to properly model actions.

5.1 The Active Acoustic Action Recognition Model

Assume that an appropriate vocabulary, $\mathcal{H}$, of skeletal poses exist. A sequence of skeletons can then be described as $\mathbf{H} = h_0, ..., h_T$, where $h_t \in \mathcal{H}$. Also assume that an appropriate alphabet, $\mathcal{V}$ of acoustic spectrogram slices exist. A spectrogram can then be described as $\mathbf{V} = v_1, ..., v_T$, where $v_t \in \mathcal{V}$. The methodology for generating the vocabulary and alphabet is developed later in Sections 5.7 and 5.8, respectively. There is also a set of action class labels $\mathcal{C}$ that enumerate the twenty-one actions contained in the JHUMMA dataset. Each sequence $\mathbf{H}$ is generated by an action, $a \in \mathcal{C}$, that modifies the parameters of the probability distributions accordingly.

The goal of the action recognizer is to estimate the most likely action that produced the visible sequence $\mathcal{V}$ of spectrogram slices. This can be expressed as

$$\hat{a} = \arg\max_{a} \Big(\max_{\mathbf{H}} \; P_a(\mathbf{V}|\mathbf{H}) P_a(\mathbf{H}) \Big). \tag{5.3}$$

where the same Bayes decomposition is applied to split the probability of a skeletal pose sequence given a spectrogram into a product of the action model, $P_a(\mathbf{H})$, and the active acoustic model, $P_a(\mathbf{V}|\mathbf{H})$, that is used in the speech recognition setting. However, the action recognition task differs from speech recognition in that the focus is on correctly identifying the particular action that gave rise to the observed acoustic modulations instead of the the unobserved sequence of sub-motions. This change is manifest in the additional maximization over the possible action labels such that $\hat{a}$ is the best estimate for that action.

An HMM can be used to model a single pair of visible and hidden sequences, whose joint probability can be decomposed as $P_a(\mathbf{V}|\mathbf{H})P_a(\mathbf{H})$. In order to leverage this model for recognizing actions, a set of HMM parameters are each trained separately on the portions of the training data that contain examples of a single action, a. When a new test sequence of acoustic spectrogram slices, $\mathbf{V}$, is observed, each of the individual action HMMs are used to predict the most likely sequence, $\mathbf{H}$, of unobserved sub-actions. Computing the likelihoods of the sequences produced using each set of action parameters allows the models to predict the most likely action a by choosing the model that produces the most likely sequence. In Section 5.2, the HMM model for sub-motions and active acoustics is developed further.

Finally, there are actually three independent sets of ultrasound observations in the JHUMMA dataset. In order to investigate the effects of using active acoustics from different orientations, three separate sets of HMMs are developed, one for each ultrasound sensor. While they can all share the same skeletal pose state space built upon the Kinect sensor data, their observation state spaces are all unique, requiring independent sets of parameters.

5.2 The Hidden Markov Model

The chain rule of probability states that the joint distribution of a set of D random variables can be decomposed into a product of joint probabilities such that,

$$P(X_1, ..., X_D) = P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)\dots P(X_D|X_{D-1}, ..., X_1), \tag{5.4}$$

where the order of the random variables is interchangeable. When using a stochastic process to model a temporal sequence of data, this decomposition of the joint distribution becomes problematic as the number of conditioning terms in the individual conditional probability terms grows linearly with the length of the sequence in addition to the total number of terms growing with the length of the sequence. However, it is possible to make assumptions about the conditional independence between some of the random variables in order to produce a simpler factorization of the joint distribution.

A Markov chain is the simplest stochastic process that still incorporates memory. The Markov property states that the probability of the next state in the process depends only on the previous state. The joint distribution of a sequence of D random variables can now be decomposed as,

$$P(X_1, ..., X_D) = P(X_1)P(X_2|X_1)P(X_3|X_2)\dots P(X_D|X_{D-1}). \tag{5.5}$$

Now the number of terms still grows linearly with the sequence, but the number of random variables each term is conditioned on is constant. Of course the random variables are no longer interchangeable, but that is a reasonable restriction for modeling temporal sequences.

An HMM is an extension of a Markov chain where the random variables in the Markov sequence are considered hidden and not observed. Instead, an additional visible random variable is observed at each step in the sequence. The visible random variable is conditional independent of all the other hidden and visible random variables given the hidden variable at that step. Figure 5.1 depicts the basic structure of the HMM used to represent the active acoustic action model.

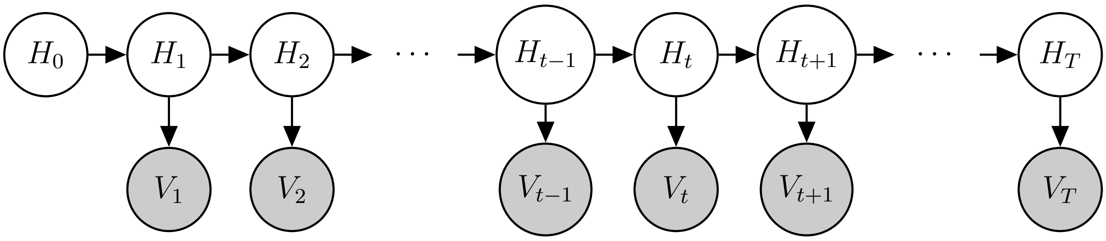


Figure 5.1: Structure of the HMM used to capture sequences of Doppler modulations given a hidden sequence of skeletal poses.

The hidden sequence is encoded by the variables $\mathbf{H} = H_0, H_1, ..., H_T$. The visible sequence is encoded by the variables $\mathbf{V} = V_1, ..., V_T$. The HMM also has a start state H_0 that is used to encode a set of prior probabilities indicating the likelihood that a chain starts in each state.

This HMM encodes the structure of the sub-motion and spectrogram slice sequences for a specific action a in the action recognition model if the sub-motions are assumed to adhere to the Markov property. Under this condition, the joint probability of the HMM can be decomposed as,

$$P_a(\mathbf{H}_a, \mathbf{V}) = P_a(\mathbf{V}|\mathbf{H}) \cdot P_a(\mathbf{H}) = \Big(\prod_{t=1}^{T} P_a(V_t|H_t)\Big) \cdot \Big(\prod_{t=1}^{T} P_a(H_t|H_{t-1})\Big) P_a(H_0). \quad (5.6)$$

The factorization of the joint distribution coincides with the conditional independence assumptions encoded in the graphical structure of the HMM. In Equation 5.6, the dependence of the individual hidden variables H_t on the action a was suppressed for conciseness. The HMM is a generative model because it encodes the full joint probability distribution instead of just the discriminative class conditional distribution $P_a(\mathbf{H}|\mathbf{V})$.

The HMM parameters for action a are $\boldsymbol{\theta}_a = (\boldsymbol{\pi}_a, \mathbf{A}_a, \mathbf{B}_a)$, where $\boldsymbol{\pi}_a$ is the vector of state priors, $\mathbf{A}_a$ is the matrix of transition probabilities between the hidden skeletal pose states and $\mathbf{B}_a$ is the matrix of emission probabilities of spectrogram slices from each of the hidden states. There are $|\mathcal{H}|$ hidden skeletal pose states and $|\mathcal{V}|$ visible spectrogram slice states. If $i \in \{1, ..., |\mathcal{H}|\}$ indexes into the the set of possible hidden states, then the elements of the state prior vector are,

$$\pi_a(i) = P_a(H_0 = i). \quad (5.7)$$

If $i, j \in \{1, ..., |\mathcal{H}|\}$ both index into the set of possible hidden states, then the elements of the transition matrix are,

$$A_a(i,j) = P_a(H_t = j | H_{t-1} = i). \tag{5.8}$$

If $i \in \{1, ..., |\mathcal{H}|\}$ indexes into the the set of possible hidden states and $k \in \{1, ..., |\mathcal{V}|\}$, then the elements of the emission matrix are,

$$B_a(i,k) = P_a(V_t = k | H_t = i). \tag{5.9}$$

5.3 Learning the Hidden Markov Model Parameters

When the training data includes instances of both the hidden and visible sequences, the parameters for the HMM can be learned via closed-form maximum likelihood estimates (MLE).[47,51–53] To derive the MLE estimates, consider the joint probability of a training example, $(\mathbf{V} = \mathbf{v}, \mathbf{H} = \mathbf{h})$, where both the hidden and visible variables are known. Using Equation 5.6 gives the probability of the training example,

$$\begin{aligned} P_a(\mathbf{H}=\mathbf{h},\mathbf{V}=\mathbf{v}) &= \Big(\prod_{t=1}^{T} B_a(V_t,H_t)\Big)\cdot\Big(\prod_{t=1}^{T} A_a(H_t,H_{t-1})\Big)\pi_a(H_0) \\ &= \prod_{t=1}^{T}\prod_{i=1}^{|\mathcal{H}|}\prod_{k=1}^{|\mathcal{V}|} B_a(i,k)^{\mathbb{I}(H_t=i,V_t=k)} \\ &\times \prod_{t=1}^{T}\prod_{i=1}^{|\mathcal{H}|}\prod_{j=1}^{|\mathcal{H}|} A_a(i,j)^{\mathbb{I}(H_t=j,H_{t-1}=i)} \\ &\times \prod_{i=1}^{|\mathcal{H}|} \pi_a(i)^{\mathbb{I}(H_0=i)}. \end{aligned} \tag{5.10}$$

The parameters $\boldsymbol{\theta}_a$ have been substituted for the appropriate probability distributions and the indicator function $\mathbb{I}$ is used to specify the number of times each probability term occurs. The probability of L independent training sequences is simply $\prod_{l=1}^{L} P_a(\mathbf{H}=\mathbf{h}_l,\mathbf{V}=\mathbf{v}_l)$. sequences of training examples. Taking the log of this distribution yields,

$$\begin{aligned} \sum_{l=1}^{L}\log P_a(\mathbf{H}=\mathbf{h}_l,\mathbf{V}=\mathbf{v}_l) &= \sum_{i=1}^{|\mathcal{H}|}\sum_{k=1}^{|\mathcal{V}|} N_{ik}\log B_a(i,k) \\ &+ \sum_{i=1}^{|\mathcal{H}|}\sum_{j=1}^{|\mathcal{H}|} N_{ij}\log A_a(i,j) \\ &+ \sum_{i=1}^{|\mathcal{H}|} N_i \log \pi_a(i). \end{aligned} \tag{5.11}$$

Here the emission counts across the training set are defined as,

$$N_{ik} = \sum_{l=1}^{L} \sum_{t=1}^{T} \mathbb{I}(H_{l,t} = i, V_{l,t} = k). \tag{5.12}$$

The transition counts across the training data are defined as,

$$N_{ij} = \sum_{l=1}^{L} \sum_{t=1}^{T} \mathbb{I}(H_{l,t} = j, H_{l,t-1} = i). \tag{5.13}$$

The prior counts across the training data are defined as,

$$N_{i} = \sum_{l=1}^{L} \mathbb{I}(H_{l,0} = i). \tag{5.14}$$

It is necessary to add additional constraints via Lagrange's multiplier. Essentially, the fact that the parameters are also proper probability distributions, and therefore sum to unity, must be enforced. That is, $\sum_{i=1}^{|\mathcal{H}|} \pi_a(i) = 1$, $\sum_{j=1}^{|\mathcal{H}|} A(i,j) = 1$ and $\sum_{k=1}^{|\mathcal{V}|} B(i,k) = 1$. To find the MLE estimates for the various parameters, first add the appropriate constraint to the log-likelihood in Equation 5.11. Let λ be the Lagrange multiplier coefficient. Then take the partial derivatives of the constrained log-likelihood with respect to both the parameter of interest and λ. This results in two equations and two unknowns. For more details on using the Lagrangian to find the MLEs of the parameters, see Chapter 3 in Murphy.[54] Solving the system of equations for the state prior probabilities yields,

$$\hat{\pi}_a(i) = \frac{N_i}{\sum_{i'=1}^{|\mathcal{H}|} N_{i'}}. \tag{5.15}$$

Solving for the transition probabilities yields,

$$\hat{A}_a(i,j) = \frac{N_{ij}}{\sum_{j'=1}^{|\mathcal{H}|} N_{ij'}}. \tag{5.16}$$

Solving for the emission probabilities yields,

$$\hat{B}_a(i,k) = \frac{N_{ik}}{\sum_{k'=1}^{|\mathcal{V}|} N_{ik'}}. \tag{5.17}$$

Under the supervised training paradigm, finding the MLE estimates for the HMM parameters essentially boils down to counting the number of times the relevant event occurred in the training data and normalizing the results into proper distributions. In order to train one set of HMM parameters for each action $a \in \mathcal{C}$, the training data is split according to the action that generated it and the parameters for each action are trained solely on the associated training data.

Many of the possible hidden state transitions and visible observation combinations were never observed in the training sets. To alleviate this, add-one smoothing was applied to the MLE estimates. This technique amounts to adding one phantom count to each element prior to normalization.

5.4 Finding the Most Likely Hidden Sequence in a Hidden Markov Model

Given the trained parameters for an HMM and a test sequence of observations, the Viterbi algorithm[47, 52, 55] can be used to find the most likely sequence of hidden states. The Viterbi algorithm is a dynamic programming technique to efficiently compute the maximum a posteriori (MAP) probability estimate of the most likely sequence in a chain-structured graphical model, such as the HMM.

The Viterbi algorithm is composed of a forward pass through all possible sequences of states where the likelihood of ending up in state $j \in \{1, ..., |\mathcal{H}|\}$ at time $t \in \{1, ..., T\}$ is computed for each state. Given an observed sequence $V_1 = k_1, ..., V_T = k_T$, the likelihood $\delta_t(j)$ of a state j, at each time step t, can be computed based on the likelihoods of the states at the previous time step $t-1$, the transition probabilities between the states and the probability that each state emits the current observed symbol k_t,

$$\delta_t(j) = \max_{i=1,...,|\mathcal{H}|} \delta_{t-1}(i)A(i,j)B(j,k_t). \tag{5.18}$$

The forward pass can be initialized using the prior probability of each state such that,

$$\delta_1(j) = \pi(j)B(j,k_1). \tag{5.19}$$

In addition to tracking the likelihood of each state, the previous state that gave rise to the likelihood is also tracked.

$$\alpha_t(j) = \underset{i=1,\ldots,|\mathcal{H}|}{\arg\max}\ \delta_{t-1}(i)A(i,j)B(j,k_t). \tag{5.20}$$

The Viterbi algorithm for an HMM terminates once the final time step T is reached. At this point the sequence of most likely states can be traced backwards through time. Beginning at time step T, the most likely state is

$$h_T^* = \underset{i=1,\ldots,|\mathcal{H}|}{\arg\max}\ \delta_T(i), \tag{5.21}$$

and the sequence is unrolled using the previous states that were tracked. Thus,

$$h_t^* = \alpha_{t+1}(h_{t+1}^*). \tag{5.22}$$

5.5 Splitting the JHUMMA Dataset into Examples and Batches

The JHUMMA dataset provides a perfect foundation for building HMMs that jointly model sequences of skeletal poses and sequences of Doppler-modulations and to evaluate their ability to classify actions sequences. The classification task requires a well-defined set of actions. The freestyle mopping and freestyle walking sequences were omitted as some of the motions involved in the sequences are examples of some of the other actions, such as walking and pivoting. This leaves 21 distinct action classes on which to evaluate the performance of the HMMs.

As described in Section 4.4, the JHUMMA dataset contains a sequence of spectrogram slices for each of the three ultrasound sensors and a sequence of skeletal frames for each of the actions performed during each of the thirteen trials. Unfortunately, each of these coarsely labeled sequences contains multiple repetitions of the same action. Nominally each sequence contains ten repetitions, although there are several instances where the actor lost track of the count. In order to generate test sequences suitable for training and testing HMMs, each sequence was split into ten examples of equal numbers of consecutive frames. Any remaining frames were appended to the last example so that temporal cohesion is maintained.

Five batches of training and testing data were set up for cross-validation. For each action/trial pair, two of the ten data sequences were randomly selected as test examples, while the remaining eight were selected as training examples. The random permutations were constructed such that each of the ten examples serves as a test sequence in exactly one of the five batches. One of the actors accidentally skipped three actions, so there are precisely $2,160$ training examples and 540 test examples in each batch.

5.6 Learning Cluster Prototypes

The K-means algorithm is a common method for performing vector quantization,[54,56] a technique for modeling probability densities based on the location of prototype vectors. The idea behind K-means was first proposed by Steinhaus as least squares quantization in pulse-code modulation (PCM) and the standard algorithm used to implement the technique was first published by Lloyd[57,58] with efficient large scale applications of the algorithm advanced by Coates[59] and Kanungo.[60] In Sections 5.7 and 5.8 the details of using K-means to learn prototype clusters for both the skeletal poses and spectrogram slices are described. Here the basic algorithm is developed for performing unsupervised clustering.

Let $\mathbf{X} = \mathbf{x}_1, ..., \mathbf{x}_N$ be a set of unlabeled training data, where each $\mathbf{x}_i \in \mathbb{R}^D$. Define a set of $j = \{1, ..., K\}$ cluster prototypes $\boldsymbol{\mu}_j \in \mathbb{R}^D$. The cluster prototypes are initialized using the K-means++ algorithm,[61] which randomly selects one of the data points from $\mathbf{X}$ to be the first cluster prototype and selects subsequent points, one at a time, from $\mathbf{X}$ to be initial cluster prototypes with probability inversely proportional to their distance from the nearest existing selected prototype.[60]

Once all K of the clusters have been initialized, the training data points are all assigned to the nearest cluster. In this work, the distance between any data point in the training set and any cluster mean is given by

$$d(\mathbf{x}_i, \boldsymbol{\mu}_j) = \sqrt{\sum_{d=1}^{D}(x_d - \mu_d)^2}, \tag{5.23}$$

the Euclidean distance in D-dimensional space. Once the cluster assignment is complete, the cluster prototypes are updated by computing the mean value of all the data points in the cluster assignment. Then, using these new cluster prototypes, the procedure is repeated. The stopping criterion is generally when no data points change clusters in successive iterations.

In this work, the K-means algorithm was performed four times with different random data points used for the K-means++ cluster initialization. The decision to use four starting points was based on the number of available independent CPU cores. The number of iterations was also capped at 750, although the cluster prototypes converged before that in all cases.

5.7 Skeletal Pose State Space Model

Ideally, the model for the hidden state variables would capture the skeletal pose precisely at a given instant in time. However, one limitation of the HMM is that the state space is finite, so there must be a finite number of hidden states. The approach taken in this work is to find a set of skeletal poses that suitably approximate the space of skeletons recorded by the Kinect sensor in the JHUMMA dataset. This was accomplished through unsupervised clustering, using the K-means algorithm described in Section 5.6, to find cluster prototypes given all of the skeletal frames in the training data of a given batch. The process was then repeated separately for each of the cross-validation datasets. The principle parameter involved with this method is the degree to which the training skeletons are quantized, which is the number of skeletal clusters, K. The hidden state variables H_t take on values $h_t \in \{1, ..., K\}$, which index the set of skeletal pose clusters.

The skeletal poses from the Kinect were adapted in three ways to simplify the problem and facilitate clusterings that capture the most important information. The first adaptation was to remove the translation of the hip joint from the features included in the skeletal clusters. As discussed in Section 3.3, this provides translation invariance, which is critical so that the pose clusters that are learned are applicable to any location in the dataset. It would be prohibitively expensive to produce and label a dataset extensive enough to support learning individual clusterings for different spatial areas.

The second adaptation was to remove the hand and feet joints from the skeletal poses. Studying the Kinect data in the JHUMMA dataset reveals that the hands and feet tend to be the noisiest joint estimates. The feet in particular tend to exhibit a significant amount of jitter from frame to frame. Removing these joints prevents the learned skeletal clusters from spending any modeling power accounting for the jitter in these noisy joints. It also has the added benefit of reducing the dimensionality of the skeletal pose features, which is also the dimension of the cluster space. Removing the hands and feet left only 16 joints in the abbreviated skeleton structure.

The third adaptation was to use the rotation representation of the skeletal pose, described in Section 3.3. This allows all of the training data, regardless of the actor, to be merged together. The skeletal poses are clustered in limb rotation space, which is more amenable to cross-training between actors than cartesian joint coordinates. The limb rotations are referenced to the vertical and only take on values in the range of 0 radians, which corresponds to straight up, to π radians, which corresponds to straight down. In this representation, the discontinuity between 0 radians and 2π radians is avoided, so the Euclidean distance remains a natural choice of metric. Applying all three of these adaptations resulted in each skeletal pose being represented by a 45-dimensional feature vector.

In order to explore the effect of different quantization levels in the pose space, the K-means clustering procedure was performed for various numbers of clusters on the first batch of data. Figure 5.2 shows the average joint error for each set of skeletal pose clusters. The error was calculated by computing the distance between each joint in each skeletal frame of the training data and the corresponding joint in the cluster mean that the training frame was associated with. For the purposes of computing the error, the rotation representation of the cluster mean was transformed back into cartesian coordinates. The error was summarized by averaging across each of the 16 joints in each of the $225,483$ training frames, which were pooled across all thirteen trials and 21 actions in the first batch of data.

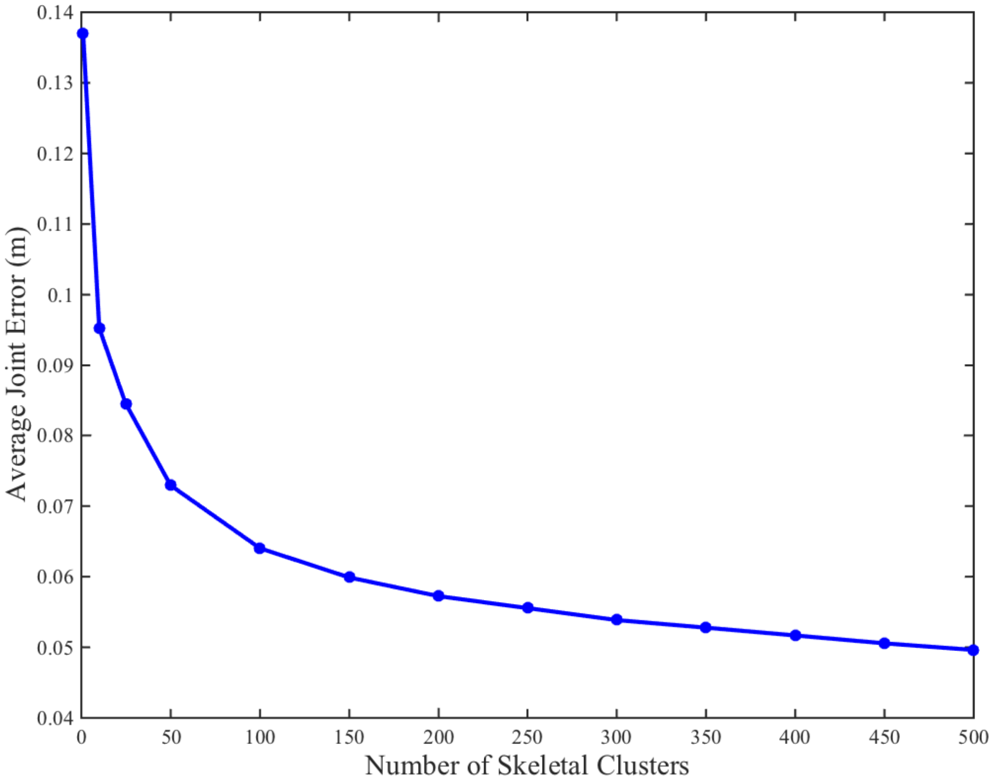


Figure 5.2: Comparison of the error between each skeletal frame in the training data and the associated skeletal cluster for various numbers of clusters.

The curve in Figure 5.2 illustrates a tradeoff between model complexity and accuracy. As the number of skeletal clusters increases, the clusters do a better job of approximating the training data, so the error decreases. However, more clusters require more model parameters to be estimated. Unless otherwise specified, the data shown in the following sections was generated using 200 skeletal clusters, which errs on the side of accurately modeling the skeletal poses with a more complex model. The effect of the number of clusters on the ability of the model to predict action sequences is explored more in Section 5.10.

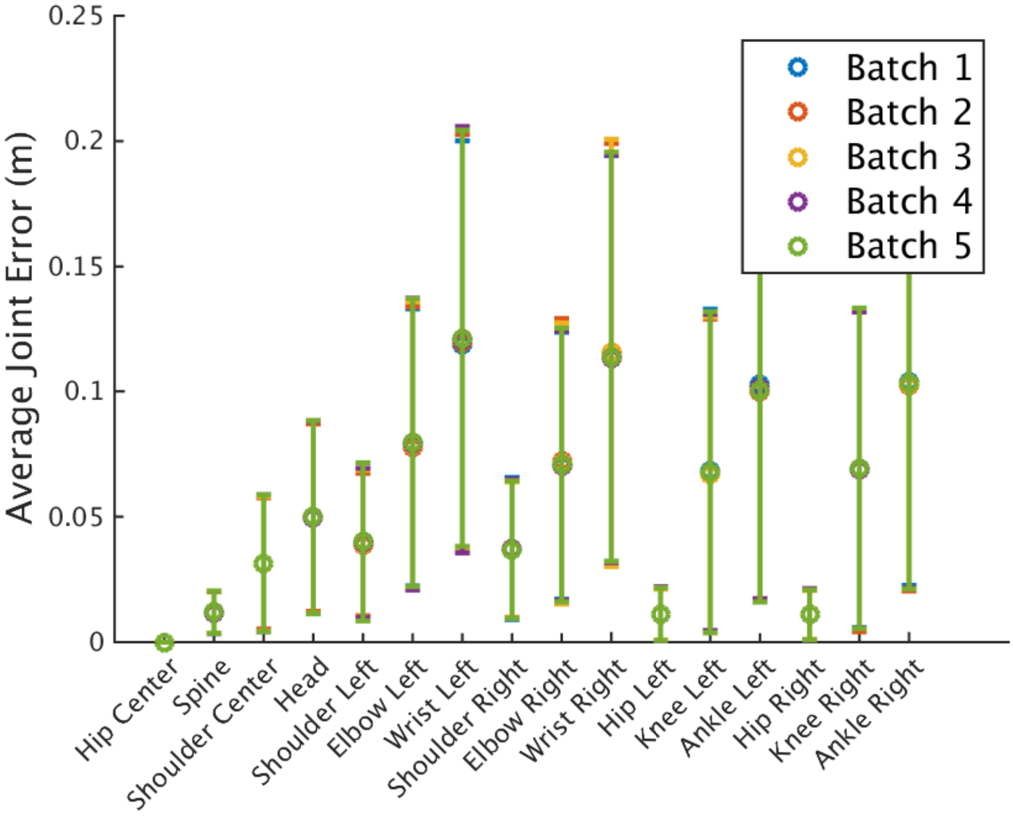


Figure 5.3: Comparison of the error between each skeletal joint in the training data and the closest skeletal cluster for each data batch. The number of skeletal clusters was fixed at 200.

In order to confirm that the training data in each cross-validation batch produces similar quantization results, the error of each joint labeled in Figure 3.9 was investigated. The error was computed as the Euclidean distance from each joint in the training data relative to the corresponding joint in the associated skeletal cluster. Figure 5.3 shows the error for each of the 16 joints, averaged across all of the training examples in each of the five batches. The error bars in Figure 5.3 correspond to one standard deviation of the joint errors.

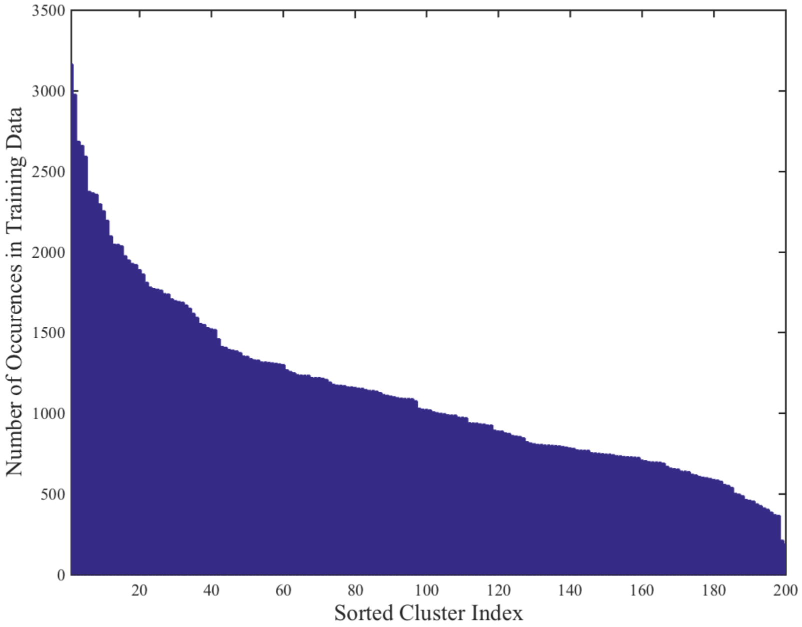


Figure 5.4: Histogram illustrating the number of occurrences of each skeletal cluster. The cluster indices have been sorted by their frequency.

As mentioned earlier, the hip translation was removed from the representation, so all of the hip joints were fixed to the origin when the other joint errors were computed, which is why they appear to have zero error. It is also interesting to note that the wrist and ankle joints have significantly higher error and variance than the others. This makes sense because they tend to move more during actions. They are also more likely to be tracked erroneously by the kinect. This result supports the decision to omit the hand and foot joints, which were even more unreliable.

Figure 5.4 shows the frequency of each skeletal pose cluster in the training data for the first cross-validation batch. The cluster indices are sorted according to their

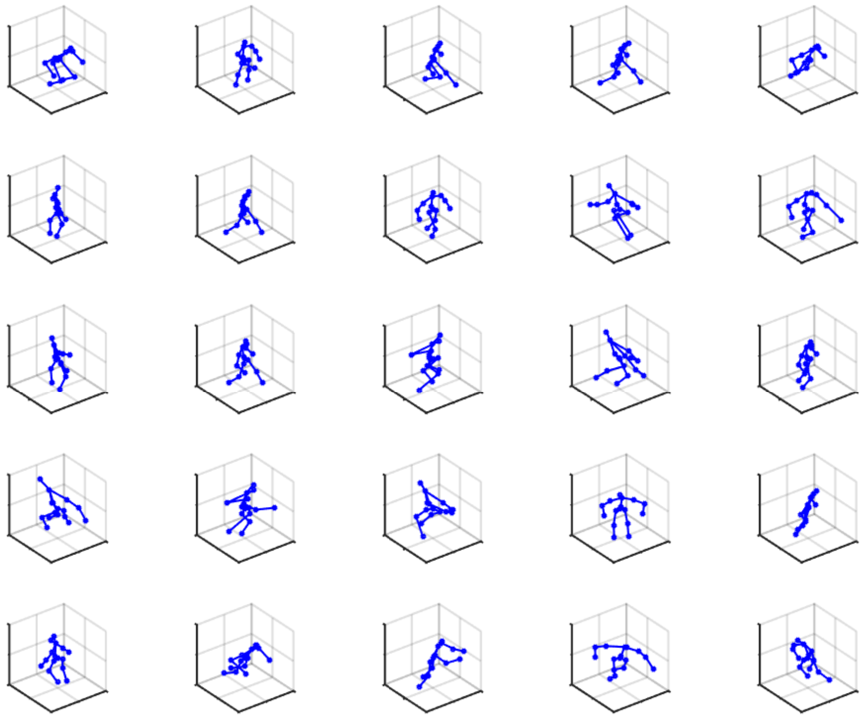


Figure 5.5: A random sampling of 25 of the 200 skeletal clusters learned from the first batch of training data.

frequency. Although the frequency is not uniform, the balance between cluster frequencies appears reasonable. Some actions have relatively unique skeletal poses that are not exhibited often, while many actions share similar skeletal poses that are clustered together and occur more frequently.

Figure 5.5 shows a random sampling of the skeletal pose clusters learned from the first batch of cross-validation data. These poses appear to be relatively diverse and interesting, indicating that the unsupervised clustering approach is at least reasonable.

5.8 Doppler Modulation Observation Model

While the hidden variables for each of the three HMM models can all utilize the same set of skeletal pose clusters, it is necessary to develop sets of spectrogram slice clusters that are tuned to each of the three ultrasound sensors individually.

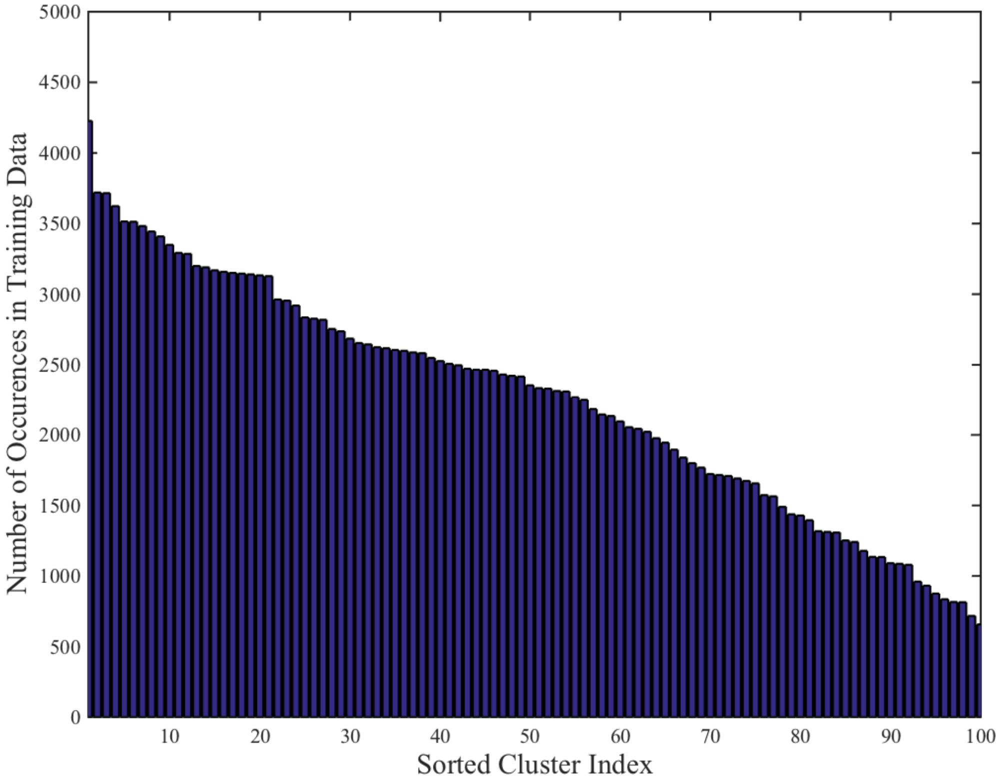


Figure 5.6: Histogram illustrating the number of occurrences of each 40kHz ultrasound cluster. The cluster indices have been sorted by their frequency.

An approach similar to the skeletal clustering was taken to quantize the spectrogram slices associated with each ultrasound sensor. The spectrogram slices from all of the training sequences were pooled together and the K-means algorithm was again used to choose a set of average clusters that were representative of the entire set. Although the Euclidean, or L_2, distance metric is not an obvious choice for comparing two spectrogram slices, empirical testing demonstrated almost no difference between the character or performance of spectrogram clusters created using the L_2 distance metric versus the L_1 distance metric. For consistency with the skeletal pose clustering, the spectrogram data presented here was generated using the Euclidean distance metric.

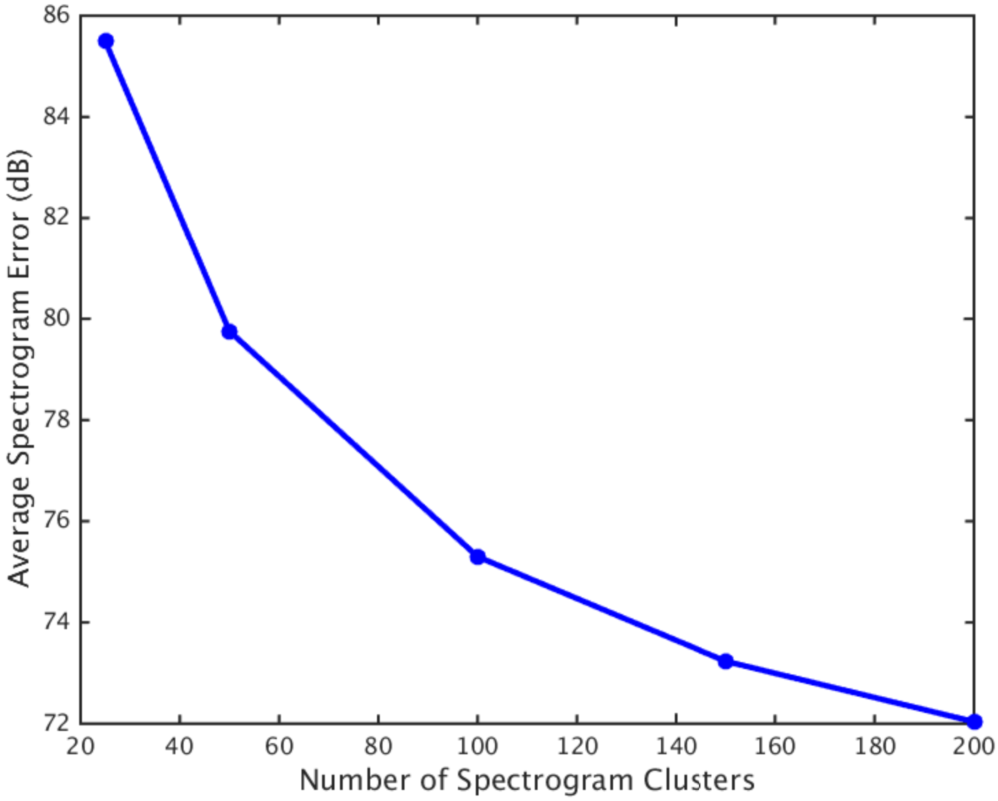


Figure 5.7: Comparison of the error between each 40kHz spectrogram slice in the training data and the associated spectrogram cluster for various numbers of clusters.

Figure 5.7 shows the average error for a spectrogram slice in the first batch of 40kHz ultrasound data over several values of K. For clustering the ultrasound spectrogram slices, a value of $K = 100$ was used. This process was repeated separately for the data from each of the three ultrasound sensors and for each of the cross-validation datasets.

Figure 5.6 shows the frequency of each 40kHz ultrasound cluster extracted from the first batch of cross-validation data. The cluster frequencies appear reasonable. Some are certainly more frequent than others, but no cluster dominates.

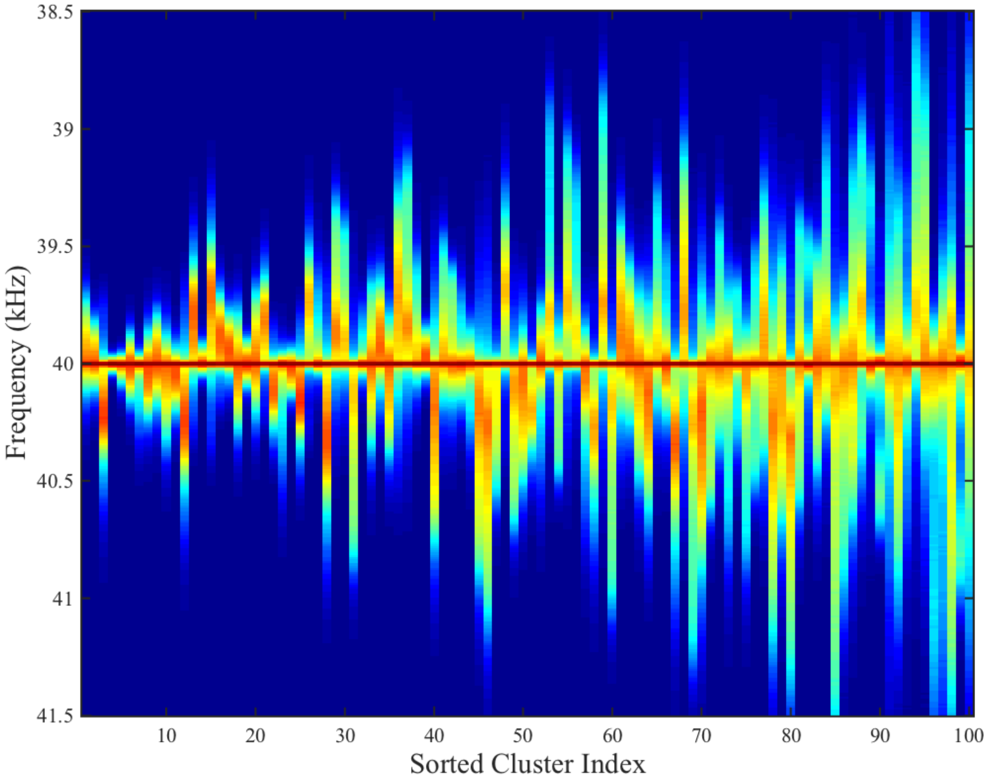


Figure 5.8: The collection of 40kHz ultrasound representative spectrogram slice cluster means, ordered by their frequency in the training data.

One nice feature of the spectrogram slices is that they are relatively easy to display and interpret in two-dimensions. Figure 5.8 shows all of the cluster means for the 40kHz training data. These representative spectrogram slices are sorted according to their frequency in the training data. Actions are still more often composed of periods of little movement, with large motions being relatively rare, which parallels the general trend of clusters with larger Doppler modulations being less frequent.

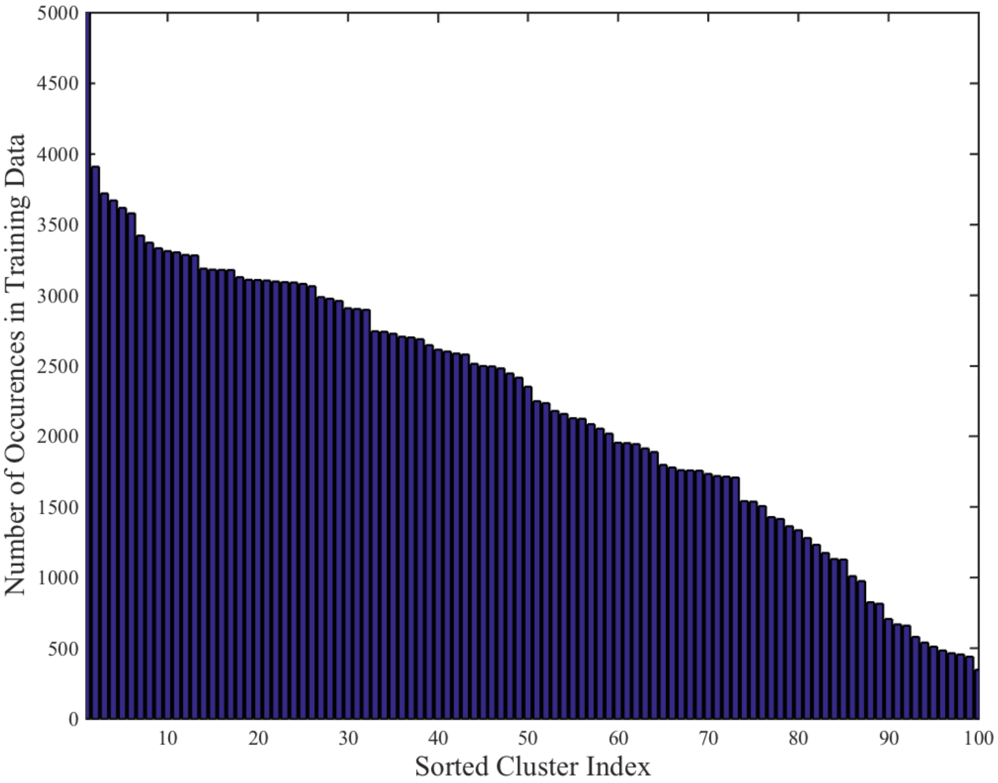


Figure 5.9: Histogram illustrating the number of occurrences of each 33kHz ultrasound cluster. The cluster indices have been sorted by their frequency.

Figure 5.9 shows the frequency of each 33kHz ultrasound cluster extracted from the first batch of cross-validation data. The distribution is slightly more skewed than the one for the 40kHz data.

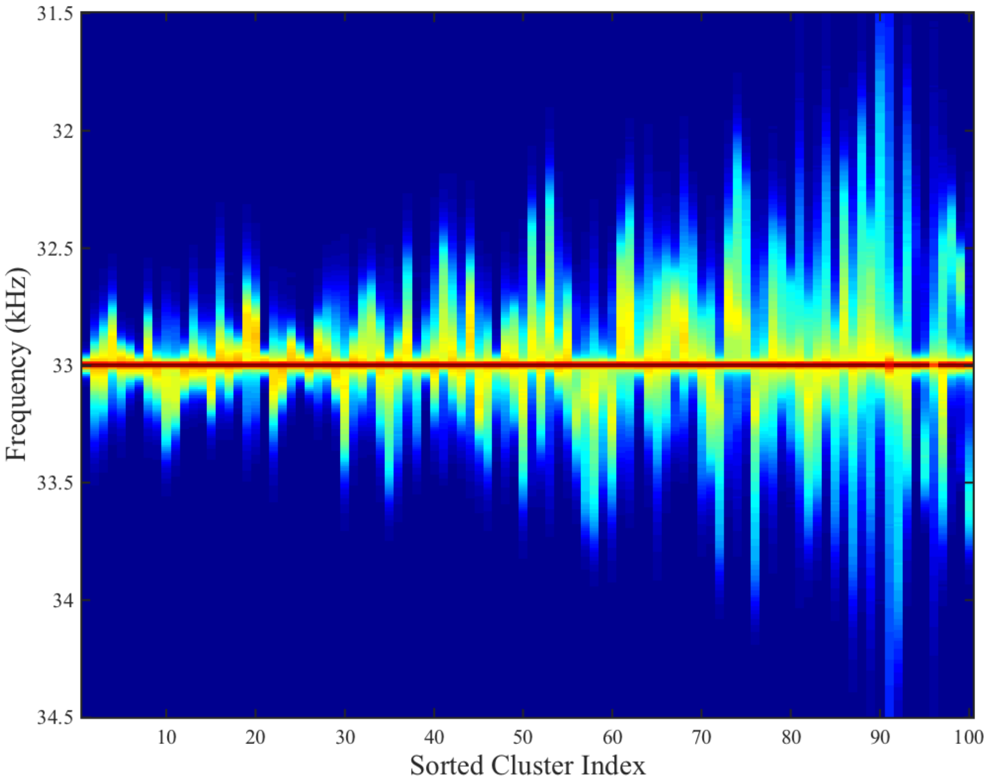


Figure 5.10: The collection of 33kHz ultrasound representative spectrogram slice cluster means, ordered by their frequency in the training data.

Figure 5.10 shows all of the cluster means for the 33kHz training data. The clusters are very similar in character to those culled from the 40kHz data. The modulations are smaller overall, but this is due to the lower carrier frequency and the fact that the sensor was positioned to the side of the majority of the actions in the JHUMMA dataset. The side sensors tended to observe smaller velocity components for the majority of actions. This is also supported by the histogram of the clusters, which indicates that the higher modulation clusters, indicative of more motion towards the side sensors, are less frequent compared to the histogram of the 40kHz sensor, which was positioned directly in front of most of the actions.

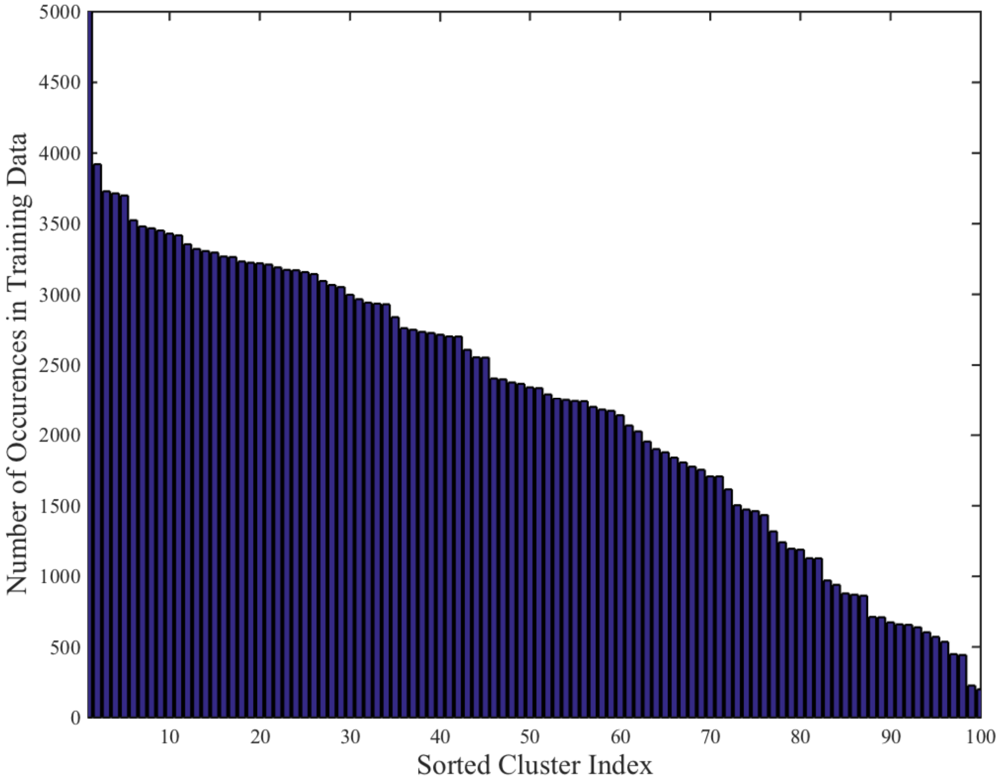


Figure 5.11: Histogram illustrating the number of occurrences of each 25kHz ultrasound cluster. The cluster indices have been sorted by their frequency.

Figure 5.11 shows the frequency of each 25kHz ultrasound cluster extracted from the first batch of cross-validation data. Similarly to the 33kHz spectrogram slice clusters, the 25kHz spectrogram slice clusters also appear to have a more skewed distribution than the 40kHz spectrogram slice clusters. This is in line with the less variable nature of both the positioning of the sensor off to the side and the lower magnitude of the 25kHz modulations. Figure 5.12 shows all of the cluster means for

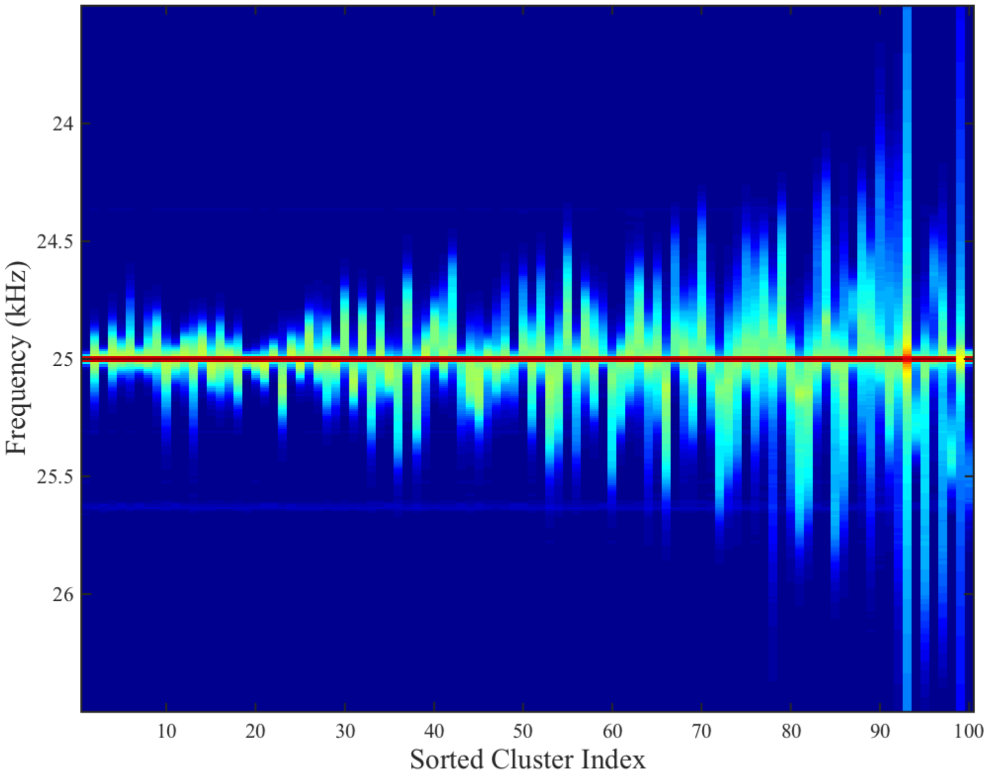


Figure 5.12: The collection of 25kHz ultrasound spectrogram slice cluster means, ordered by their frequency in the training data.

the 25kHz training data.

5.9 Classifying Human Action Sequences

Now that an appropriate vocabulary of skeletal pose prototypes has been constructed from the training data, and alphabets of spectrogram slice prototypes have been learned separately for each of the ultrasound frequencies, the parameters for each of the actions classes and ultrasound sensors can be computed using Equations 5.15, 5.16 and 5.17. To classify a novel test sequence from one of the ultrasound sensors, it must first be translated into a sequence of spectrogram slice prototypes. This is done by choosing the prototype with the smallest Euclidean distance from each spectrogram slice in the test sequence. Examples of a single test spectrogram for each action and the associated sequence of spectrogram slice prototypes are included in Section 5.11 in Figures 5.17 through 5.37.

Once the test data is translated into spectrogram prototypes $\mathbf{v}$, the most likely sequence of hidden skeletal pose prototypes $\mathbf{h}_a^*$ is computed using the Viterbi algorithm, described in Section 5.4. This procedure is repeated for each set of parameters $\boldsymbol{\theta}_a$. Note that only the parameters trained for that particular ultrasound frequency are considered and the subscript on the most likely hidden sequence is used to indicate the action the set of HMM parameters used to produce it was trained on.

The log-likelihood of a hidden sequence $\mathbf{h}$ and an observed sequence $\mathbf{v}$, normalized for the number of time steps in the sequences, is

$$\mathcal{L}_a(\mathbf{h}, \mathbf{v}) = \log \pi_a(h_0) + \sum_{t=1}^{T} \log A(h_{t-1}, h_t) + \sum_{t=1}^{T} \log B(v_t, h_t) - \log T. \tag{5.24}$$

After computing the log-likelihood of the hidden sequence produced by each action model, the sequence is classified as the action that best modeled the sequence. That is,

$$\hat{a} = \arg\max_a \mathcal{L}_a(\mathbf{h}, \mathbf{v}). \tag{5.25}$$

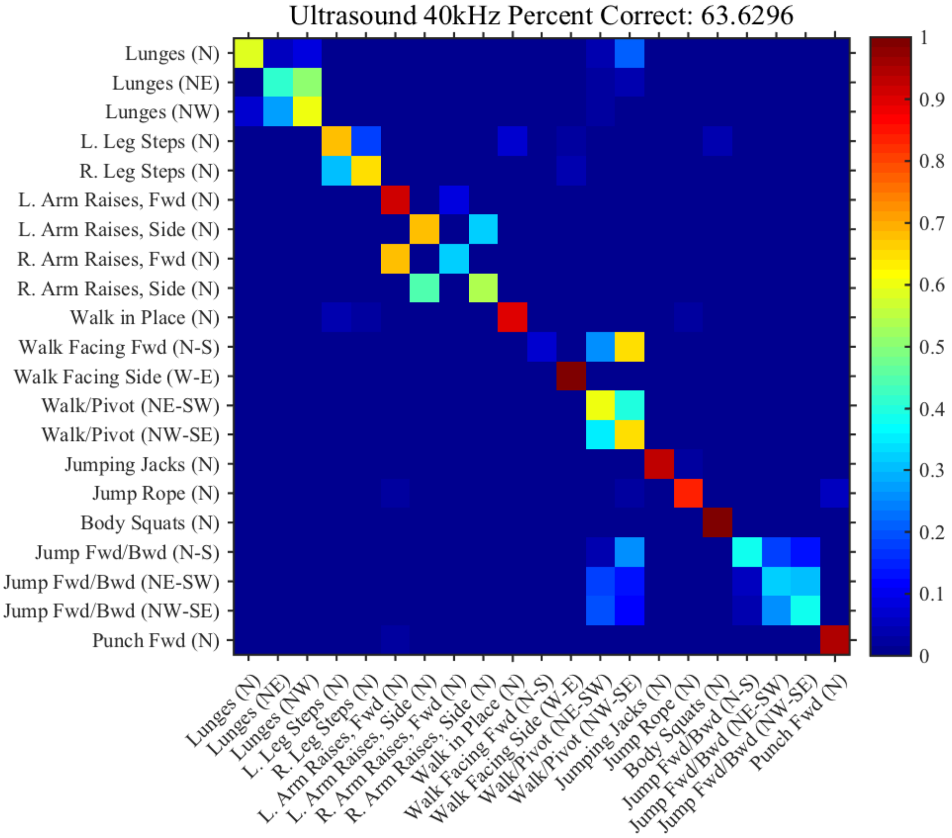


Figure 5.13: Confusion matrix enumerating the action classification decisions resulting from the 40kHz ultrasound model.

Figure 5.13 shows the confusion matrix for the action classification task that results from HMMs trained on the 40kHz ultrasound data. Overall, the HMM model correctly classified 63.63% of the 2700 test examples in the JHUMMA dataset. There were twenty-one actions, so classifying the actions by chance would yield a classification rate of under 5%. These results are compiled using all five of the cross-validation batches.

The confusion matrix indicates that the model tends to make very specific types of errors. It has significant difficulty distinguishing left versus right orientation among the action classes that have a similar action but different orientation. This is evident by the blocks of misclassification errors that are formed around many of the actions that have multiple orientations. One such example is the classification of the left leg steps and the right leg steps. The classifier places almost all of the probability mass on one of those two actions, but there is a lot of error between them.

Recall that the 40kHz ultrasound sensor was positioned to the north of the actor, which is roughly the line of symmetry for left versus right actions. Section 2.4 discussed the limitations of using micro-Doppler modulations to distinguish actions along this line of symmetry. With limited spatial information in the modulations, distinguishing between arm raises to one side or the other is difficult and results in significant classification errors.

On the other hand, the 40kHz ultrasound HMM does a good job of predicting actions with unique orientations such as punching and jumping jacks. This indicates that the modulations themselves are reasonable informative patterns to use for classifying coarse-grained action sequences.

Figure 5.14 shows the confusion matrix for the action classification task that results from HMMs trained on the 33kHz ultrasound data. Overall, the HMM model correctly classified 73.70% of the 2700 test examples in the JHUMMA dataset. Almost all of the actions with multiple orientations were symmetric with respect to the North to South axis of the JHUMMA setup. Therefore, it makes sense that the HMM trained on the micro-Doppler modulations recorded by the 33kHz ultrasound sensor,

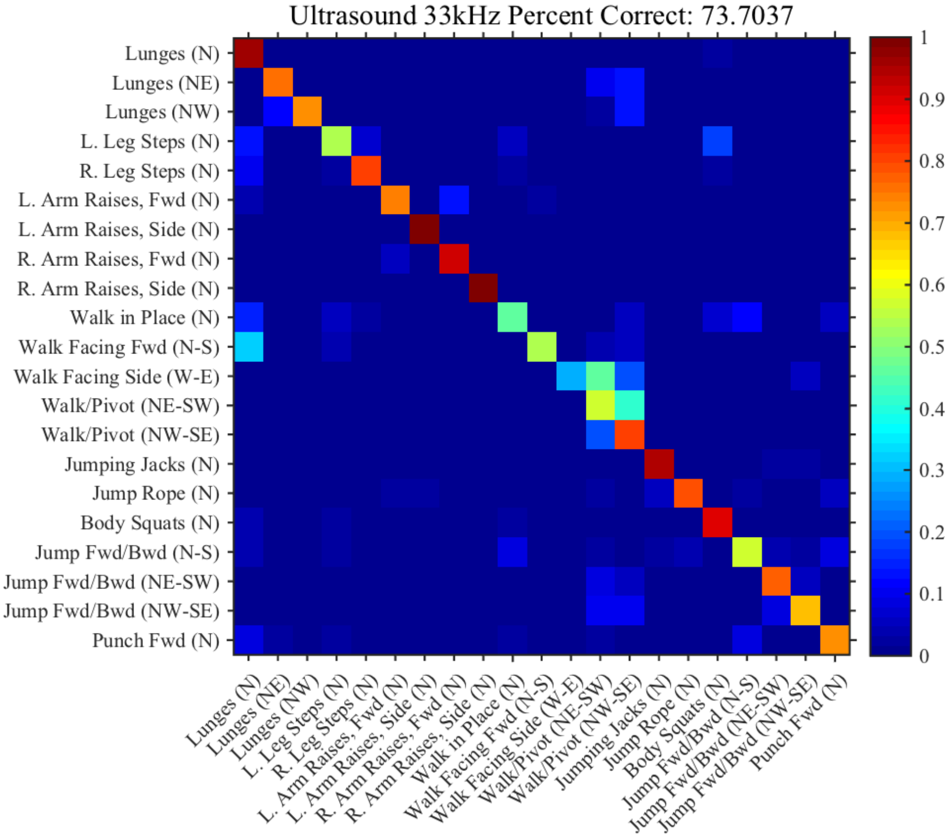


Figure 5.14: Confusion matrix enumerating the action classification decisions resulting from the 33kHz ultrasound model.

which was off to the West, made significantly fewer errors than the 40kHz ultrasound sensor. In fact, the one set of actions that did have some orientations facing the 33kHz sensor, walking back and forth, exhibited the same block error patterns in the confusion matrix as are evident in the 40kHz ultrasound classifications.

Figure 5.15 shows the confusion matrix for the action classification task that results from HMMs trained on the 25kHz ultrasound data. Overall, the HMM model correctly classified 75.30% of the 2700 test examples in the JHUMMA dataset. The errors made by the HMM model trained on data from the 25kHz ultrasound sensor,

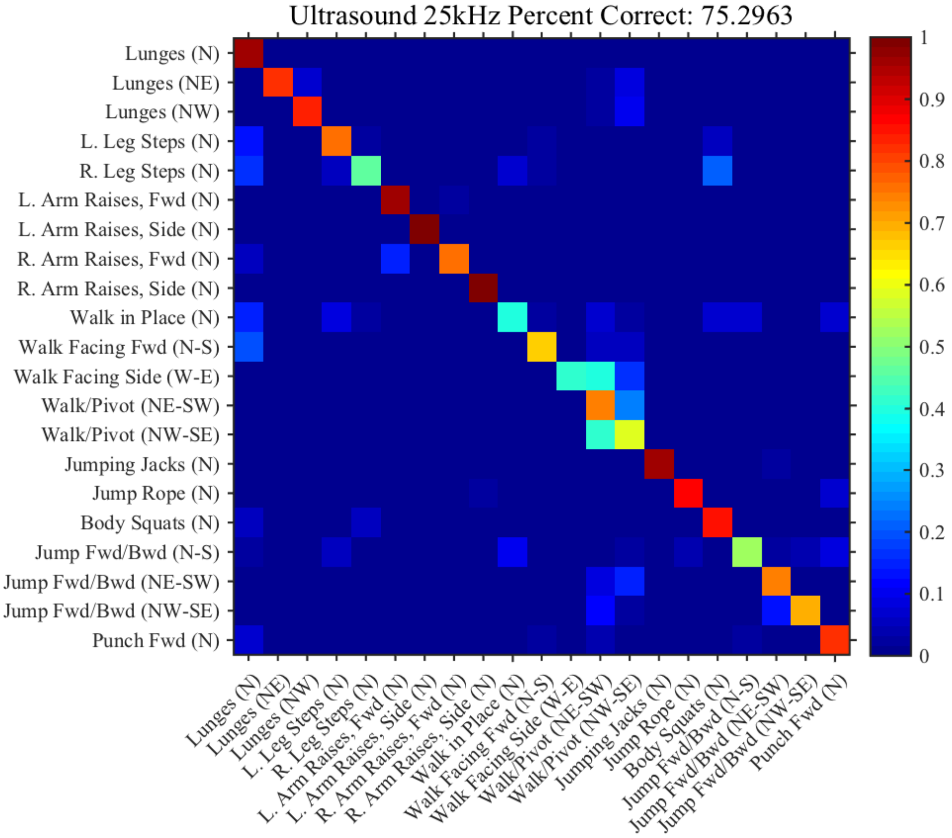


Figure 5.15: Confusion matrix enumerating the action classification decisions resulting from the 25kHz ultrasound model.

which was positioned to the East, is qualitatively similar to the errors made by the 33kHz ultrasound sensor. This is reasonable as both sensors were on the same cardinal axis and, therefore, encountered the same ambiguities due to the orientation of the actions in the JHUMMA dataset.

Given that the position of the ultrasound sensor has a significant effect on the classification accuracy of the model trained on data recorded by it, a fourth model that is a fusion of all three individual ultrasound HMMs was created to investigate the benefits of combining multiple ultrasound sensors to disambiguate the orientations of the actions. The combined model was constructed as a product of the individual models by summing the log-likelihoods for each action that were produced by the individual ultrasound models prior to choosing the most likely action. A test sequence $\mathbf{v}$ is now classified based on,

$$\hat{a} = \arg\max_{a} \Big(\mathcal{L}_a^{40}(\mathbf{h}, \mathbf{v}) + \mathcal{L}_a^{33}(\mathbf{h}, \mathbf{v}) + \mathcal{L}_a^{25}(\mathbf{h}, \mathbf{v}) \Big). \tag{5.26}$$

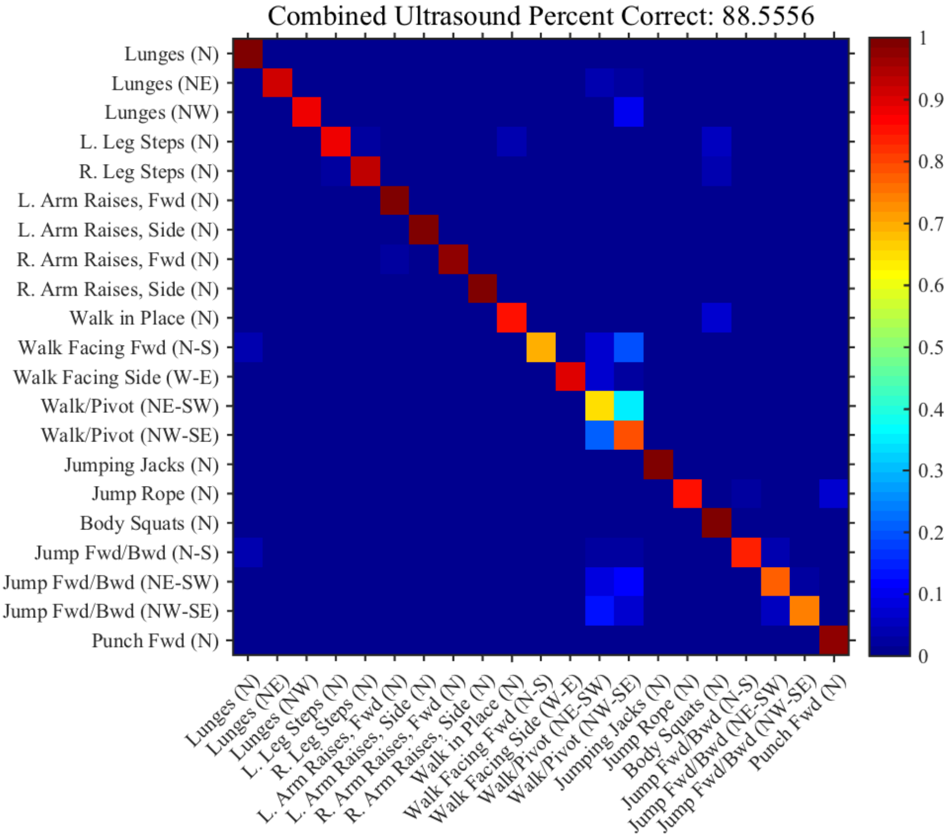


Figure 5.16: Confusion matrix enumerating the action classification decisions resulting from combining the output of the three ultrasound models as a product of experts.

Figure 5.16 shows the confusion matrix for the action classification task that results from combining each of the individual ultrasound HMM as a “product of experts” model. Overall, the HMM model correctly classified 88.56% of the 2700 test examples in the JHUMMA dataset. Combining the output of the individual HMM models gives a significant boost in classification performance and appears to be a reasonable approach to leveraging multiple ultrasound sensor units.

Table 5.2: Action classification performance of the HMM models on the JHUMMA dataset.

Action	25kHz	33kHz	40kHz	Combined	Examples
Lunges (N)	95.38	95.38	59.23	100.00	130
Lunges (NE)	81.54	75.38	40.77	91.54	130
Lunges (NW)	83.85	73.08	60.77	87.69	130
L. Leg Steps (N)	75.83	53.33	67.50	87.50	120
R. Leg Steps (N)	46.67	80.83	64.17	93.33	120
L. Arm Raises, Fwd (N)	96.15	73.85	90.77	100.00	130
L. Arm Raises, Side (N)	99.23	100.00	68.46	100.00	130
R. Arm Raises, Fwd (N)	76.15	91.54	31.54	96.92	130
R. Arm Raises, Side (N)	100.00	99.23	54.62	100.00	130
Walk in Place (N)	39.23	45.38	89.23	85.38	130
Walk Facing Fwd (N-S)	66.92	53.85	7.69	69.23	130
Walk Facing Side (W-E)	40.77	29.23	98.46	89.23	130
Walk/Pivot (NE-SW)	74.62	57.69	60.77	64.62	130
Walk/Pivot (NW-SE)	58.46	80.77	65.38	78.46	130
Jumping Jacks (N)	95.38	93.85	93.08	98.46	130
Jump Rope (N)	86.15	78.46	83.85	84.62	130
Body Squats (N)	84.62	89.23	100.00	100.00	130
Jump Fwd/Bwd (N-S)	53.08	56.92	37.69	83.85	130
Jump Fwd/Bwd (NE-SW)	73.85	77.69	32.31	76.92	130
Jump Fwd/Bwd (NW-SE)	70.00	68.46	37.69	74.62	130
Punch Fwd (N)	81.67	72.50	95.00	98.33	120
Overall	75.30	73.70	63.63	88.56	2700

Table 5.2 gives a more detailed breakdown of the exact classification rates for each of the three individual ultrasound models as well as the product of experts

model combining them all.

5.10 Classifying Human Body Poses

Table 5.3 presents a comparison of classification performance for several different numbers of skeleton pose cluster prototypes. On the left side, the overall classification results for the action sequences are shown. On the right side, the pose classification rate for the hidden sequence of cluster prototypes predicted from the data of each ultrasound band are shown. The pose classification rate is computed by comparing the closest skeletal pose prototype, at each time step in a test sequence, to the skeletal pose prototype predicted by the HMM given the test sequence of ultrasound modulations.

Table 5.3: Action and pose classification performance on the JHUMMA dataset.

	Action Classification			Pose Classification		
Number of Clusters	25kHz	33kHz	40kHz	25kHz	33kHz	40kHz
100	73.56	72.89	60.63	25.36	25.53	20.91
150	74.07	74.00	64.00	23.34	23.25	19.80
200	75.30	73.70	63.63	22.12	21.80	17.97
300	75.11	74.63	62.67	19.30	19.39	15.82

In general, more skeletal pose prototypes result in a more expressive state space that is able to model the actual recorded skeletal poses more closely. However, this precision comes at the price of a significantly larger model that now has many more parameters to estimate from the same fixed pool of training data. This is a classic model selection tradeoff and the results in Table 5.3 illustrate this. The action classification rates generally increase with the number of skeletal pose prototypes. However, the pose classification rates increase with fewer skeletal pose prototypes. This is reasonable because fewer prototypes make estimating the closest one significantly easier. Overall, using 200 skeletal pose prototypes, the conclusion drawn from the tradeoff in Figure 5.2, seems to be a reasonable compromise between these two trends.

5.11 Visualizing Ultrasound Sequences and Hallucinating Novel Sequences from Skeletal Poses

Figures 5.17 through 5.37 show several aspects of a single test sequence from each action class. Each column represents results derived from a single ultrasound sensor, and the time window from which the data was taken is identical across the sensors. The first row shows the spectrogram computed from recorded ultrasound data for each sensor. The second row shows the decomposition of the test sequence into the spectrogram cluster prototypes learned for that particular ultrasound bandwidth. This row is presented primarily to show the level of quality achieved by the vector quantization in the time-frequency domain. The third row shows an example of micro-Doppler modulations hallucinated by the corresponding HMM.

One of the attractive aspects of generative probabilistic models, such as the HMM, is their ability to hallucinate novel output sequences based on the parameters learned from the training data. These generated sequences can provide an important tool for understanding precisely what the model has captured from the training data. More importantly, the sequences can also provide important insights into the limitations of the generative model to capture certain aspects of the data.

The HMMs, developed in this chapter, turn out to be fairly unremarkable in their ability to generate convincing micro-Doppler modulations for human actions. Clearly they have culled enough information to be able to reliably estimate action sequences, but they are built using the simplest Markov dynamics, which fail to capture the long term correlations necessary to structure convincing hallucinations of human action. Each skeletal pose in the generated sequence is based purely from the transition statistics of the previous pose. This limited modeling power results in the jumpy structure in the modulations produced by the HMMs.

There are definitely certain aspects of the generated sequences that are correlated with the actions. The overall strength of the modulations tends to change appropriately from one action relative to another. However, without the longterm temporal correlations to structure the hallucinated modulation, it is difficult to recognize the result.

In order to improve these HMM models, it is possible to build more complex states that include dependencies on the previous two or even three poses. Unfortunately, the hidden state space is already quite large, so this approach does not seem practical. Additionally, the HMM generates the observed spectrogram slice based purely on statistical parameters culled from the training data. The physics behind generating micro-Doppler modulations is well understood. In Chapter 6, a more sophisticated forward model that incorporates the physics of the Doppler phenomenon will be developed. In Chapter 7, a neural network, specifically tailored to model more longterm temporal correlations, is developed. Incorporating aspects from both of these models may allow for improved generation of novel micro-Doppler modulations of human action.

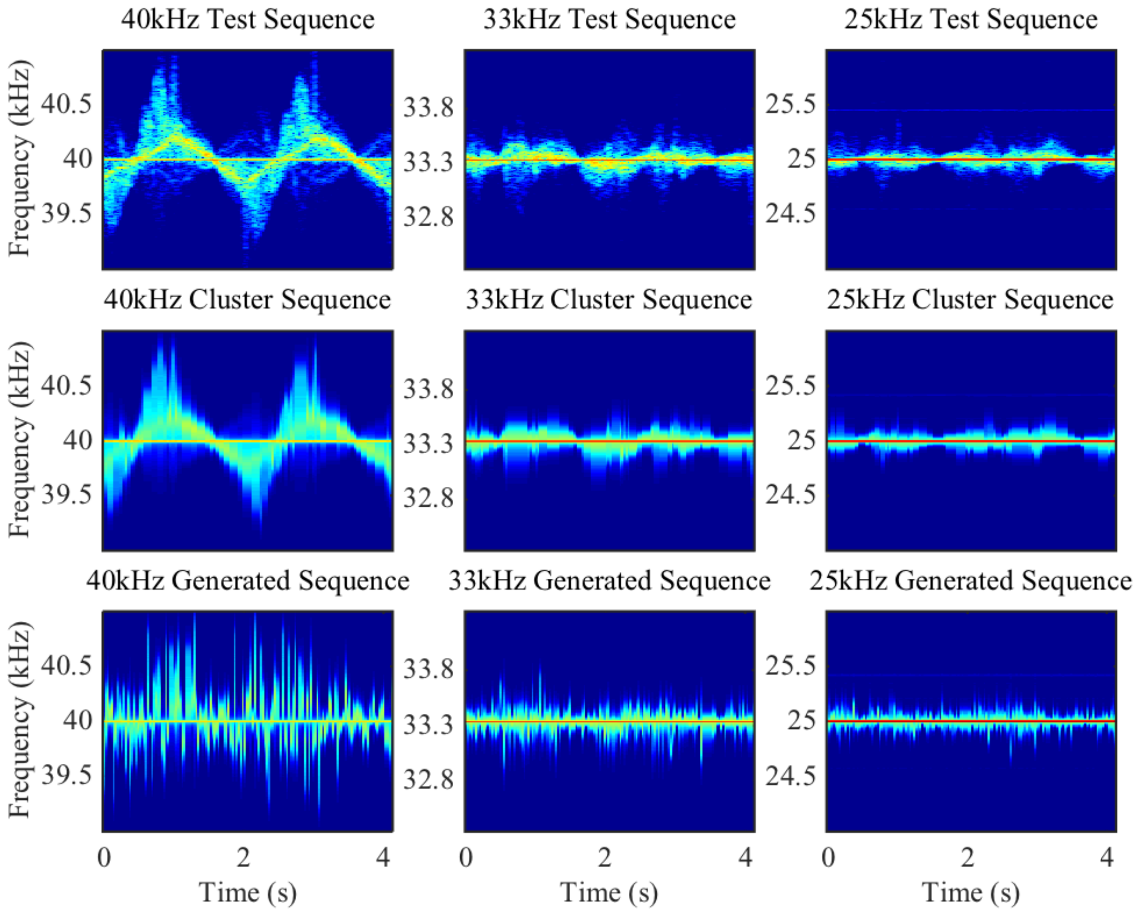


Figure 5.17: An example of a test data sequence for the lunge (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

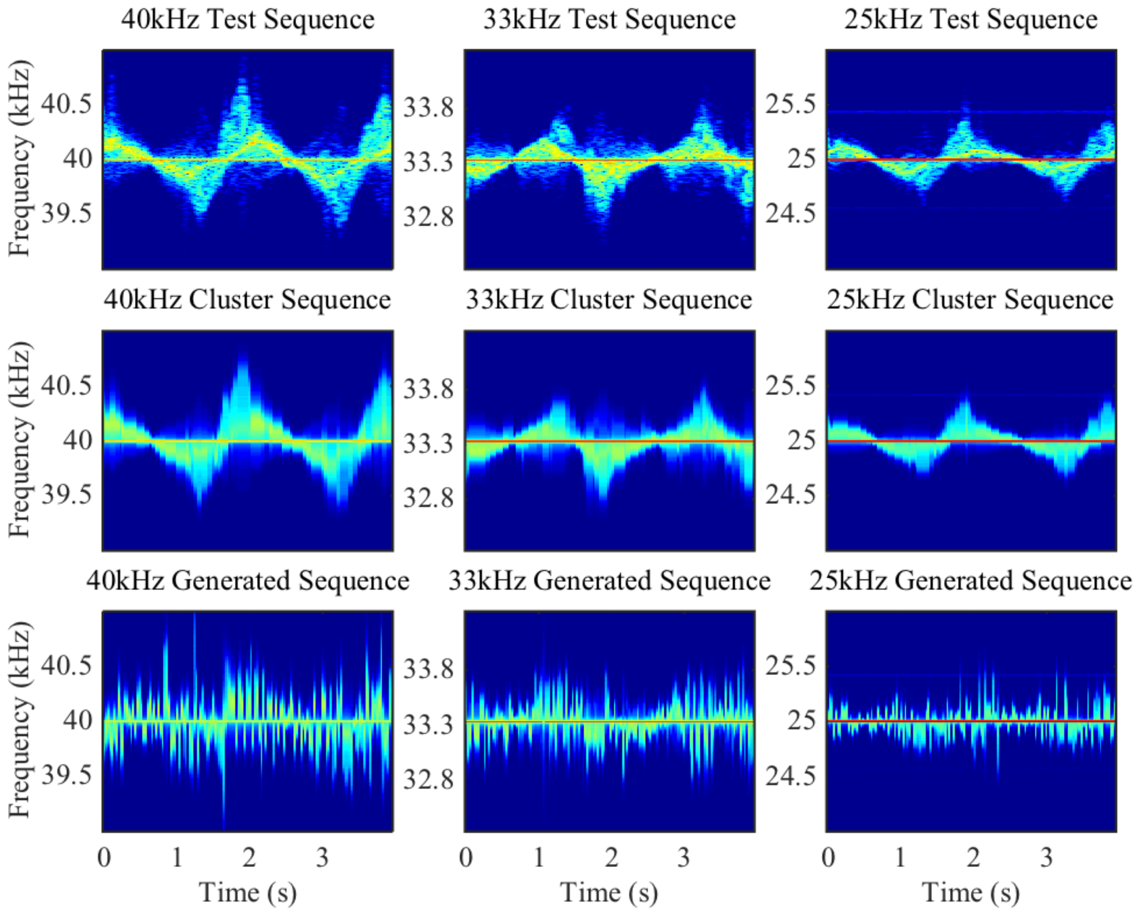


Figure 5.18: An example of a test data sequence for the lunge (NE) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

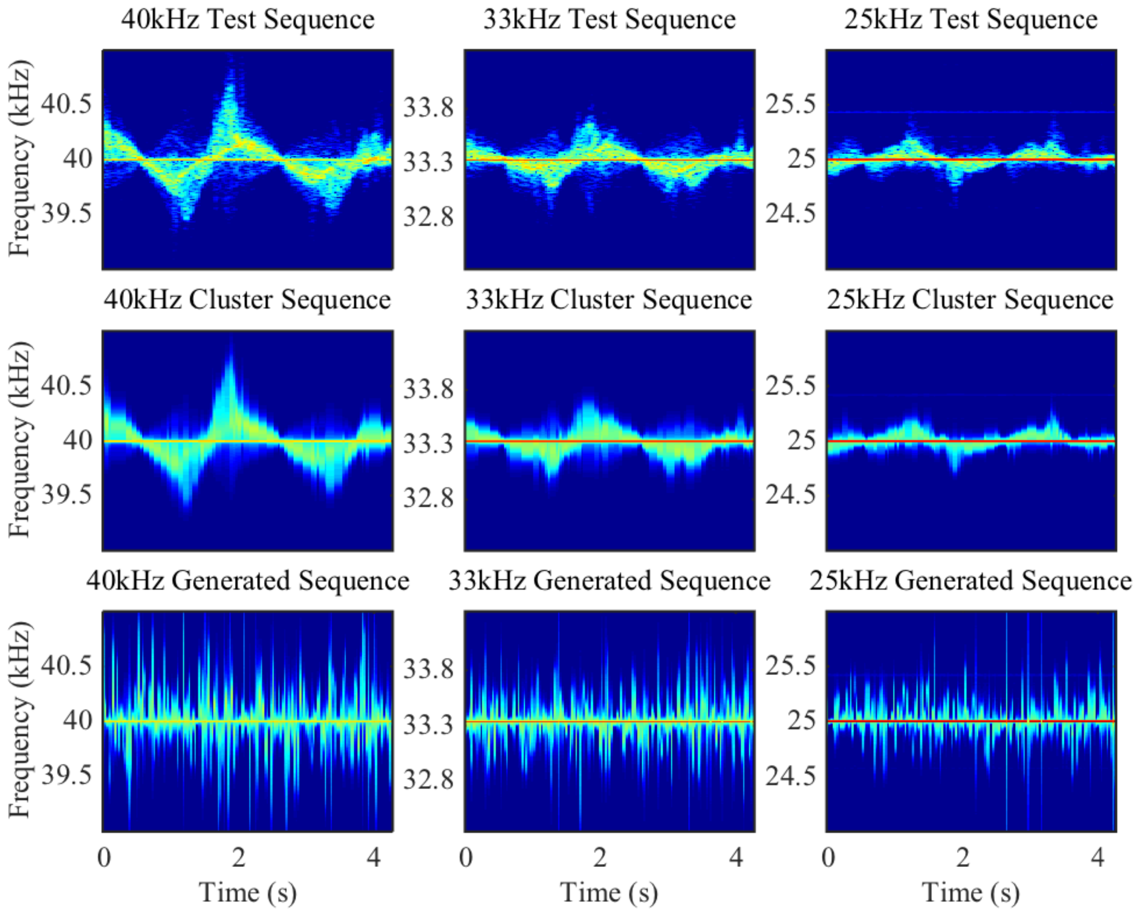


Figure 5.19: An example of a test data sequence for the lunge (NW) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

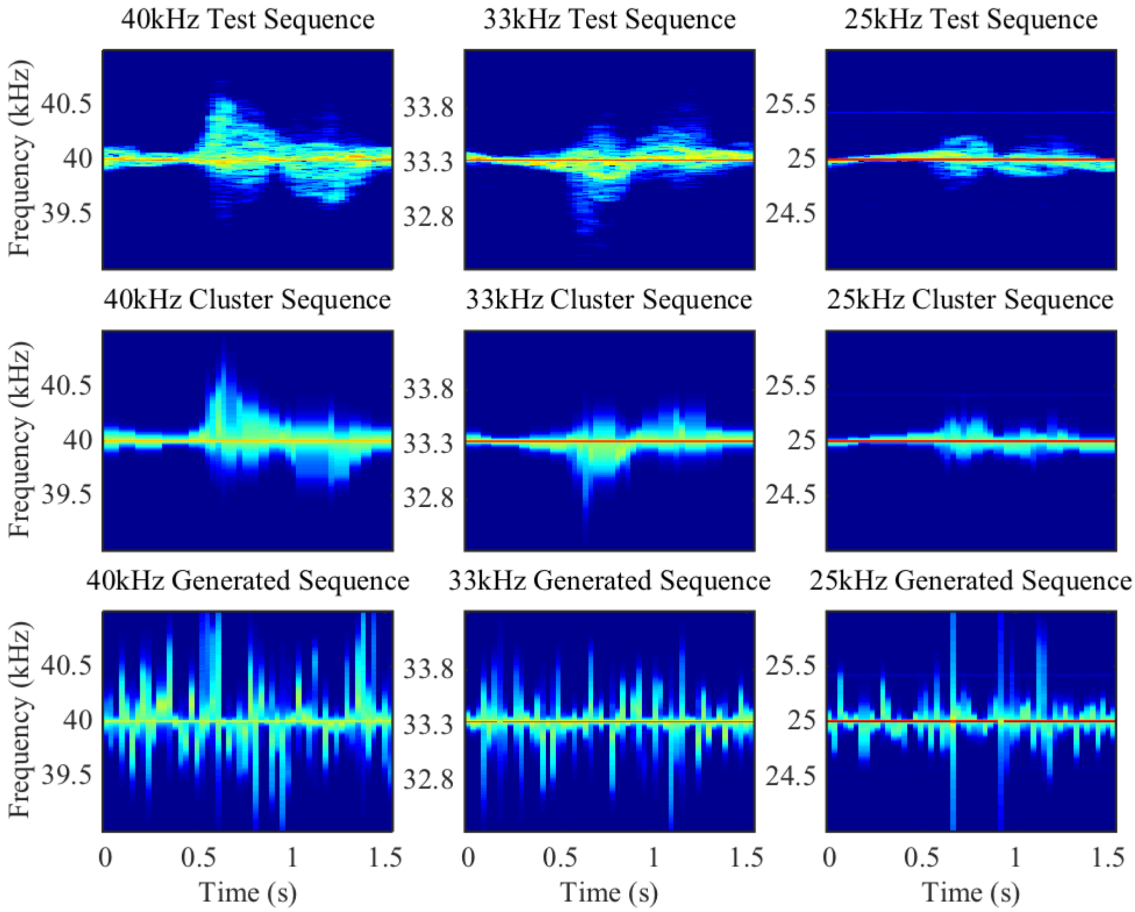


Figure 5.20: An example of a test data sequence for the left leg step (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

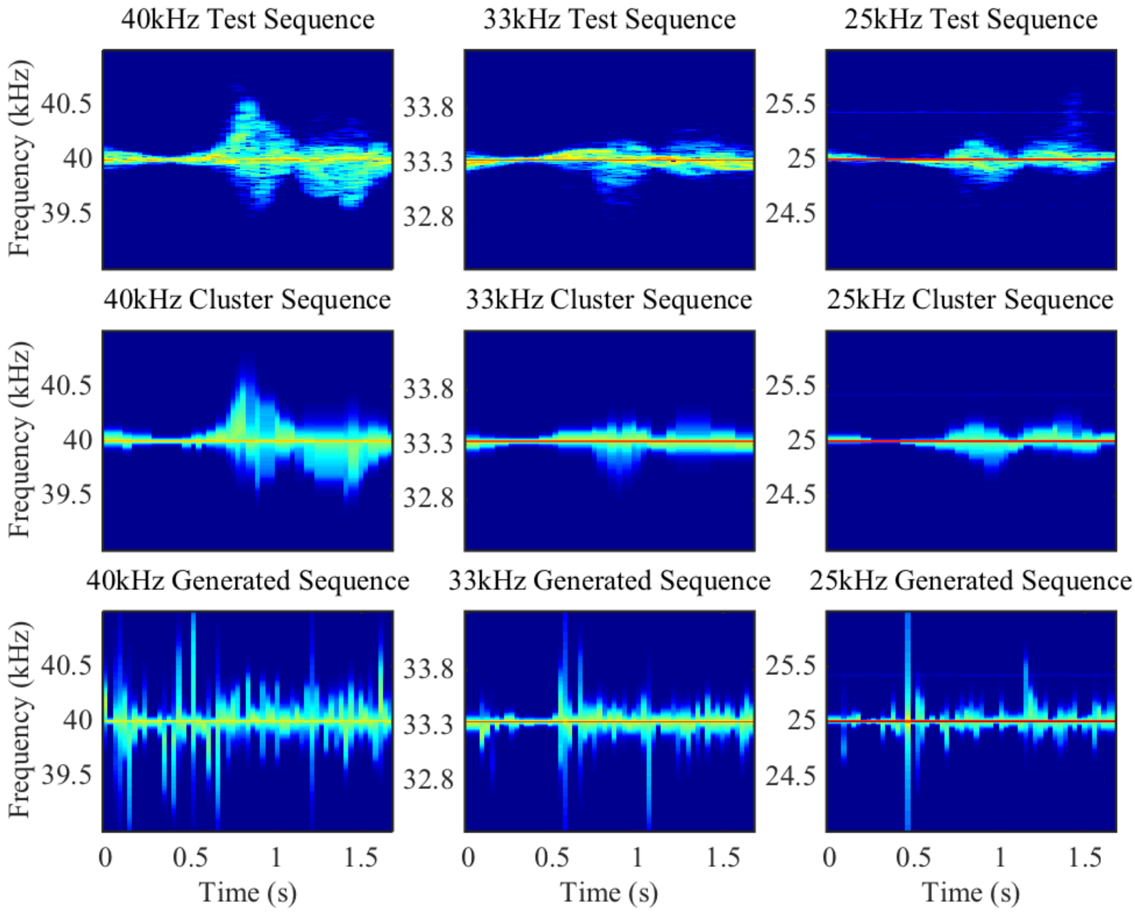


Figure 5.21: An example of a test data sequence for the right leg step (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

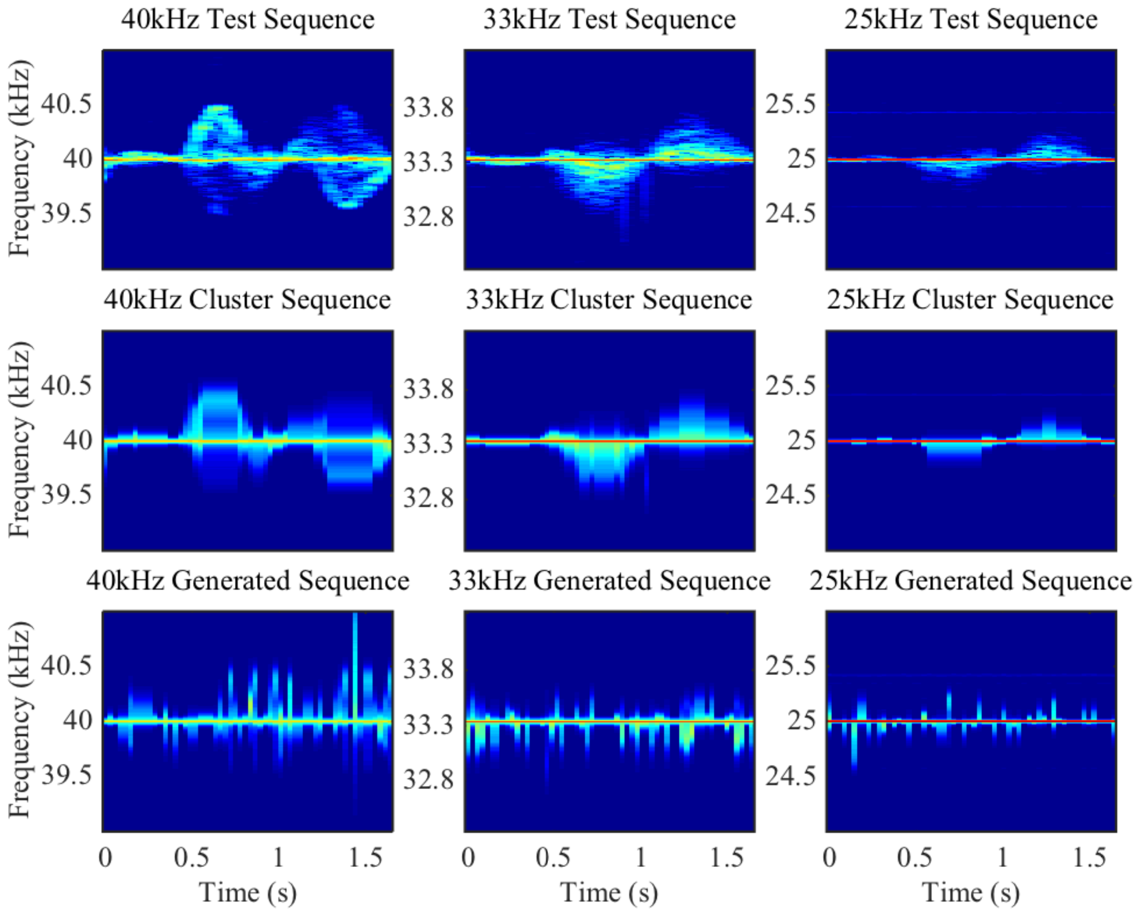


Figure 5.22: An example of a test data sequence for the left forward arm raise (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

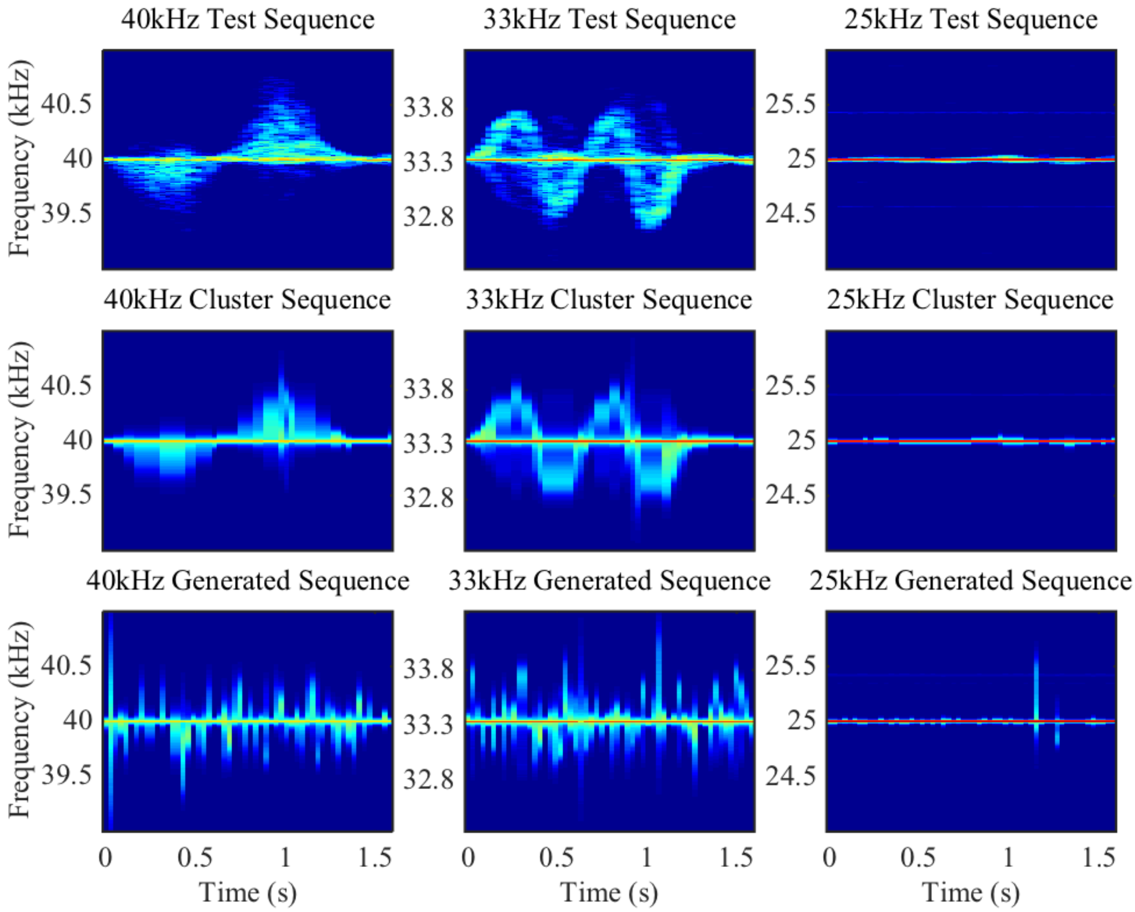


Figure 5.23: An example of a test data sequence for the left side arm raise (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

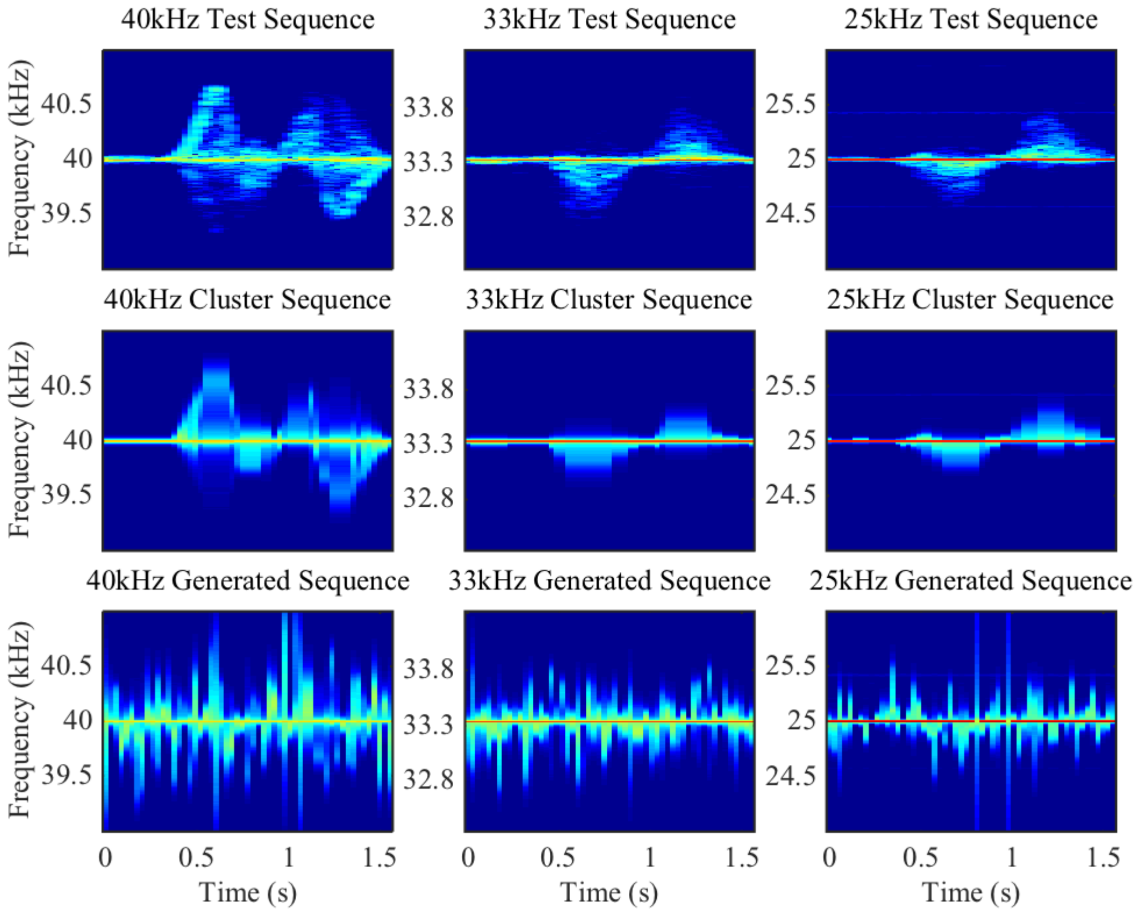


Figure 5.24: An example of a test data sequence for the right forward arm raise (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

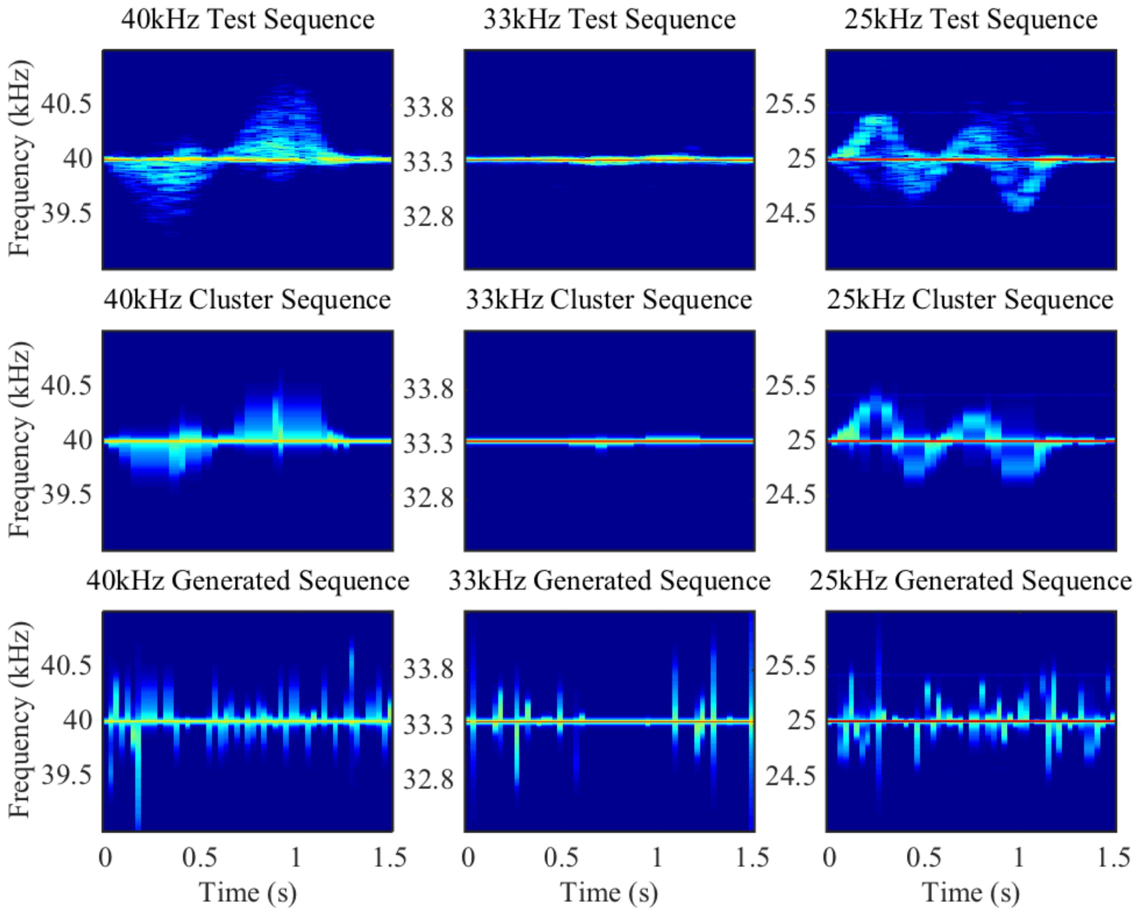


Figure 5.25: An example of a test data sequence for the right side arm raise (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

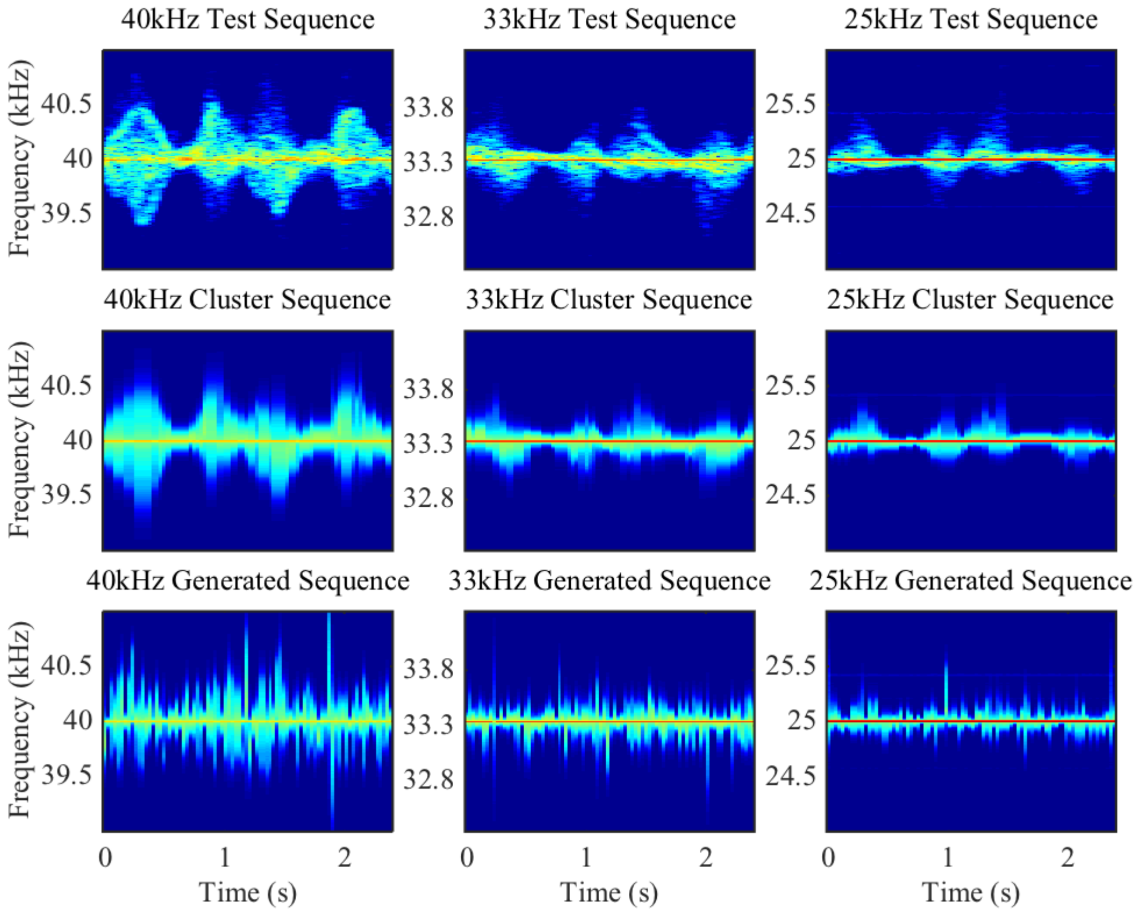


Figure 5.26: An example of a test data sequence for the walk in place (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

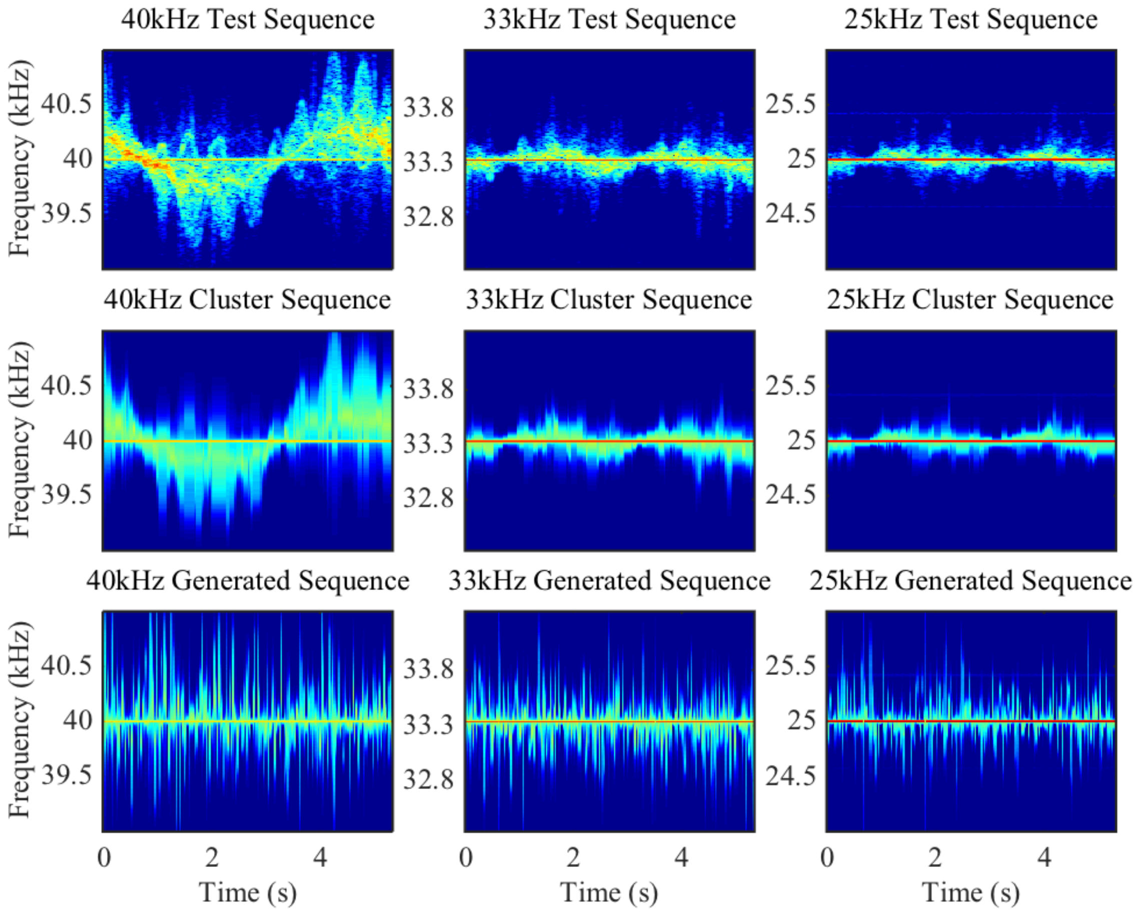


Figure 5.27: An example of a test data sequence for the walk facing forward (N-S) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

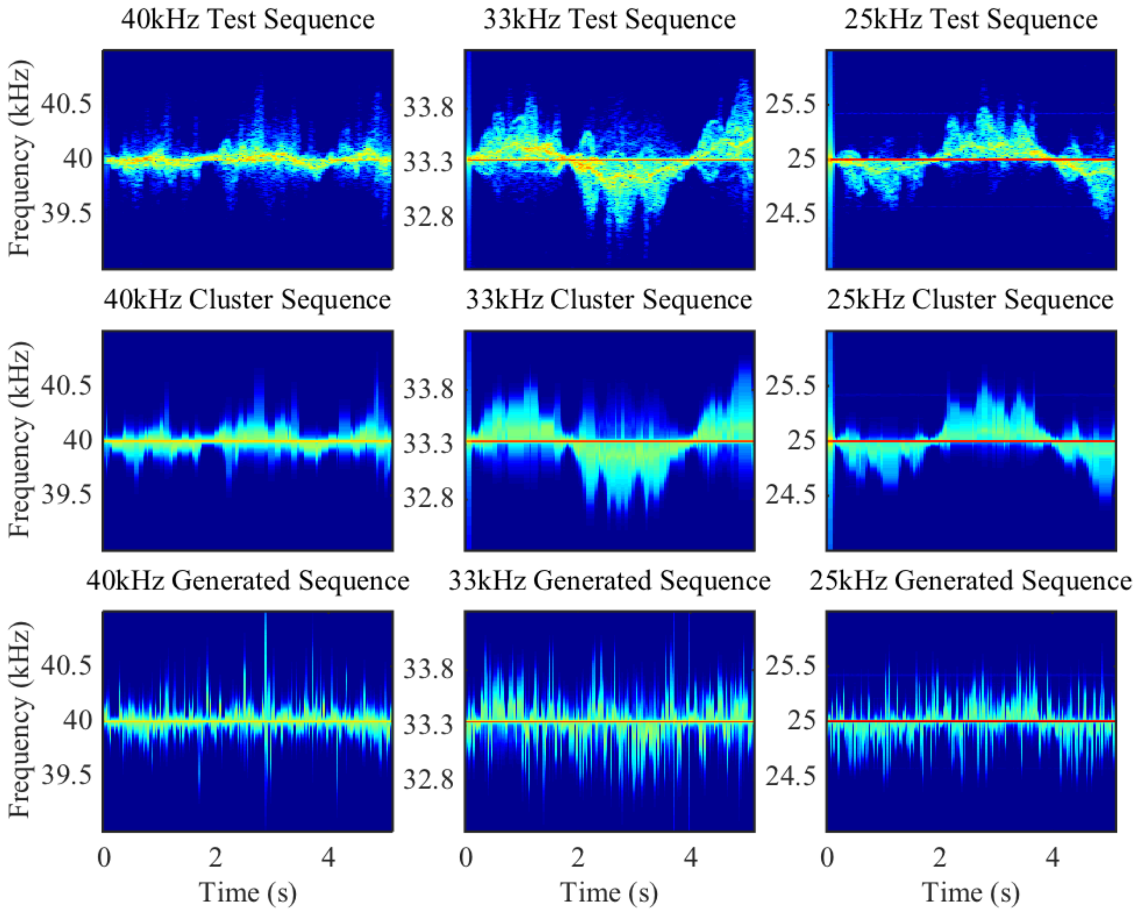


Figure 5.28: An example of a test data sequence for the walk facing side (W-E) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

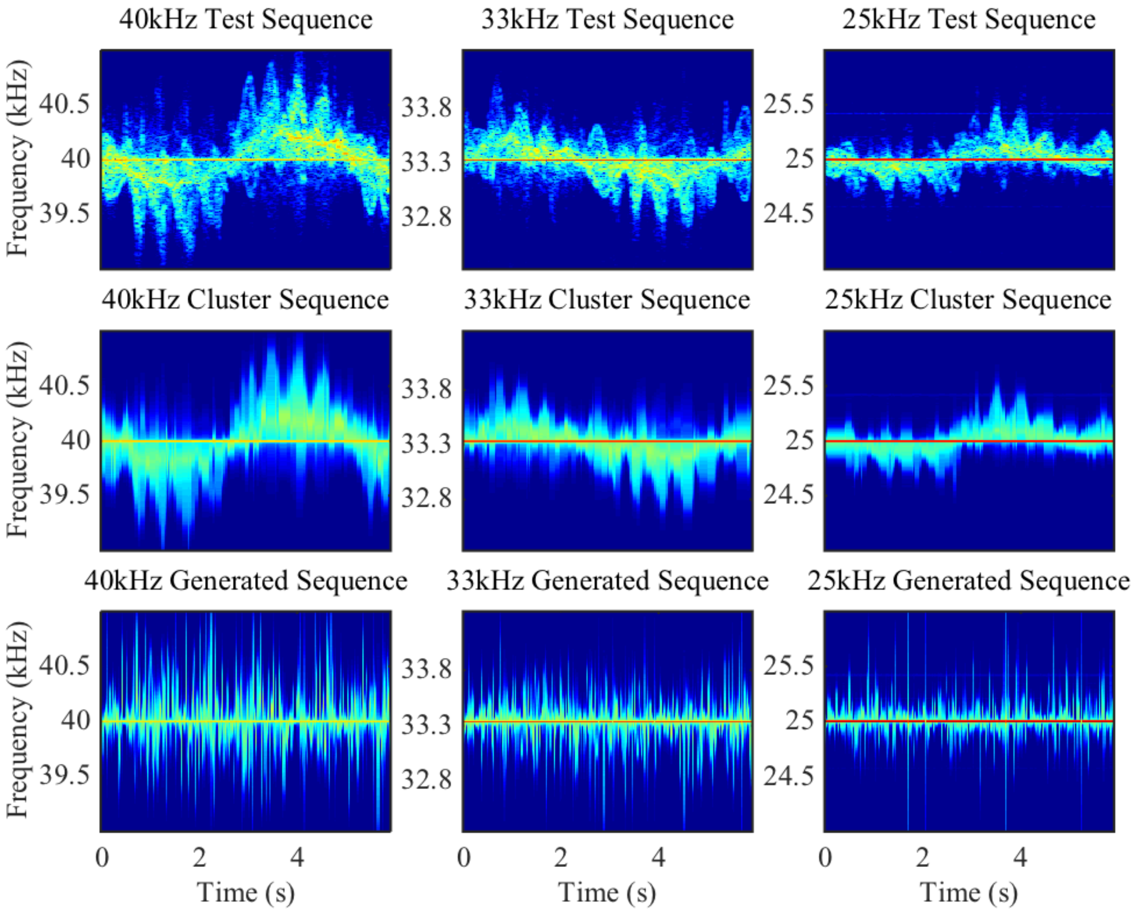


Figure 5.29: An example of a test data sequence for the walk and pivot (NE-SW) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

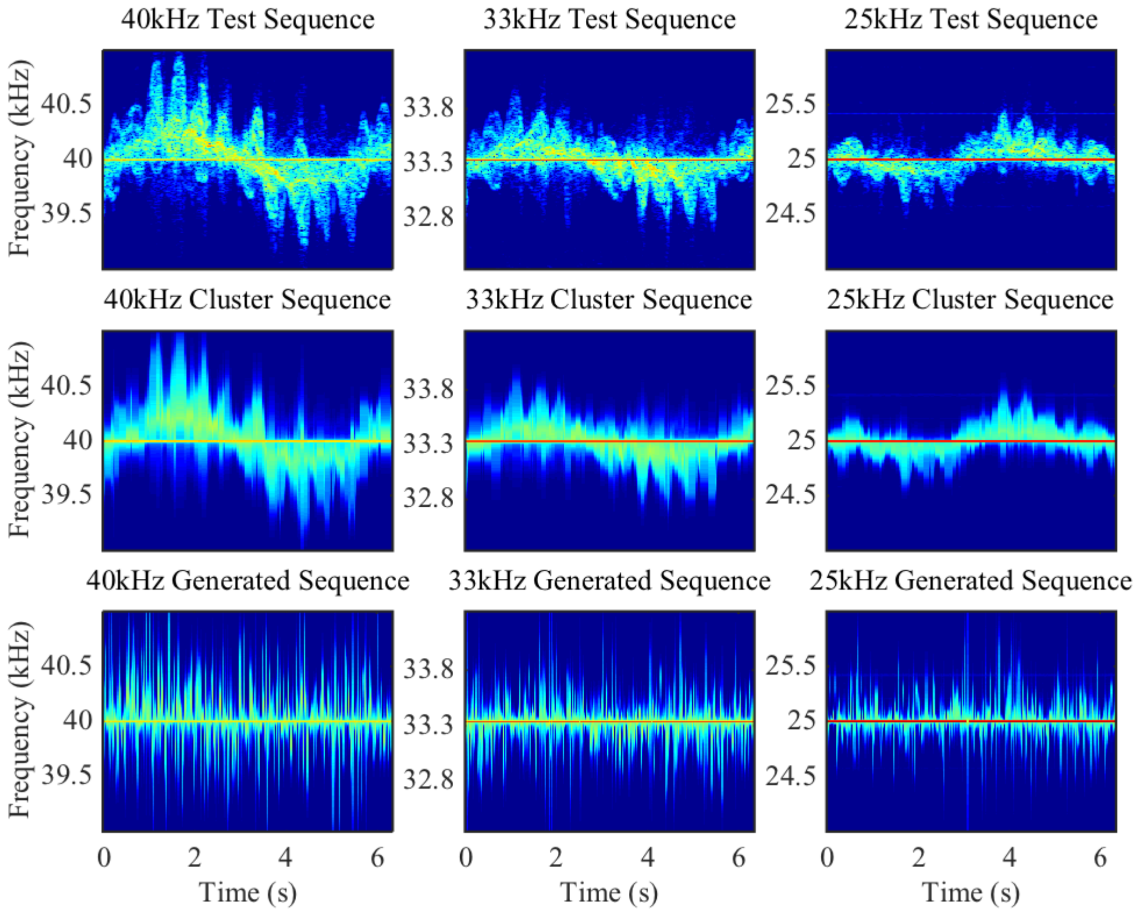


Figure 5.30: An example of a test data sequence for the walk and pivot (NW-SE) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

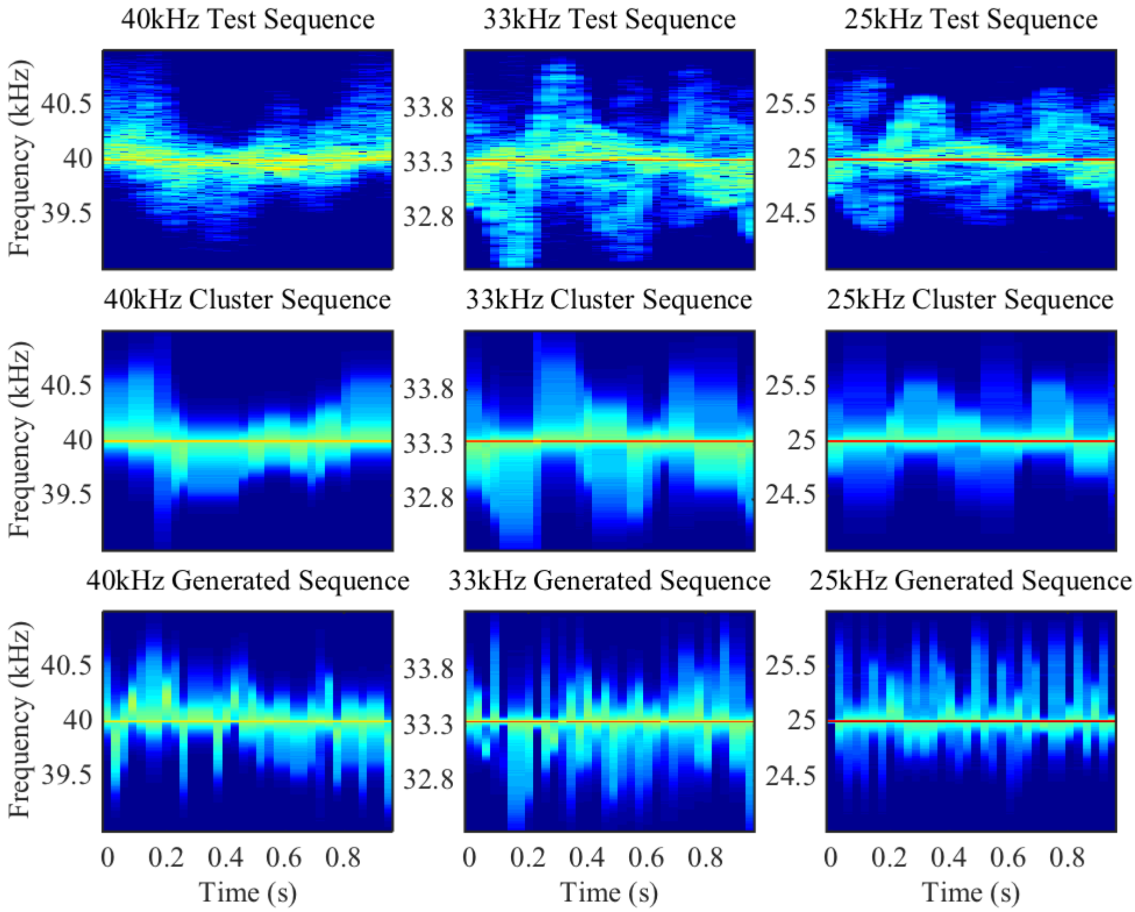


Figure 5.31: An example of a test data sequence for the jumping jacks (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

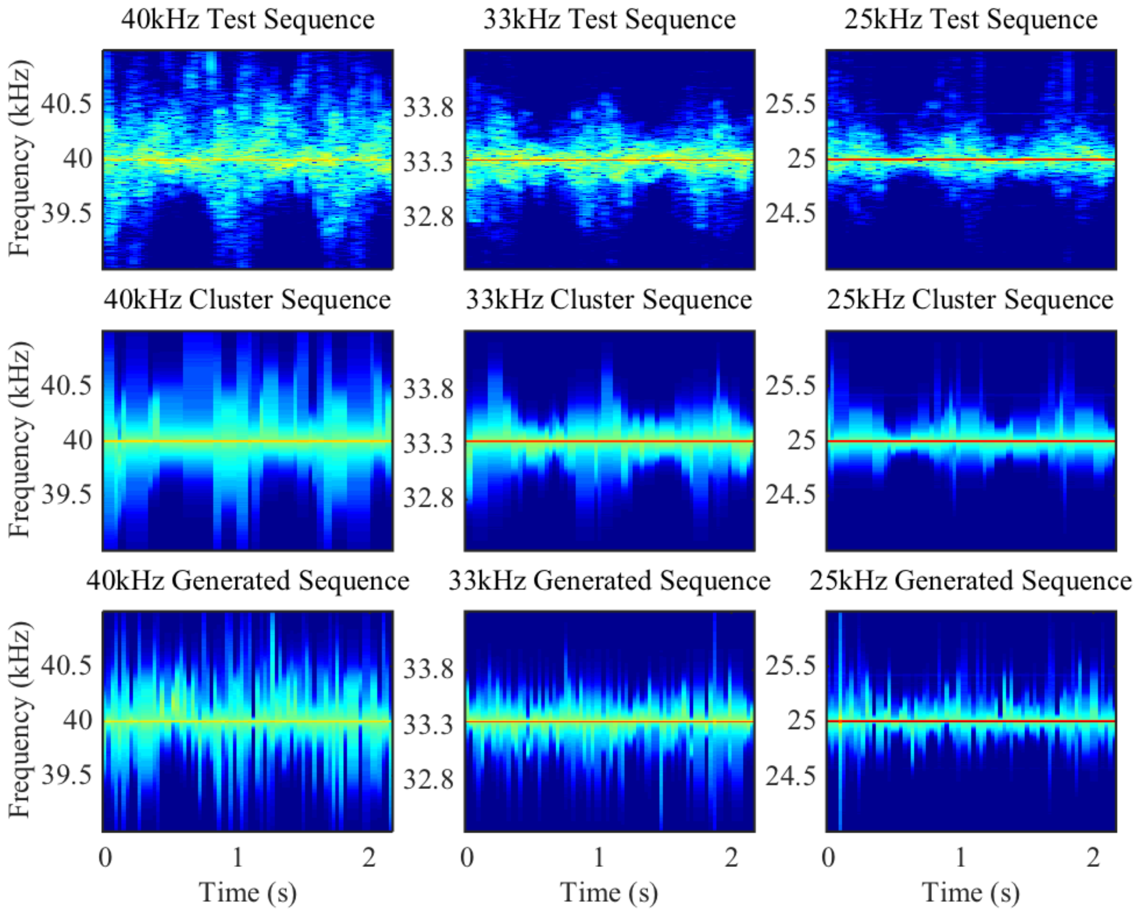


Figure 5.32: An example of a test data sequence for the jump rope (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

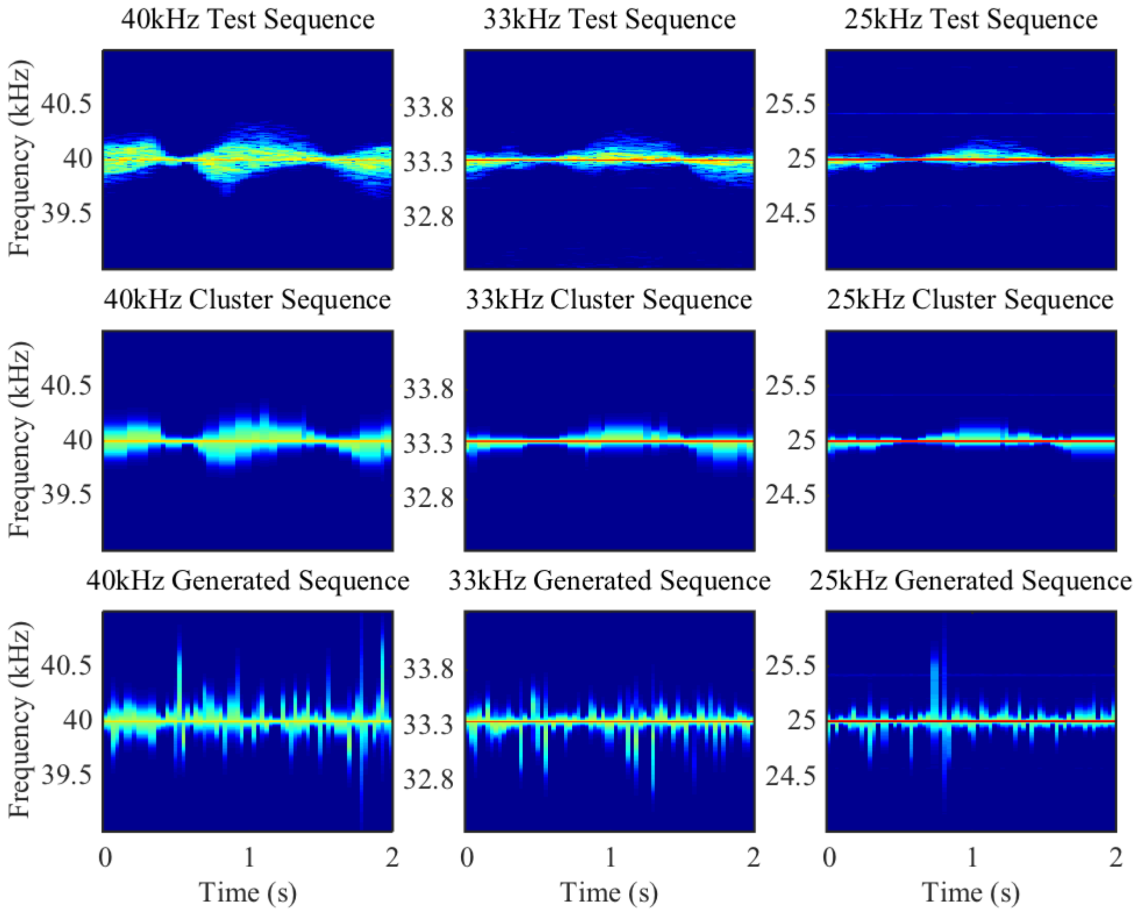


Figure 5.33: An example of a test data sequence for the body squats (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

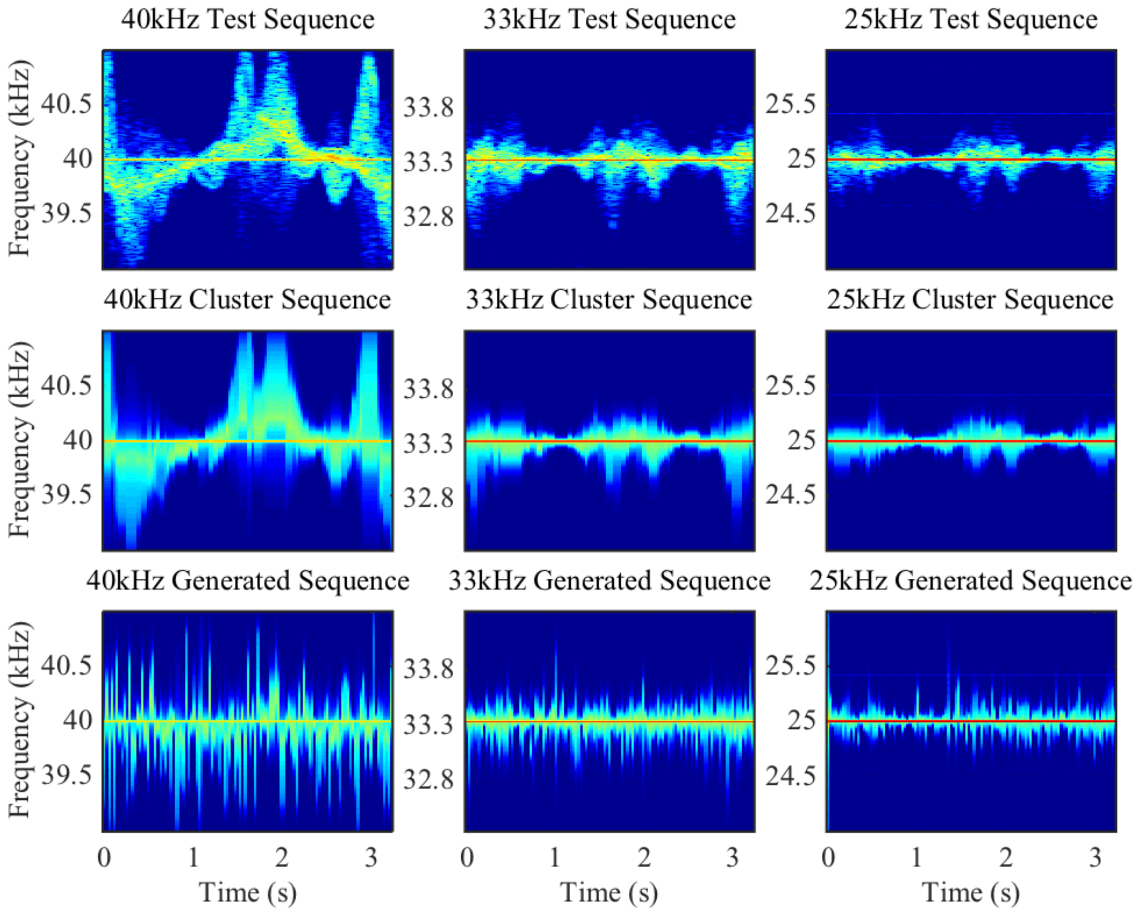


Figure 5.34: An example of a test data sequence for the jump forward and backward (N-S) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

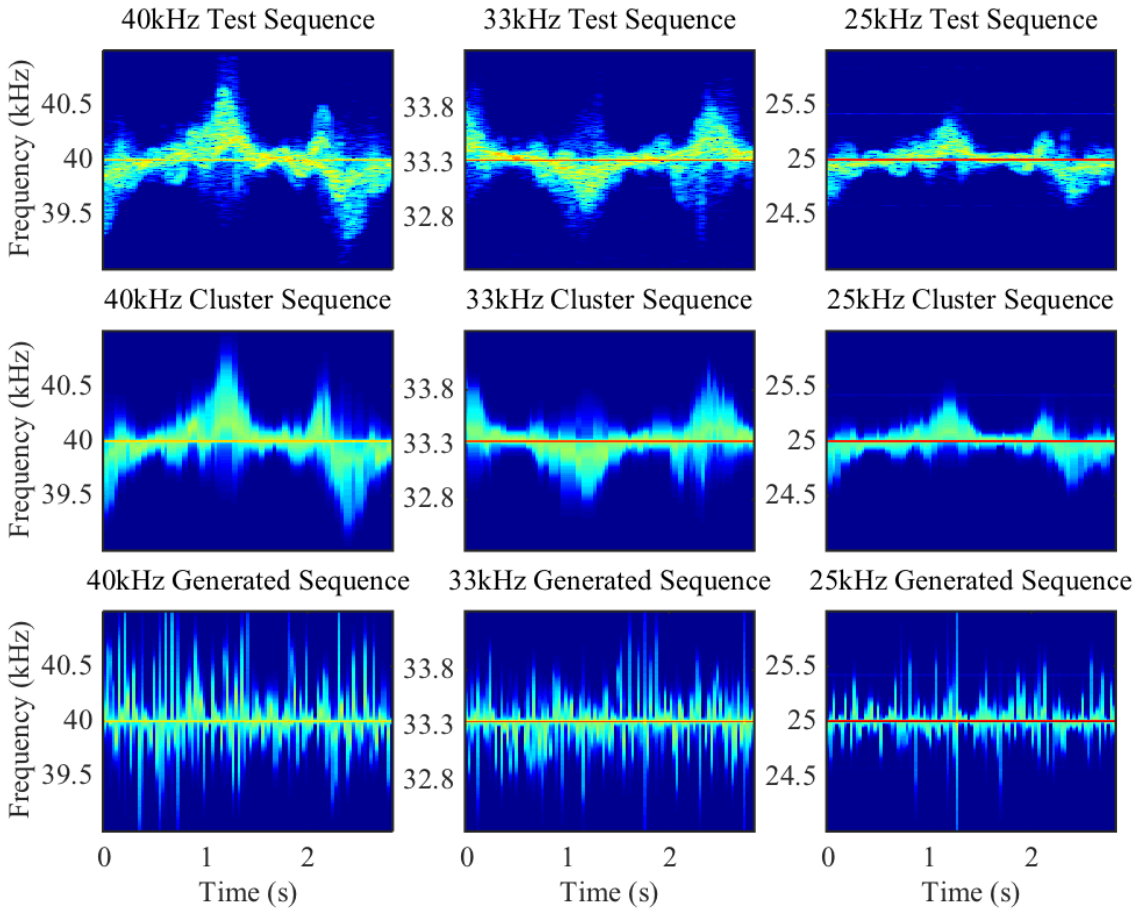


Figure 5.35: An example of a test data sequence for the jump forward and backward (NE-SW) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

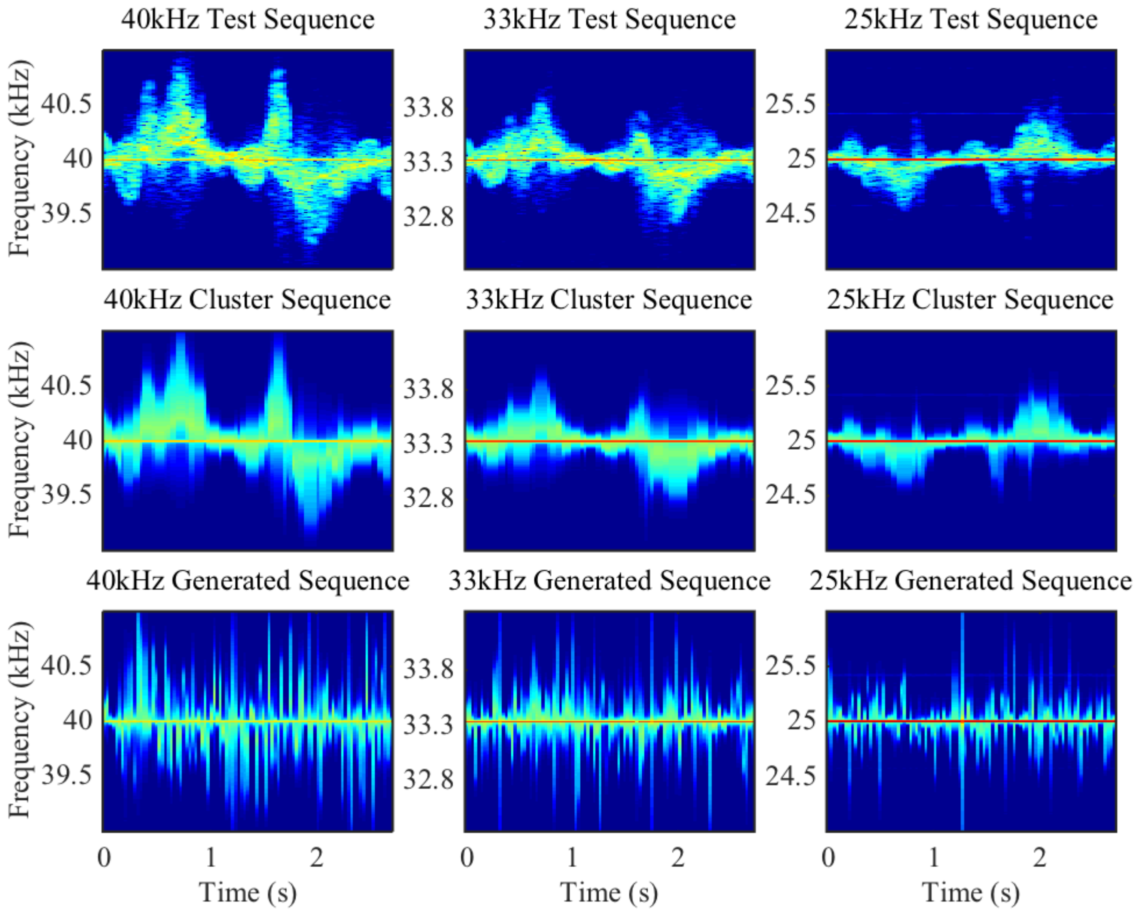


Figure 5.36: An example of a test data sequence for the jump forward and backward (NW-SE) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

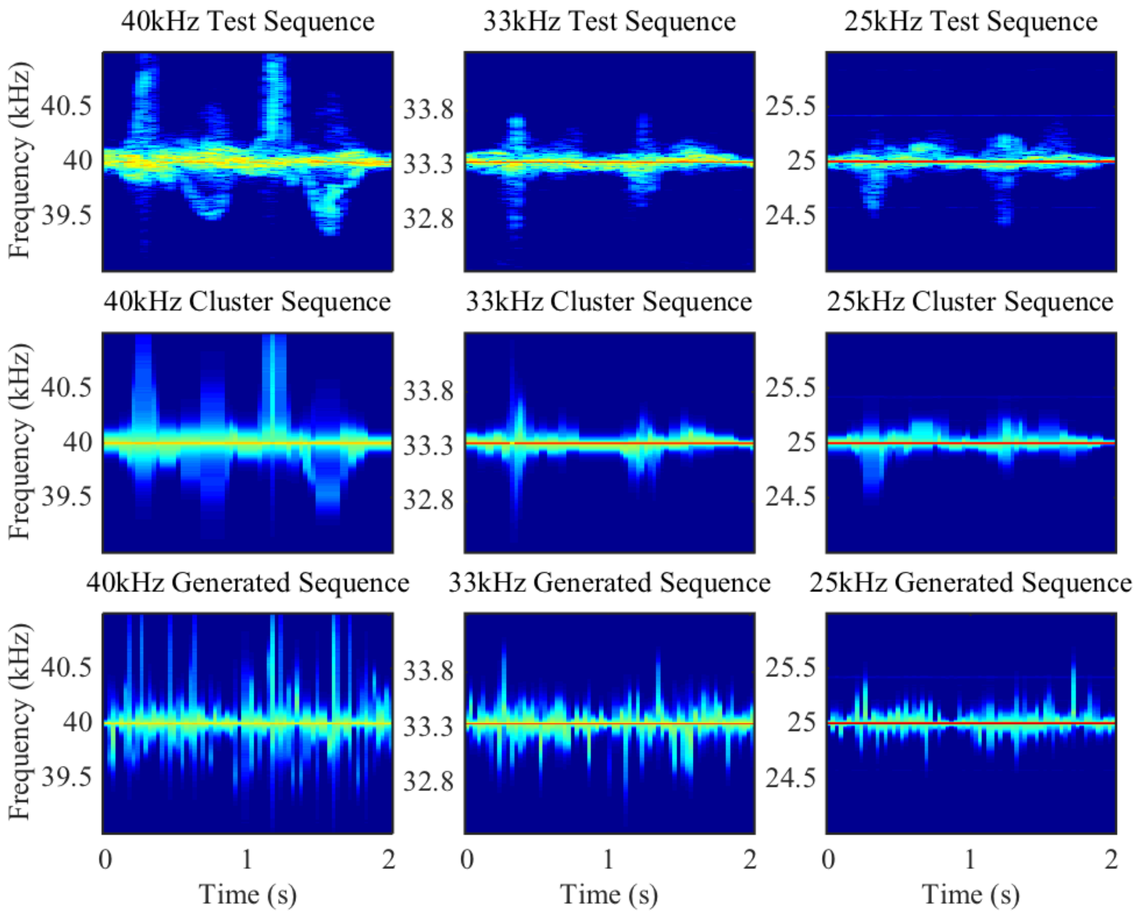


Figure 5.37: An example of a test data sequence for the punch forward (N) action is shown in the first row for each of the three ultrasound sensors. The second row shows the corresponding observation sequence composed of elements from the appropriate set of ultrasound clusters. The third row shows an example ultrasound sequence generated by each of the three HMM models.

Chapter 6

Simulating the Doppler Physics of Human Action

In this chapter a more sophisticated, physics-based, forward model for predicting micro-Doppler modulations from a sequence of skeletal poses is developed. The model approximates the scattering surfaces of the human body with a set of ellipsoids and uses the physics of the Doppler effect to model the frequency shifts induced by motion in the action sequences. Although the model assumes a monostatic sonar configuration, the location of the sonar sensor is configurable. Therefore, a single sequence of skeletal poses can be used to estimate the resulting micro-Doppler modulations that would be observed at any location around the subject. This is an important step toward developing algorithms that can operate in new environments, because it allows a learning algorithm to potentially leverage data collected under one configuration to train the parameters for a novel configuration without collecting a new dataset.

6.1 Calculating Doppler Components for Kinect Joints

The physics-based micro-Doppler model assumes the monostatic sonar configuration described in Section 2.2. It is also tailored for the continuous-wave transmission model used by the ultrasound sensor units, described in Section 3.2. The transmitted signal is a continuous sinusoid with a fixed carrier frequency f_c. The joints on a skeletal track from the Kinect sensor form a set of time-series of points in three-dimensional space. The timestamp t_i is available for each Kinect frame thanks to the DACQ synchronization signal. The Doppler shift associated with a skeletal joint is computed using Equation 2.6. However, this requires the radial velocity component of the joint with respect to the sonar.

Focusing on a single joint, the velocity $\mathbf{u}_{t_i}$, at frame t_i is computed by taking the difference between the position, $\mathbf{p}_{t_i}$, in the current frame and in the previous frame, which occurred at t_{i-1}. The difference is adjusted for the elapsed time between frames and the three-dimensional velocity can be expressed as,

$$\mathbf{u}_{t_i} = \frac{\mathbf{p}_{t_i} - \mathbf{p}_{t_{i-1}}}{t_i - t_{i-1}}. \tag{6.1}$$

If $\mathbf{r}$ is the directional vector from the joint to the sonar sensor, then the radial velocity component v_{t_i} of the joint with respect to the sonar at frame t_i is,

$$\mathrm{v}_{t_i} = \frac{\mathbf{r}}{||\mathbf{r}||} \cdot \mathbf{u}_{t_i}. \tag{6.2}$$

The transmitted frequency is simply the carrier frequency f_c. Applying the radial velocity component of the joint and the carrier frequency to Equation 2.6 gives the received frequency, f_r, at the sonar as

$$f_r = \left(1 + 2\frac{\mathrm{v}_{t_i}}{c_s}\right) \cdot f_c. \tag{6.3}$$

Unfortunately, the Doppler modulation observed in the received frequency is not limited to a single skeletal joint, approximated as a point mass. Instead, a human body is composed of a number of body segments or limbs, each with its own scattering cross-section. In general, these surfaces each have their own range of radial velocity components and the carrier signal is modulated accordingly by each surface when it is scattered. The observed micro-Doppler modulation is a superposition of all of these scattered signals. In the rest of the chapter, a simple compositional model of the human body is proposed and the simple Doppler physics expressed in Equation 6.3 are extended to model the full micro-Doppler modulation induced by a moving human.

6.2 The Human Model

The human body is modeled with a set of twelve rigid ellipsoids, each of which approximates a particular body segment. This segment model was originally proposed by Bradley[16] and Boulic.[62] All of the body segments are modeled by prolate spheroids ($a = b < c$) except for the torso, which is considered wider than it is thick ($a = 2 * b < c$). An ellipsoid is defined by the equation,

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1. \tag{6.4}$$

The parameters are marked in the ellipsoid schematic, illustrated in Figure 6.1.

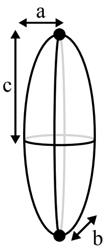


Figure 6.1: Geometry of an ellipsoid segment.

To animate the human model, the segments are mapped onto the joints of a human skeleton model. Skeletal models can be recorded using motion capture systems or extracted from RGB-Depth sensors like the Kinect. The mapping of the ellipsoids onto the skeletal model involves choosing two skeletal joints to be the end-points of each segment. Table 6.1 enumerates this mapping between ellipsoid segments and skeletal joints captured by the Kinect sensor. Note that not all of the joints in a full

Kinect skeleton are utilized when mapping the body segment model onto the Kinect skeleton data. This mapping between the ellipsoids and the joints is diagramed in Figure 6.2.

Table 6.1: Mapping ellipsoid body segments to Kinect skeletal joints.

Ellipsoid Segment	Skeletal Joint A	Skeletal Joint B
left upper leg	left hip	left knee
right upper leg	right hip	right knee
left lower leg	left knee	left ankle
right lower leg	right knee	right ankle
left foot	left ankle	left toe
right foot	right ankle	right toe
left upper arm	left shoulder	left elbow
right upper arm	right shoulder	right elbow
left lower arm	left elbow	left wrist
right lower arm	right elbow	right wrist
torso	shoulder center	hip center
head & neck	head	shoulder center

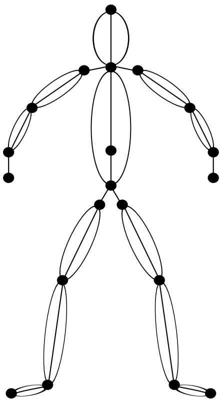


Figure 6.2: Relationship between the Kinect skeletal model and the ellipsoid segments used to represent the human body.

The ellipsoid segment model allows us to take readily available human motion data in the form of skeletal models and augment the skeleton with geometric shapes that have volume and surfaces that can reflect sound. While an ellipsoid is a rough approximation for many of the body segments, the closed form solution for an ellipsoid's cross-sectional area plays a critical role in the Doppler model described in the next section. This efficiency is important because the cross-sectional area must be recalculated every time the human model moves relative to the ultrasound sensor.

6.3 Doppler Model of a Body Segment

After decomposing the human body into a set of ellipsoid segments that are reasonably simple to deal with computationally, the next step is to model how the segments scatter the transmitted carrier frequency. The Doppler model for a single body segment combines the pair of skeletal joint sequences that define the endpoints of the segment and the ellipsoid approximation of the segment to predict the micro-Doppler modulations resulting from the segment's motion. The model results in a time-frequency representation that describes the contribution of the body segment to a spectrogram. The contributions of each segment can be developed independently before they are superimposed in the time-frequency domain.

The micro-Doppler modulation is estimated using three parameters that are extracted for each ellipsoid body segment from the the skeletal tracking data at each frame. The first two parameters are the maximum and minimum radial velocity components contributed by each segment. The third parameter is an orientation angle that summarizes how the ellipsoid is rotated relative to the ultrasound sensor.

The radial velocity component parameters are used to generate the envelope of the segments contribution in the time-frequency domain. Each ellipsoid segment is a rigid object. Therefore, the minimum and maximum radial velocity components are attributed to the endpoints of the ellipsoid. The velocities of all other points on a rigid ellipsoid are distributed between these two extremes. The endpoints are the joints from the human skeletal model associated with the given ellipsoid segment. By convention, the joint closer to the hip-center in the skeletal tree structure is labeled as joint A and the joint farther from the hip-center is labeled as joint B. This is illustrated in Figure 6.4. The radial velocities of these joints, extracted for a single frame, are v_A and v_B respectively.

Micro-Doppler modulations depend only on the radial velocity component of human motion with respect to the ultrasound sensor. Figure 6.3 illustrates the pair of time-series created by concatenating the joint velocities, bounding a left leg segment, from a sequence of Kinect frames. This defines an envelope in the time-frequency domain that constrains the range of frequency contributions from a single rigid segment.

The orientation angle is used to compute the appropriate scattering cross-sectional area, σ_θ, of the ellipsoid segment. The scattering area represents the size of the surface that is available to reflect the transmitted signal back to the ultrasound sensor. For

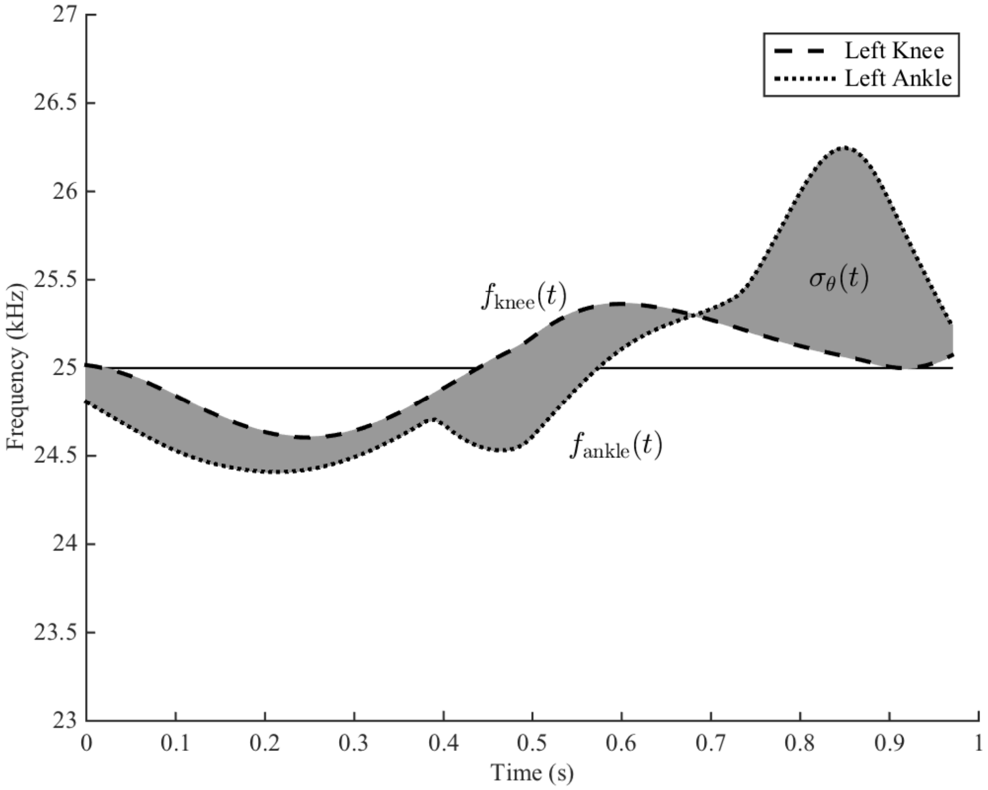


Figure 6.3: Simulation of the Doppler contribution from a single ellipsoid segment. In this case the segment is the lower left leg, joint A is the knee and joint B is the ankle.

prolate spheroids, it can be expressed as

$$\sigma_\theta = \frac{\pi a^2 b^2 c^2}{(a^2 \sin^2(\theta) + c^2 \cos^2(\theta))^{\frac{1}{2}}}. \tag{6.5}$$

This solution for the scattering area assumes that the dimensions of the ellipsoid are much larger than the wavelength of the scattered signal.[63] The amount of energy reflected by the ellipsoid segment is proportional to the scattering area σ_θ. This quantity is used to adjust the intensity contributed by a given segment to the frequency range within the envelope defined by the velocity bounds as illustrated in Figure 6.3. There are many other factors, such as the strength of the transmitted signal and the distance between the reflector and the transmitter, that affect the intensity of the reflected signal. However, the scattering area is a good first-order approximation to the relative reflected signal intensity among the segments as long as the range of the human body from the ultrasound sensor does not vary too drastically.

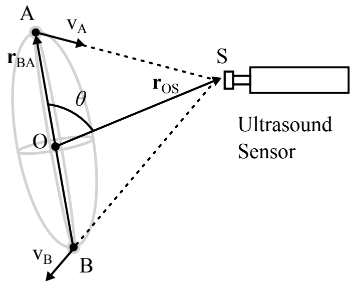


Figure 6.4: Diagram illustrating the parameters extracted for each ellipsoid segment. Note that the 2D plane of the diagram is rotated to contains the joint A, joint B and the ultrasound sensor S.

In order to compute the scattering area, the orientation and dimensions of each ellipsoid segment need to be quantified. Computing the orientation angle θ of the ellipsoid segment requires the directional vectors $\mathbf{r}_{BA}$, the vector from the lower joint B to the upper joint A, and $\mathbf{r}_{OS}$, the vector from the lower joint B to the ultrasound sensor S. Figure 6.4 illustrates these quantities and the overall geometry of an ellipsoid segment relative to the ultrasound sensor. Given these direction vectors, the orientation angle θ can be expressed as,

$$\theta = cos^{-1}\Big(\frac{\mathbf{r}_{BA} \cdot \mathbf{r}_{OS}}{||\mathbf{r}_{BA}||||\mathbf{r}_{OS}||}\Big). \tag{6.6}$$

The dimensions of the ellipsoid segments are derived from physical properties of the human body. The length of an ellipsoid segment is determined by the parameter c, which is half the length of the represented body section as illustrated in Figure 6.1. The skeletal motion data is estimated algorithmically from RGB-Depth maps, so there is some jitter in the position of the joints. The estimation of the joint positions can be affected significantly by the orientation of the human body. To mitigate these effects the limb lengths are averaged over N skeletal frames to provide a more robust estimate. Given a sequence of skeletons for a particular individual, the limb lengths are the average distance between joint A and joint B for each body segment. For each ellipsoid, the parameter c is half this average. Given a sequence of N skeletal poses,

$$c = \frac{1}{2N} \sum_{t=1}^{N} ||X_A - X_B||. \tag{6.7}$$

The thickness and width of an ellipsoid segment is determined by the parameters a and b. The majority of the ellipsoid segments in the human body model are prolate spheroids, so $a = b$. In Bradley,[16] the ellipsoid dimensions for each body segment are estimated from the overall height H in meters and mass M in kilograms of a human subject. Our model utilizes the same method for estimating the thickness of the ellipsoid segments, Table 6.2 reproduces the density ρ_s of each body segment used in the human model as well as the percentage of the total mass M attributed to the mass M_s of each segment. These anthropometric measurements were originally reported in Winter.[64]

Table 6.2: Anthropometric data used to extract ellipsoid segment dimensions .

Body Segment	Density [ρ_s] (gm/cm^3)	Mass [M_s] (kg)
upper leg	1.05	$0.100 \times M$
lower leg	1.09	$0.0465 \times M$
foot	1.10	$0.0145 \times M$
upper arm	1.07	$0.028 \times M$
lower arm	1.14	$0.022 \times M$
torso	1.03	$0.497 \times M$
head	1.11	$0.081 \times M$

Each body part is modeled by an ellipsoid segment, so the volume of a segment V_s can be expressed in closed form as

$$V_s = \frac{4}{3}\pi abc. \tag{6.8}$$

The mass of the segment M_s is related to the volume of the segment by the density of the segment ρ_s, such that

$$M_s = \rho_s V_s. \tag{6.9}$$

Setting the segment volumes to be equal yields the expression

$$\frac{M_s}{\rho_s} = \frac{4}{3}\pi abc, \tag{6.10}$$

which can be rearranged to solve for the thickness a of the ellipsoid in terms of M_s,

ρ_s, c and b such that

$$a = \frac{3}{4}\frac{M_s}{\rho_s}\frac{1}{\pi bc}. \tag{6.11}$$

Adding the assumption that the body segments are prolate spheroids, which by definition sets $a = b$, eliminates the dependence on b and results in the following expression for both the width and thickness dimensions of the ellipsoid,

$$a = \sqrt{\frac{3}{4}\frac{M_s}{\rho_s}\frac{1}{\pi c}}. \tag{6.12}$$

The prolate spheroid assumption is used for all segments except the torso, where it is assumed that the ellipsoid is twice as wide as it is thick. That is, $b = 2a$, which simplifies to,

$$a = \sqrt{\frac{3}{8}\frac{M_s}{\rho_s}\frac{1}{\pi c}}. \tag{6.13}$$

6.4 Combining the Modulations in the Time-Frequency Domain

In the previous section the capability of modeling the Doppler modulation contribution of a single segment was developed. In this section, the contributions are combined to form a complete micro-Doppler modulation. In general, the goal is to produce a spectrogram representation that mimics and is synchronized to the spectrograms derived from actual ultrasound recordings. The result is a spectrogram model that can be compared side-by-side with the actual ultrasound data.

Given a sequence of skeletal poses and the location of an ultrasound sensor, the Doppler modulation due to each of the body segments is computed every time a new FFT slice would be computed in the recorded ultrasound spectrogram. This requires a buffer to hold the previous frame of Kinect skeletal data in order to compute the joint velocities. The resulting radial joint velocities are then mapped onto a set of frequency bins identical to those used in the spectrogram representation of the recorded ultrasound data. Each pair of joints that define a body segment now also define a range of frequency bins that the segment contributes energy to. The amount of energy is proportional to the scattering cross-sectional area σ_θ normalized by the number of frequency bins the segment contributed to. Intuitively, a body segment scatters an amount of signal energy proportional to it's cross-sectional area. If the segment is moving with a uniform radial velocity, then all of this energy is concentrated in a single frequency bin corresponding to that velocity. However, if the segment has a large range of radial velocities, the reflected energy will be divided more or less evenly across a range of frequency bins that correspond to those velocities.

Finally, the contributions of all the segments are superimposed and the energy contributed by each segment to each frequency bin is accumulated together. Unfortunately, there is some unknown proportionality constant related to the strength of the transmitted signal and the range of the human body from the ultrasound sensor. In order to scale the simulated modulations appropriately, the baseline noise level of the DACQ system, which was derived in Section 3.1.3, is used as a baseline offset. Then a global gain is applied uniformly to the frequency bin contributions so that the model output roughly matches the range of the actual ultrasound recordings.

6.5 Skeletal Smoothing

One of the challenges associated with using skeletal motion data is that some of the joints, particularly the hands and feet, tend to exhibit a fair amount of jitter. Many applications that rely on skeletal motion data actually drop the noisier joints if possible as was done for the HMMs in Chapter 5. Unfortunately, the hands and feet tend to have some of the highest velocity components and are critical for accurately generating micro-Doppler modulations that have enough rich structure to be comparable to the modulations seen in the spectrograms computed from actual ultrasound sensor recordings.

As a compromise, the skeletal motion data is filtered in order to reduce the jitter and discontinuities associated with skeletal tracking failures. To achieve this a Kalman filter[65,66] was applied to the joint positions, velocities and accelerations. The jitter in joints tends to produce very high velocities (i.e. a foot jumping between two positions frame by frame) that generate very spiky modulations in the time-frequency domain. The Kalman filter is tuned to dampen these large jumps without dampening the overall motion too much. This is important because any dampening of the joint velocities creates error in the frequency modulations. However, it is reasonable to accept some amount of dampening in order to eliminate some of the erroneous velocity spikes caused by jittery skeletal motion.

6.6 Spectrogram Smoothing

Within the spectrogram output by the model itself there are also a lot of discontinuities in the frequency bins. In actual data, the contributions of the individual limbs tend to be more gradual and blended together than the hard cutoffs assumed by decomposing the human body into just a handful of discrete segments. To help remedy this, a small averaging filter, with a 3×3 kernel, was applied to the model output as a post-processing step. Although a larger kernel does a better job of smoothing over the discontinuities, it also loses a lot of the finer structure that makes the output spectrograms interesting, so the kernel size was kept small.

6.7 Example Simulated Micro-Doppler Modulations of Human Actions

Figures 6.5 through 6.25 show an example output of the micro-Doppler physics model for each of the 21 actions in the JHUMMA dataset. In each of the figures, the first row contains a spectrogram computed from recorded ultrasound data from

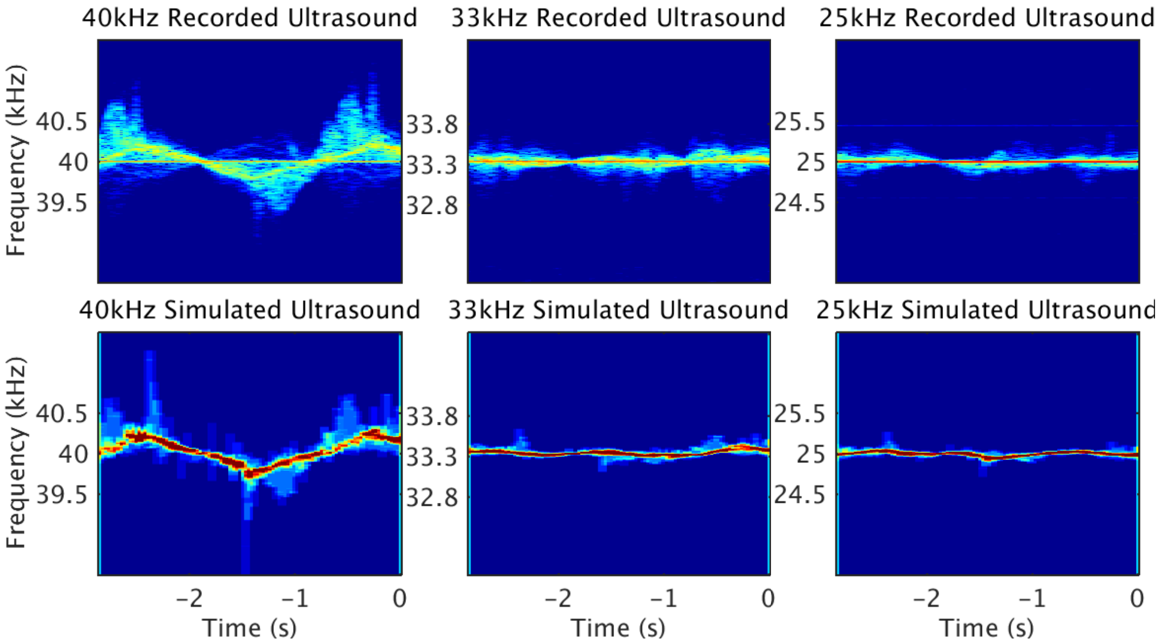


Figure 6.5: The first row shows an example of raw ultrasound data for the lunge (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

each of the three ultrasound sensors. The 40kHz unit was positioned in front of the majority of the actions, to the north, and the 33kHz and 25kHz units were positioned on the sides, to the west and east respectively. The second row of each figure shows the output of the micro-Doppler physics model configured to predict the modulations for an ultrasound unit with the same frequency and location as the actual recorded data. The model takes a single sequence of Kinect skeletons to generate all three of the outputs. The time window for the Kinect sequence and each of the recorded ultrasound time-series is identical.

For the most part, the model does a reasonable job reproducing the character of the micro-Doppler modulations. Certainly the timing of the modulations in the

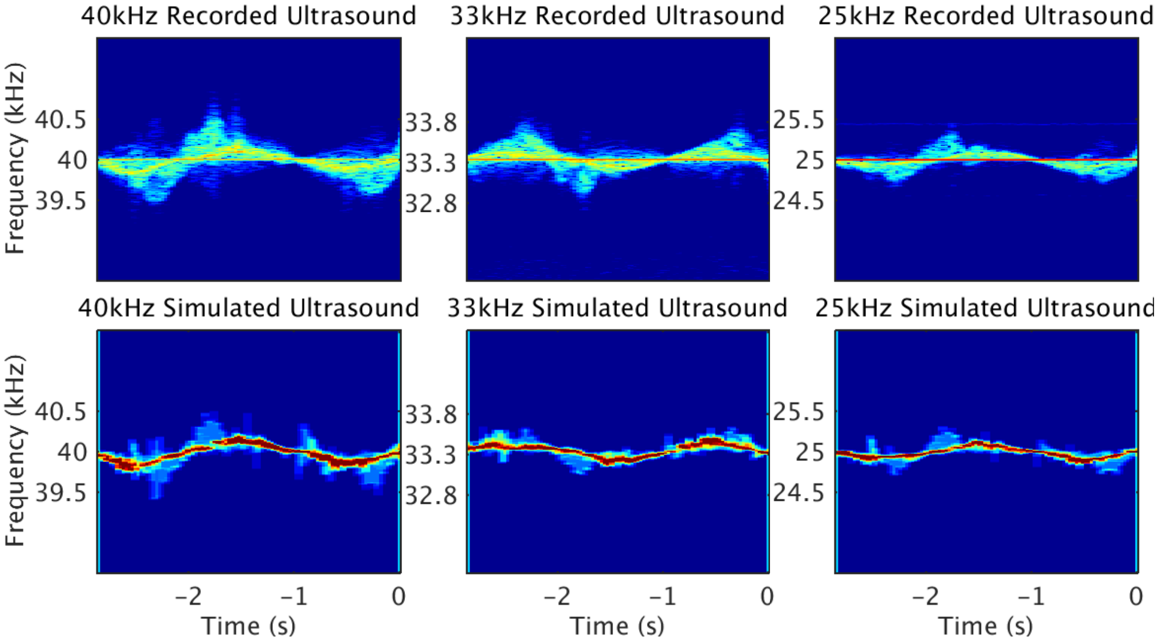


Figure 6.6: The first row shows an example of raw ultrasound data for the lunge (NE) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

model spectrograms is faithful to the spectrograms computed from recorded data. The physics model also does an excellent job tracking the slow major modulations induced by actions where the entire body, and particularly the torso, translate in space.

However, there are also mistakes that can be attributed to limitations of various aspects of the Doppler physics model. One that is easy to observe in many of the actions is shrinkage of the frequency range covered by the modulations relative to those in the recorded data. This is at least partially due to the Kalman filter used to smooth out the skeleton tracks from the Kinect sensor that feeds the Doppler physics model. Even with the smoothing, there are still noticeable modulation spikes

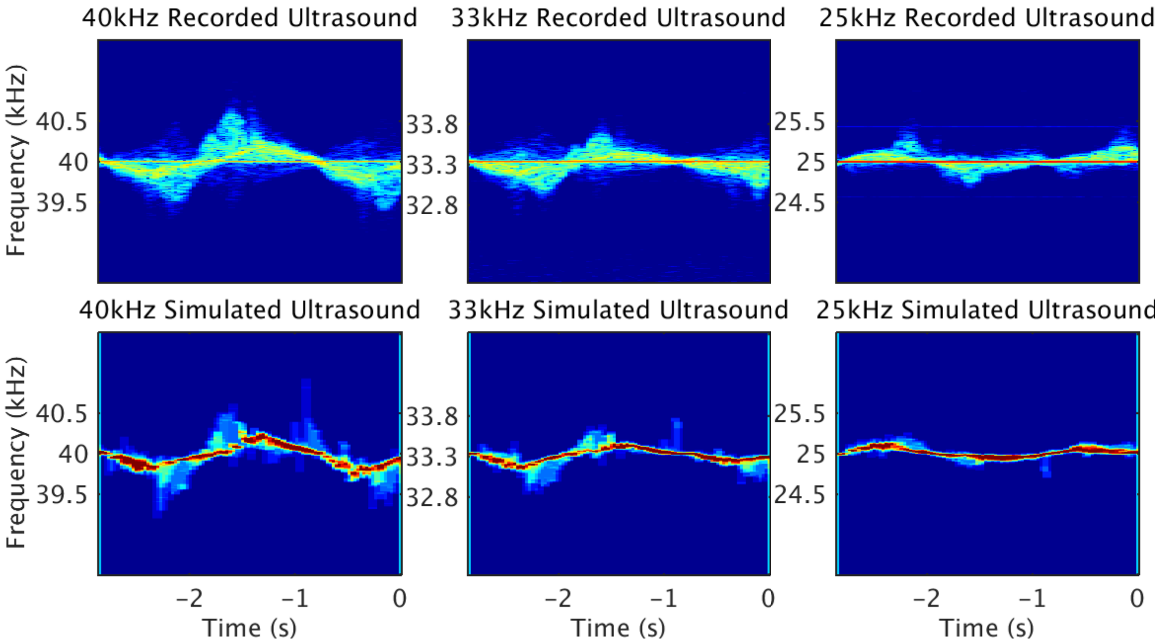


Figure 6.7: The first row shows an example of raw ultrasound data for the lunge (NW) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

in the output of the physics model that coincide with jitter and larger discontinuities due to poor tracking in the Kinect data. These spikes would be improved by more accurate skeletal tracking or better smoothing of the joint trajectories. Unfortunately the smoothing comes at the expense of the character created by actual quick actions.

Another interesting limitation of the physics model manifests very clearly in Figure 6.11 and Figure 6.13, which show the left and right side arm raise actions, respectively. In the recorded data, it is clear that the head and torso block the ultrasound sensor on the far side of the moving arm from observing much scattering from the action. For instance, on the left arm side raise, the 25kHz sensor, which was positioned off to the right of the body, is almost completely blind to the action. Conversely, on

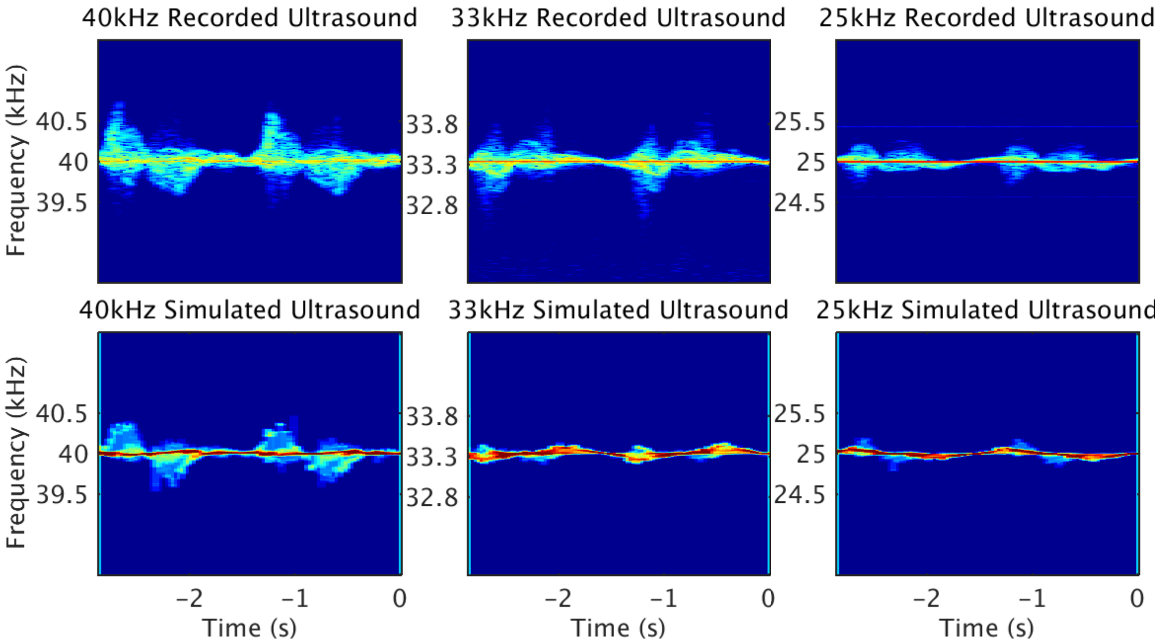


Figure 6.8: The first row shows an example of raw ultrasound data for the left leg step (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

the right arm side raise, the 33kHz sensor, which was positioned off to the left of the body, is almost completely blind to the action. However, the physics model clearly produces Doppler modulations for each of the side sensors. This is a limitation of treating each segment independently and ignoring any potential of one segment to interfere with or block a second segment from scattering the transmitted signal.

Figure 6.20 illustrates a deficiency of the Doppler physics model that is unique to the jumping rope action in the JHUMMA dataset. That is, of course, the presence of the jump rope in the scene. In the spectrogram computed from the recorded ultrasound data, large modulations due to the jump rope scattering the carrier frequencies are observed. However, the jump rope is not modeled at all in the Doppler-physics

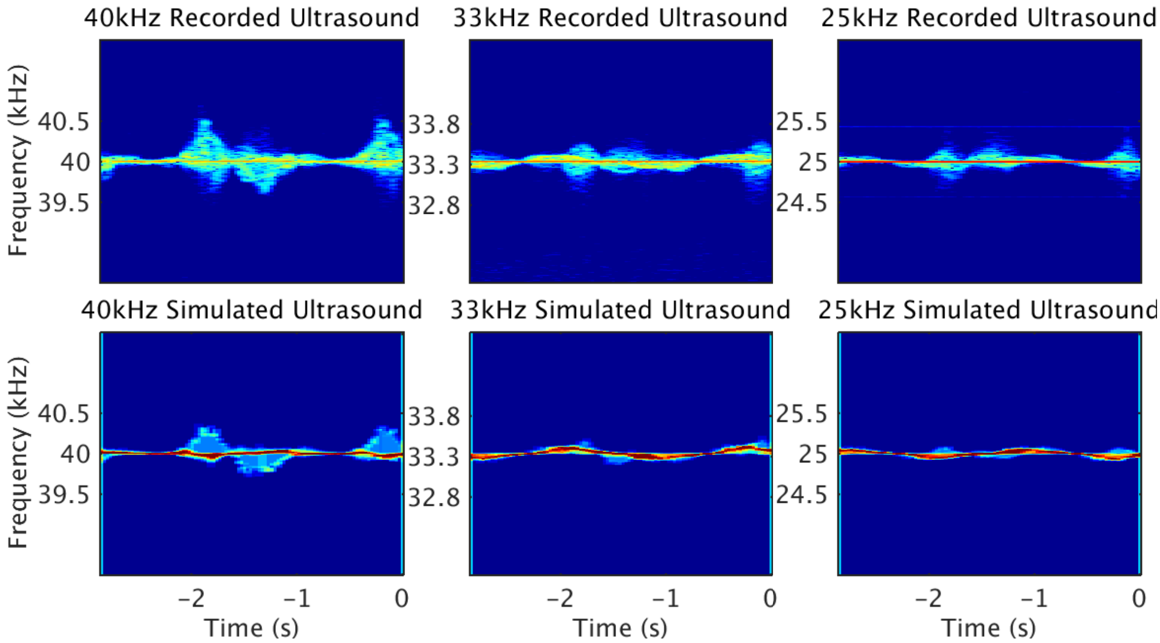


Figure 6.9: The first row shows an example of raw ultrasound data for the right leg step (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

model, so there are no additional modulations due to the jump rope in the model output. Instead, the Doppler-physics model only produces modulations due to the movement of the human actor. There are several other objects, such as clothing, that the physics model does not account for. In fact, the flapping of clothing can produce very fast modulations that add a significant amount of character to the actual spectrograms, but is missing from the output of the physics model.

While there are many ways to improve upon the accuracy of the predictions made by the Doppler physics model, it does a remarkable job of predicting the actual sensor data from a completely different sensor modality. The fact that the Kinect sensor can provide a reliable estimate of the underlying scene that can be leveraged by the physics

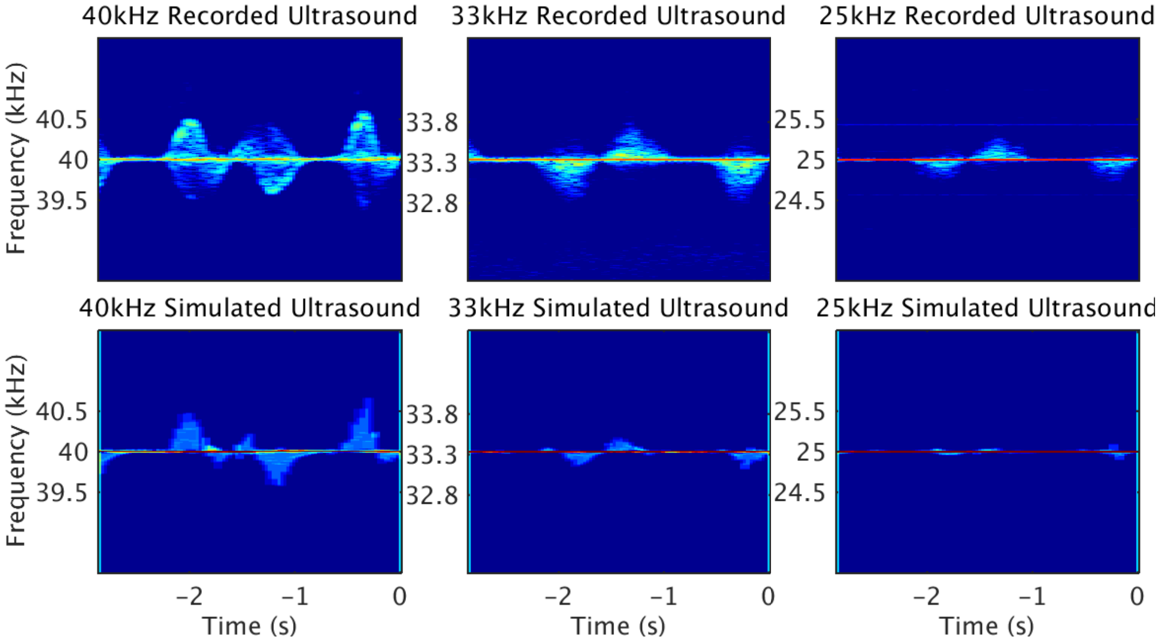


Figure 6.10: The first row shows an example of raw ultrasound data for the left forward arm raise (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

model to generate reasonable Doppler modulations is a powerful tool. Improvements to both the human body model, which could certainly be made more expressive, and the Doppler model, which, for example, could be modified to account for transmission losses due to range, would improve the accuracy of the results. However, all of these modifications add both computational and modeling complexity, which may not be necessary for the task at hand. As it stands, the Doppler-physics model provides a valuable proof of concept approach for generating examples of micro-Doppler modulations that may not be present in an existing training set.

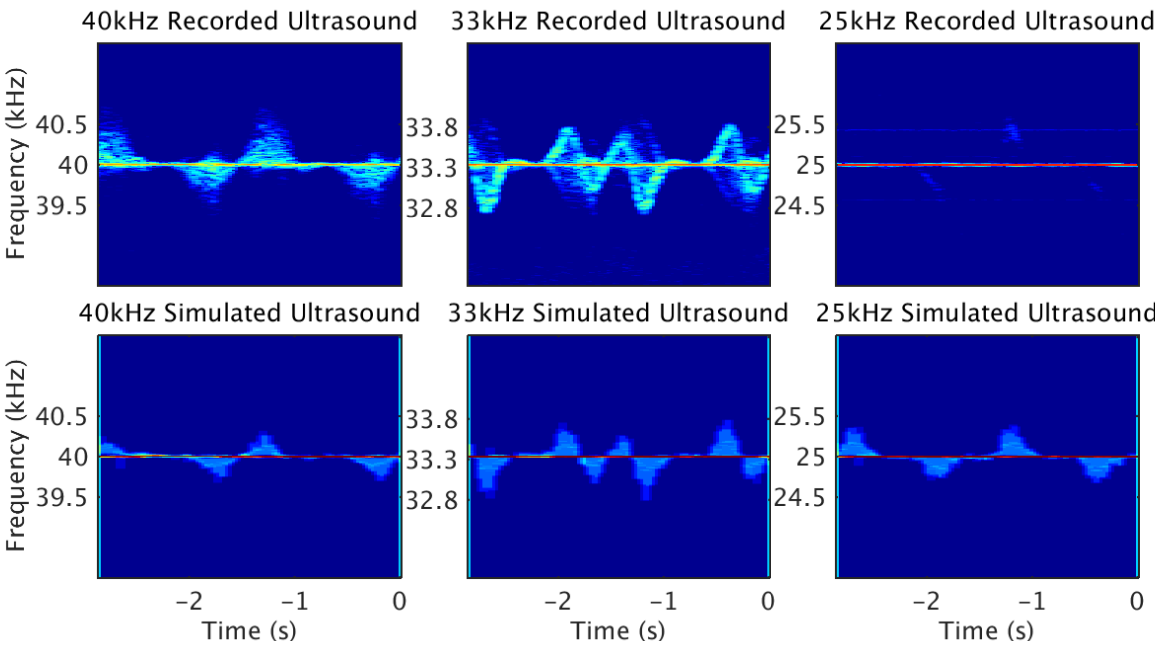


Figure 6.11: The first row shows an example of raw ultrasound data for the left side arm raise (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

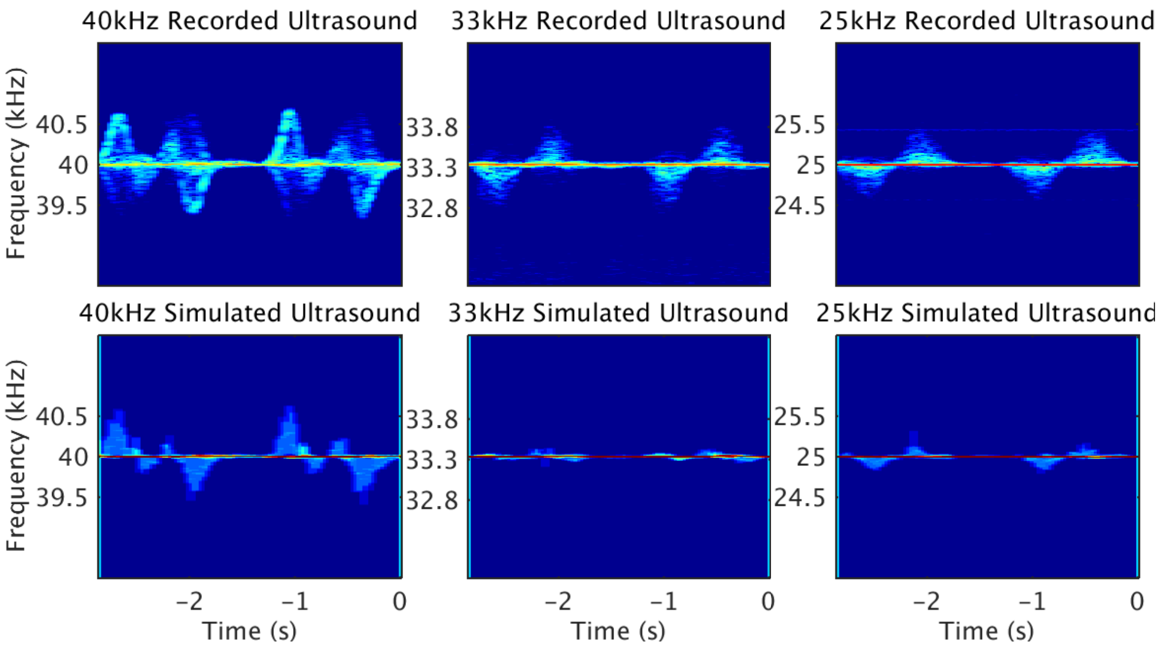


Figure 6.12: The first row shows an example of raw ultrasound data for the right forward arm raise (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

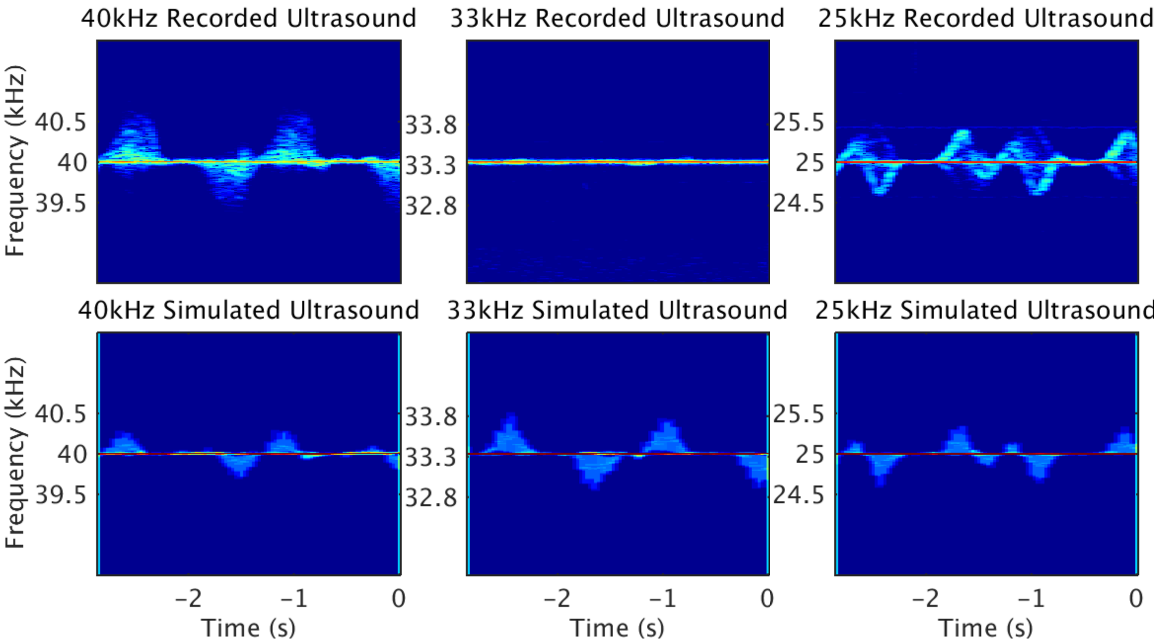


Figure 6.13: The first row shows an example of raw ultrasound data for the right side arm raise (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

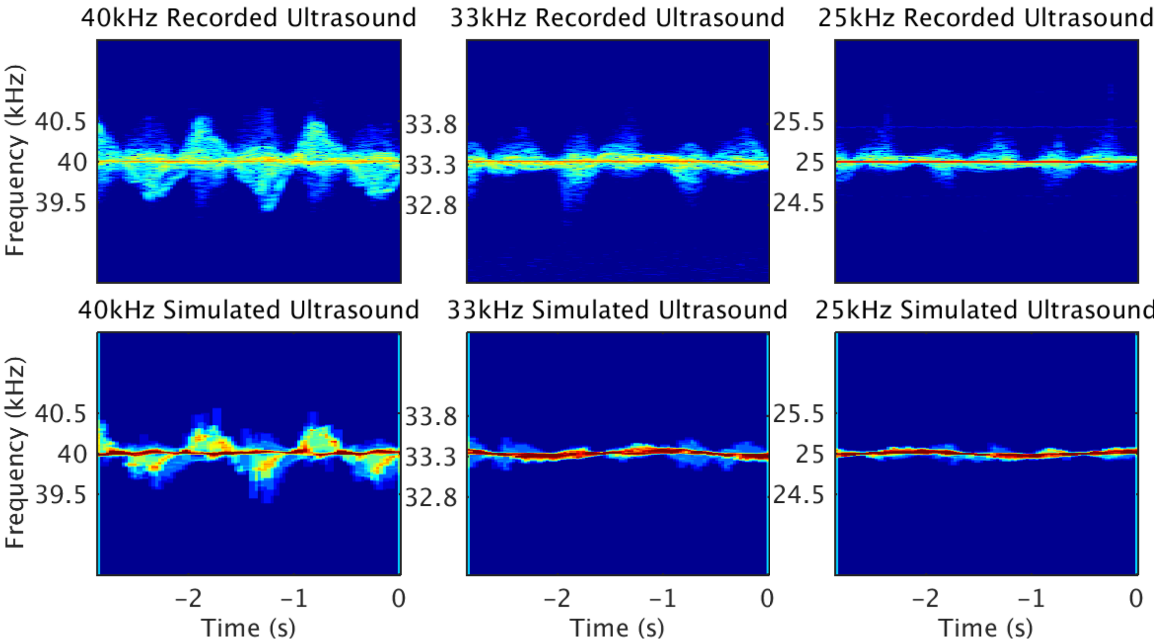


Figure 6.14: The first row shows an example of raw ultrasound data for the walk in place (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

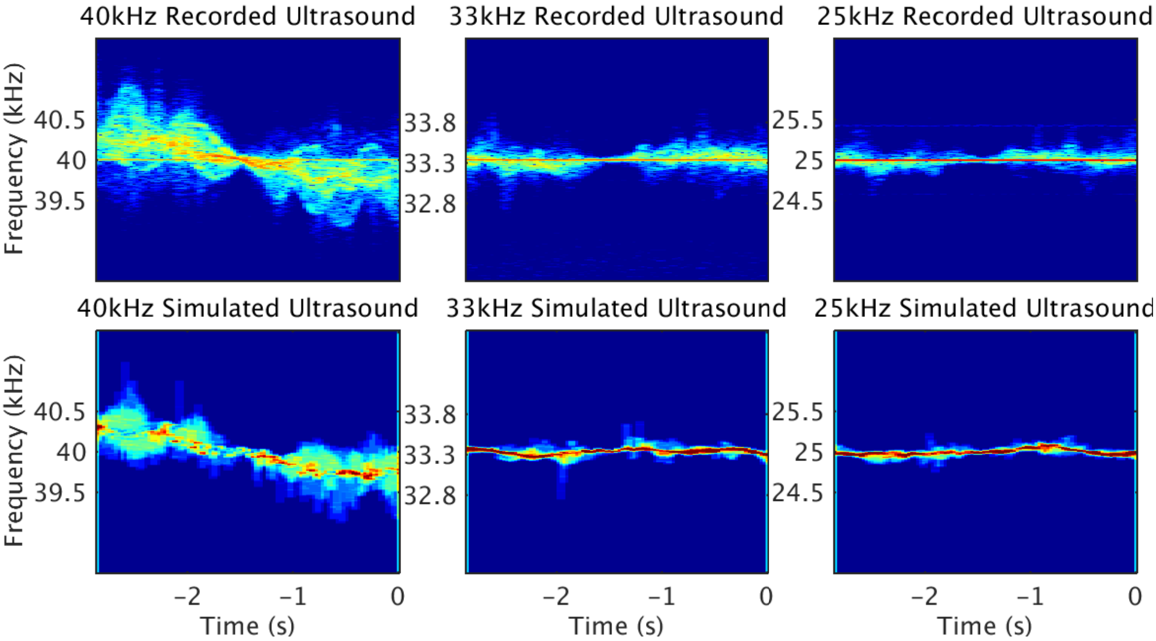


Figure 6.15: The first row shows an example of raw ultrasound data for the walk facing forward (N-S) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

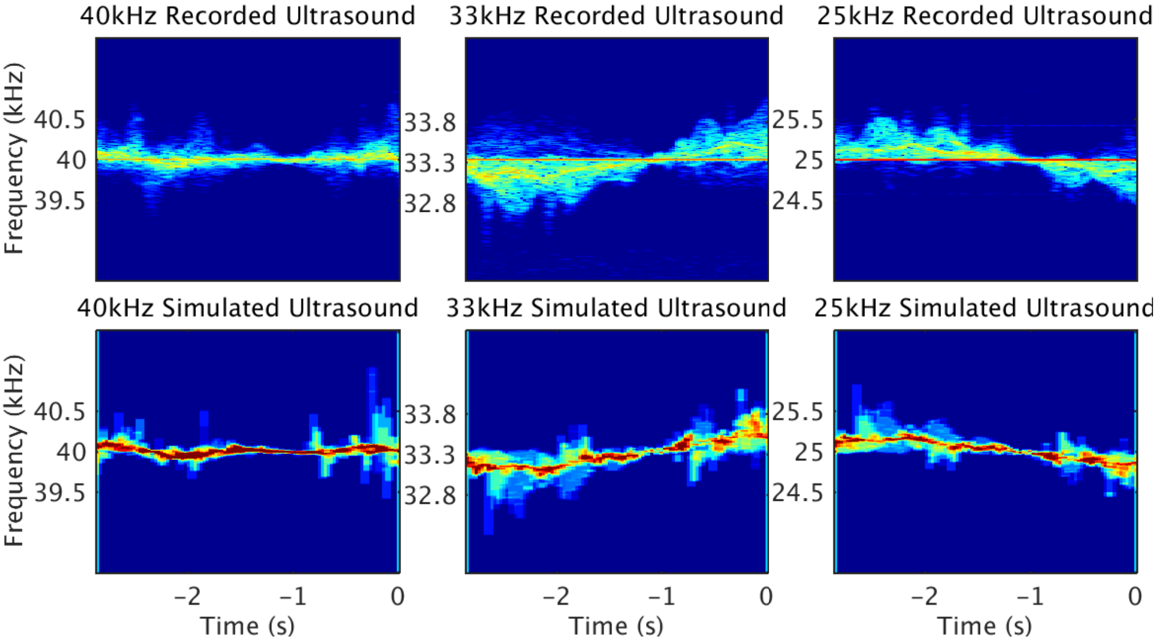


Figure 6.16: The first row shows an example of raw ultrasound data for the walk facing side (W-E) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

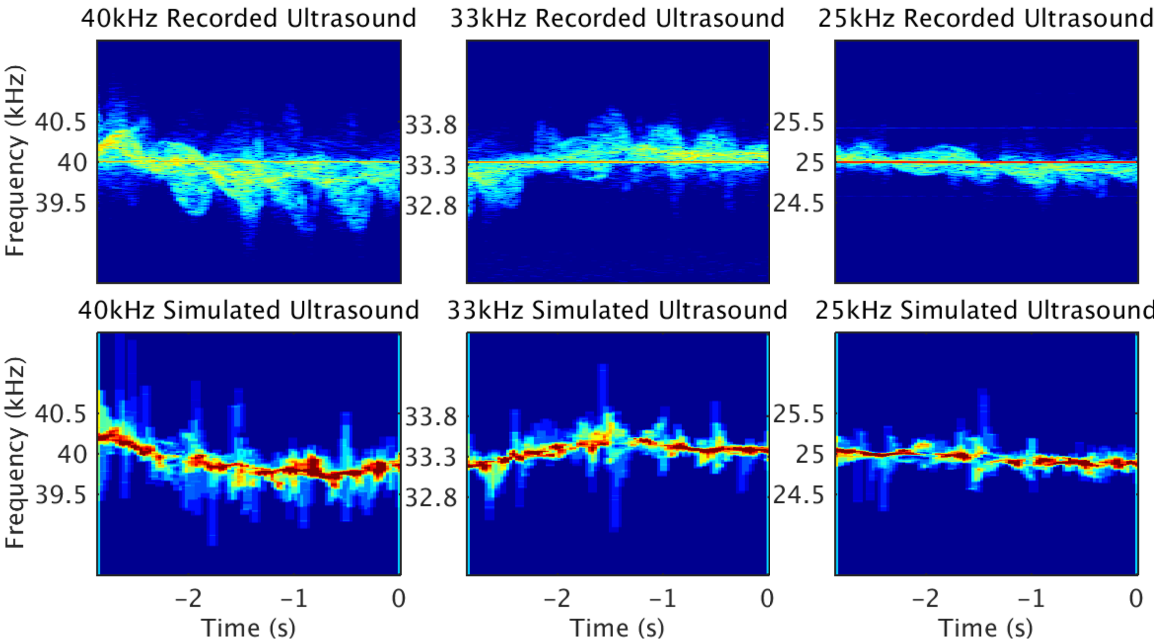


Figure 6.17: The first row shows an example of raw ultrasound data for the walk and pivot (NE-SW) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

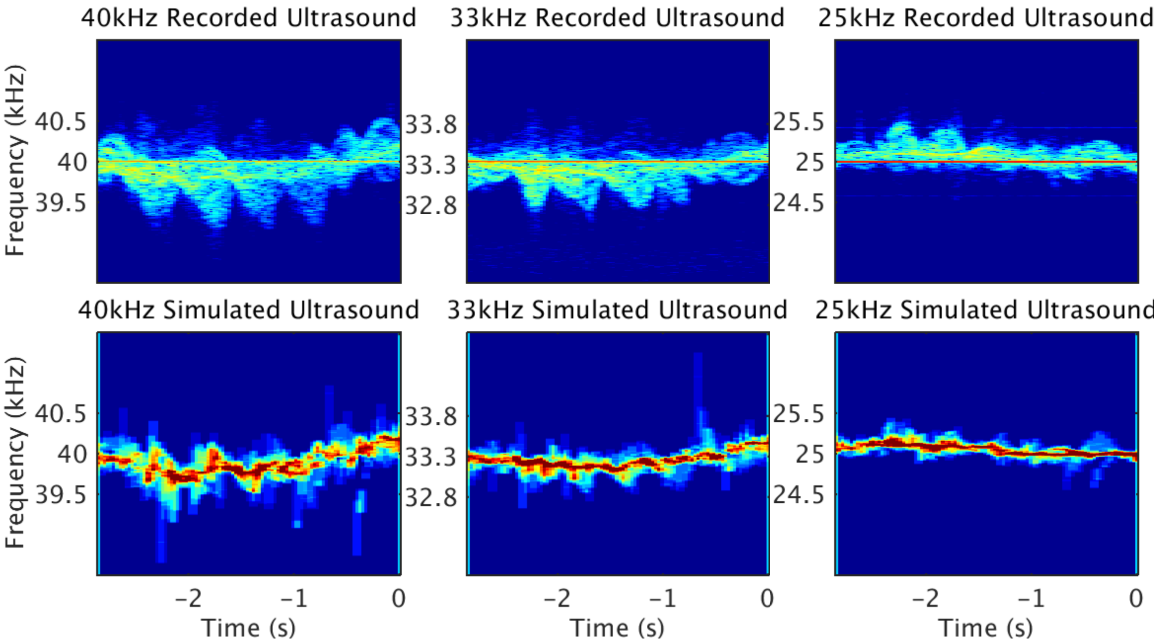


Figure 6.18: The first row shows an example of raw ultrasound data for the walk and pivot (NW-SE) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

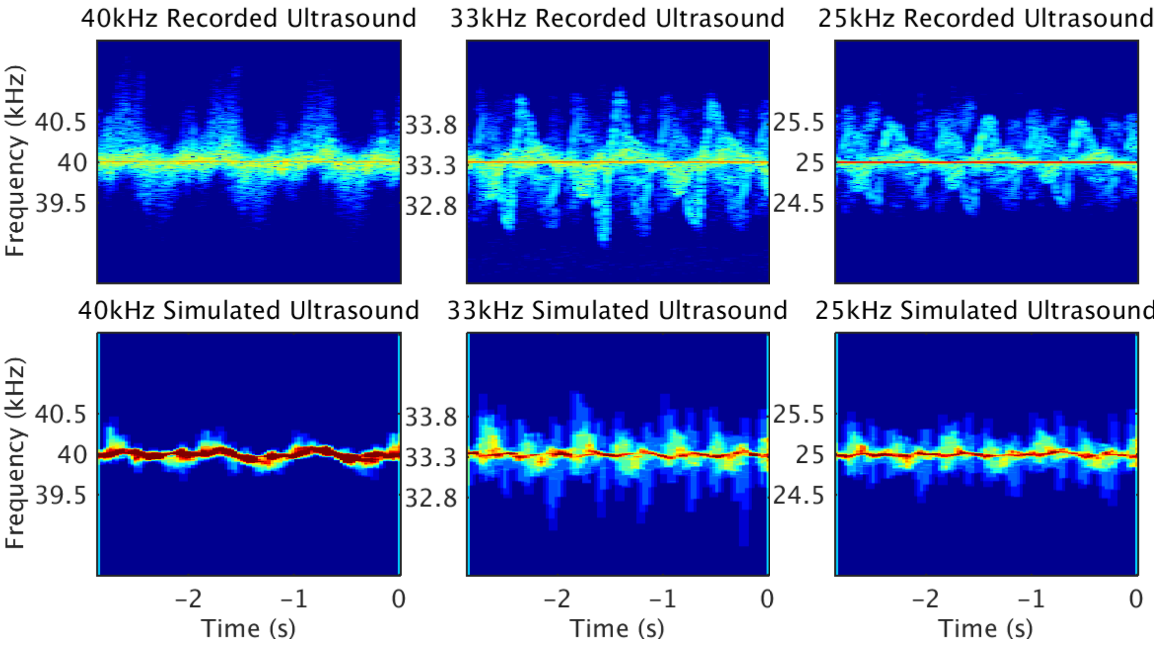


Figure 6.19: The first row shows an example of raw ultrasound data for the jumping jacks (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

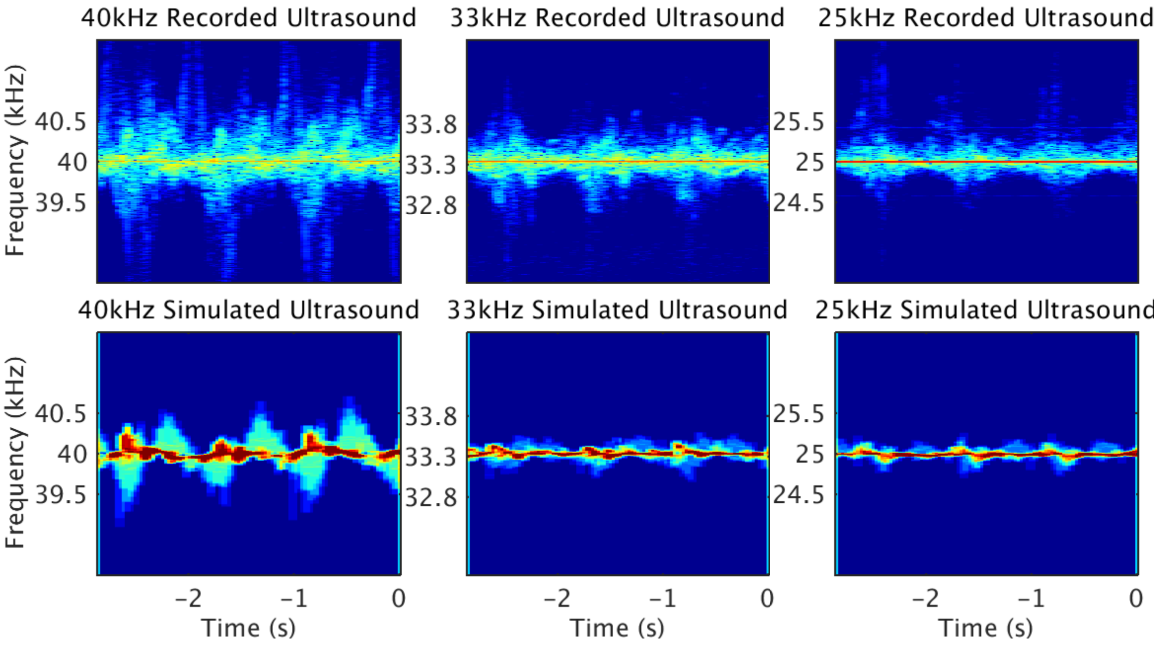


Figure 6.20: The first row shows an example of raw ultrasound data for the jump rope (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

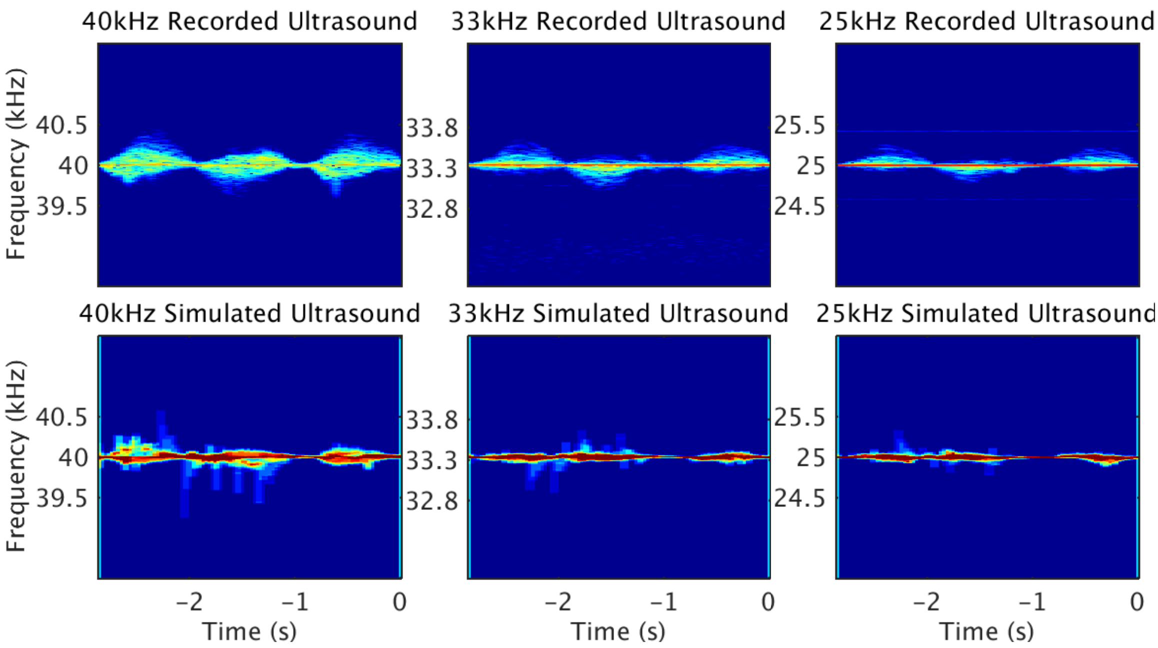


Figure 6.21: The first row shows an example of raw ultrasound data for the body squats (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

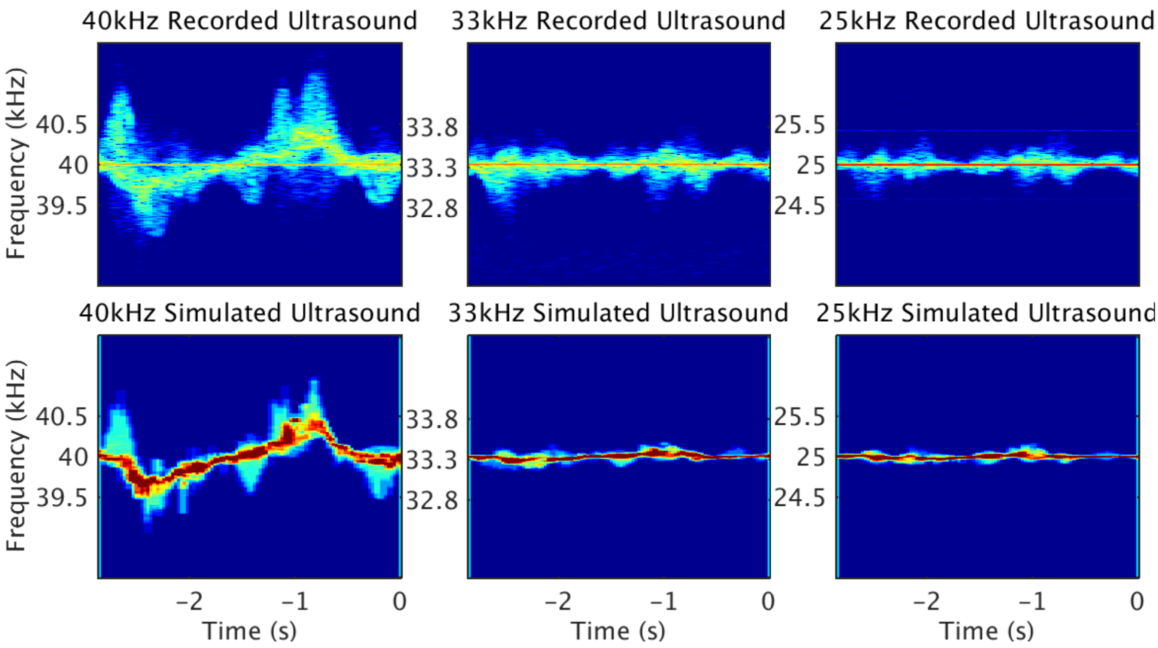


Figure 6.22: The first row shows an example of raw ultrasound data for the jump forward and backward (N-S) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

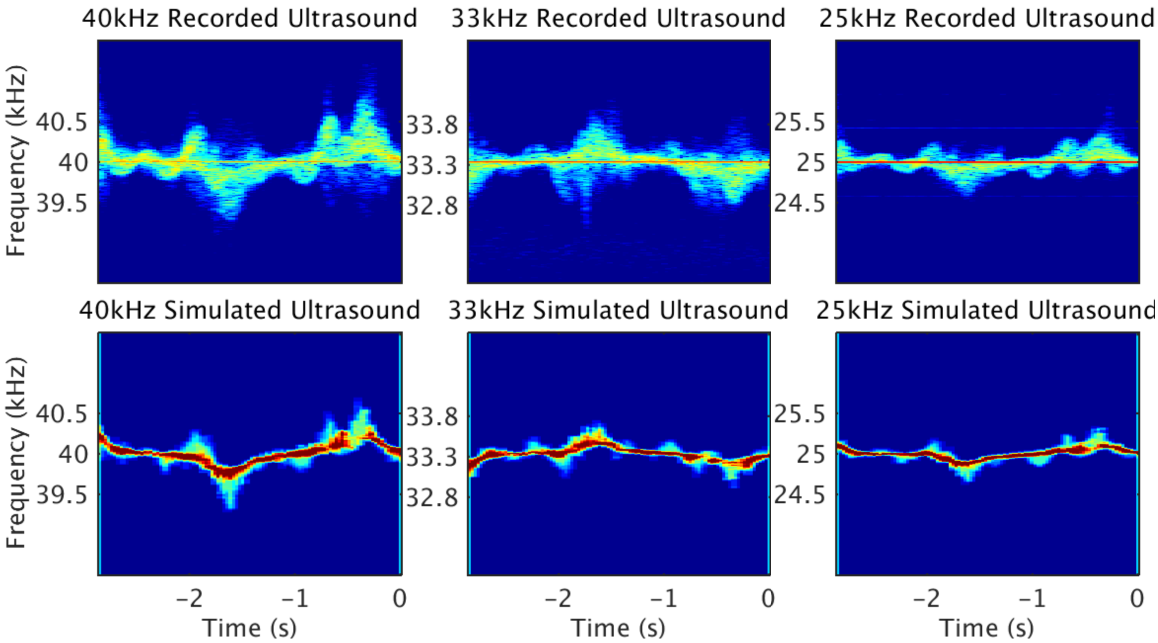


Figure 6.23: The first row shows an example of raw ultrasound data for the jump forward and backward (NE-SW) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

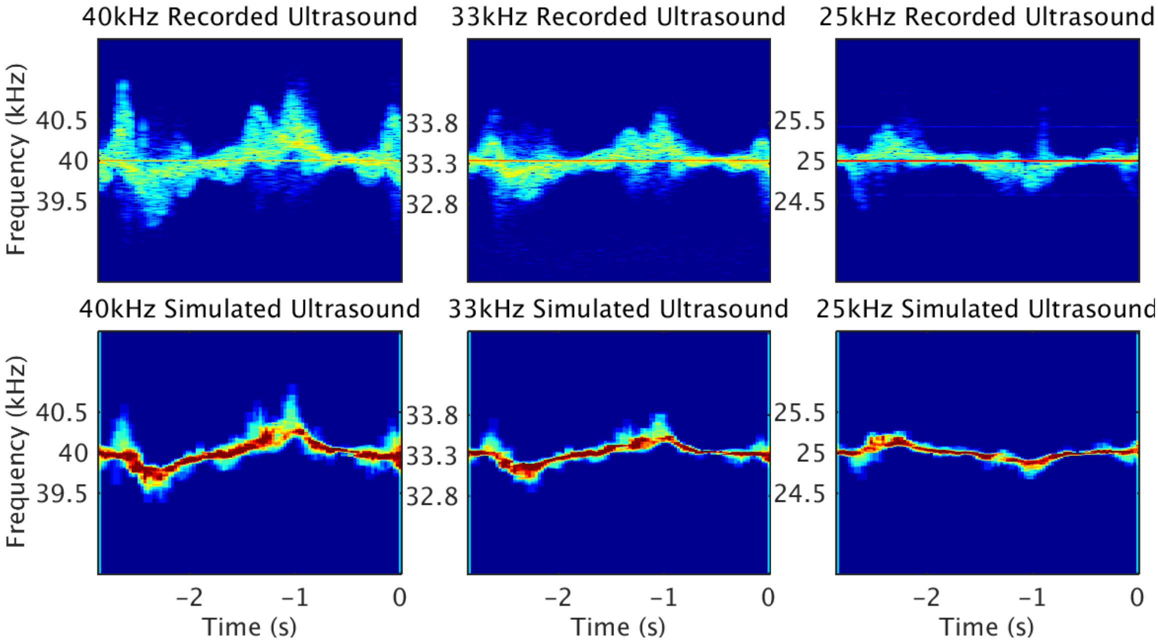


Figure 6.24: The first row shows an example of raw ultrasound data for the jump forward and backward (NW-SE) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

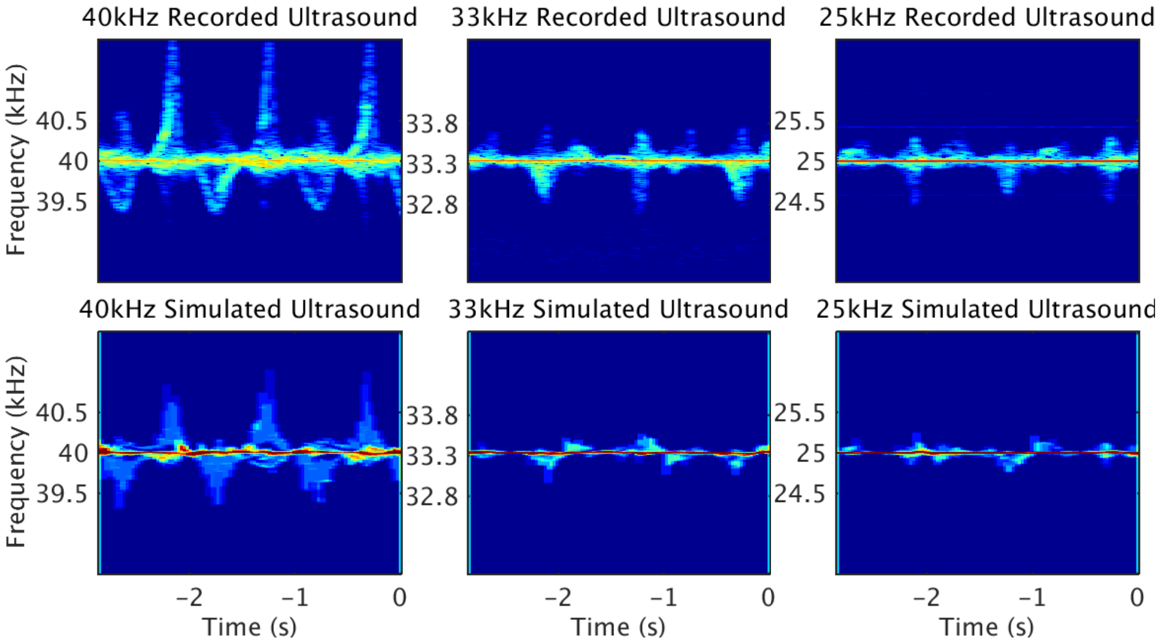


Figure 6.25: The first row shows an example of raw ultrasound data for the punch forward (N) action. The second row shows the spectrogram predicted by the Doppler physics simulation for each of the ultrasound sensors.

Chapter 7

A Generative Model for Doppler-Modulations of Action Sequences

In this chapter, the conditional deep belief network (CDBN), a type of deep neural network architecture that incorporates a limited temporal history, is developed to interface with the JHUMMA dataset. The CDBN is a more sophisticated probabilistic graphical model than the HMM utilized in Chapter 5 and is capable of hallucinating more convincing micro-Doppler modulations. Deep neural networks have had significant success in recent years[67] learning hierarchical representations of sensory input. The CDBN was originally developed in[68] and used to model motion capture data.

In this chapter, the CDBN is used to learn representations of both the ultrasound modulations and the skeletal poses. The model that learns a direct representation of the ultrasound modulations can generate novel sequences, however, the model does not offer any control over parameters related to the environment or sensor configuration. On the other hand, the model that learns a direct representation of skeletal poses can be configured to generate novel sequences for different situations. The rotational skeletal representation allows the sequences to be adapted to different humans by swapping the limb lengths. Moreover, the generated sequences can be coupled with the Doppler-physics model developed in Chapter 6, which allows the sensor geometry to be configurable. This combination of models provides a framework for generating new examples of ultrasound modulations that can be adapted to new situations.

7.1 Restricted Boltzmann Machine

Deep belief networks are generative statistical models traditionally consisting of a hierarchical layering of restricted Boltzmann machines (RBMs). An RBM is a particular type of undirected probabilistic graphical model. Undirected probabilistic graphical models are also known as Markov networks, or Markov random fields (MRFs), and were originally developed in the field of statistical physics.[51,69] In an MRF, the random variables $X_1, ..., X_N$ are represented by nodes and the edges between the nodes indicate a direct probabilistic interaction between the two random variables. The interaction between random variables, for example X_i and X_j, is captured by a factor $\phi(X_i, X_j)$, that is a function of the variables. In general factors are simply functions that map a set of random variables to a nonnegative value and are not limited to pairwise terms. Figure 7.1 illustrates a simple MRF that has five random variables. In this example, the graph is fully connected, with a factor linking every pair of random variables.

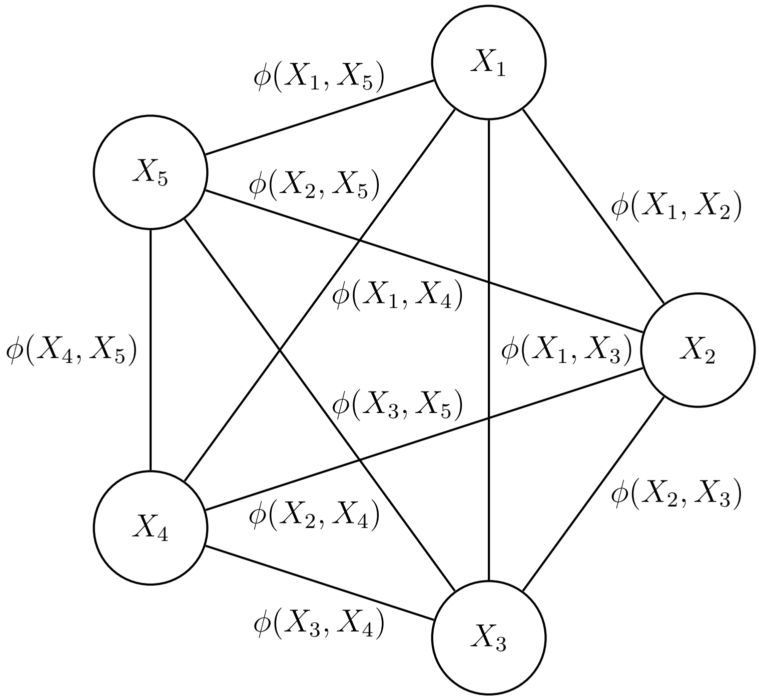


Figure 7.1: Diagram of a simple MRF.

The joint probability distribution of any MRF can be expressed as a normalized product of all the factors. Suppose that an MRF has N random variables, $\mathbf{X} = X_1, ..., X_N$, and K factors, $\Phi = \{\phi_1(D_1), ..., \phi_K(D_K)\}$, where D_j is some set of the random variables $\mathbf{X}$. Then the joint distribution of the MRF is

$$P_\Phi(X_1, ..., X_N) = \frac{1}{Z} \cdot \tilde{P}_\Phi(X_1, ..., X_N), \tag{7.1}$$

where $\tilde{P}_\Phi(X_1, ..., X_N) = \prod_{j=1}^{K} \phi_j(D_j)$ and $Z = \sum_{X_1,...,X_N} \tilde{P}_\Phi(X_1, ..., X_N)$. In general, $\tilde{P}_\Phi(X_1, ..., X_N)$ is an unnormalized distribution and the partition function Z is used to normalize it. For most MRFs, the partition function is notoriously difficult to compute.

If the sum of the log factors is substituted into Equation 7.1, then it is recognizable as the Gibbs equation, which is in fact the form of the joint probability for any MRF. The log of a factor is generally referred to as a potential function. Thus,

$$\begin{aligned} P_\Phi(X_1, ..., X_N) &= \frac{1}{Z} \cdot \prod_{j=1}^{K} \phi_j(D_j) && (7.2) \\ &= \frac{1}{Z} \cdot \exp\Big(\sum_{j=1}^{K} \log \phi_j(D_j)\Big) && (7.3) \\ &= \frac{1}{Z} \cdot \exp^{-E(X_1,...,X_N)}, && (7.4) \end{aligned}$$

where the energy function $E(X_1, ..., X_N) = -\sum_{j=1}^{K} \log \phi_j(D_j)$.

A Boltzmann machine (BM)[70] is an MRF where the factors between the random variables that compose the energy function have been restricted to linear functions. The random variables are also typically constrained to be binary and are differentiated by whether they are observed, called visible nodes, or unobserved, called hidden nodes.

Therefore the energy function is restricted to the form,

$$\begin{gathered} E(H_1 = h_1, ..., H_M = h_M, V_1 = v_1, ..., V_N = v_N) = \\ -\sum_{i=1}^{N} \sum_{j=1}^{M} W_{ij} v_i h_j - \sum_{i=1}^{M} a_i v_i - \sum_{j=1}^{N} b_j h_j \\ -\sum_{j=1}^{M} \sum_{k=1}^{M} U_{jk} h_j h_k - \sum_{i=1}^{N} \sum_{k=1}^{N} V_{ik} v_i v_k . \end{gathered} \tag{7.5}$$

There are N visible variables and M hidden variables. Here, W_{ij} are the connection weights between hidden nodes and visible nodes. The visible node bias terms are a_i and the hidden node bias terms are b_i. The parameters U_{jk} and V_{ik} are the connection weights between hidden nodes and the connection weights between visible nodes, respectively. The bias terms can also be cast as connection weights by including a node, whose value is always one, and connecting each other node to it via the bias connection. Even with these restrictions, BMs are capable of capturing complex distributions quite accurately. Unfortunately, it is very difficult to do learning efficiently in BMs. However, if additional assumptions are made about the structure of the network, the learning process can be greatly simplified and implemented efficiently.

In particular, if the visible and hidden random variables in a BM are separated into two disjoint sets such that the connections in the network are restricted to only involve variables from opposite sets, then learning can be implemented quite efficiently via approximate Markov chain Monte Carlo (MCMC) sampling methods. An MRF with this network topology condition is known as a restricted Boltzmann machine (RBM).

Figure 7.2 illustrates a simple RBM, where the five nodes have been split into a group of three hidden nodes and two visible nodes. In this arrangement, the bipartite structure of an RBM, that restricts connections to only exist between the two sets, is evident. The reason this structure is critical for efficiently implementing learning algorithms is that all of the nodes in one set are independent of each other given all the nodes in the other set. This allows for blocked sampling algorithms that can sample an entire set of variables at once, given the other set of variables. Without the independence structure asserted by the restricted connections, sampling each node would generally be a serial process.

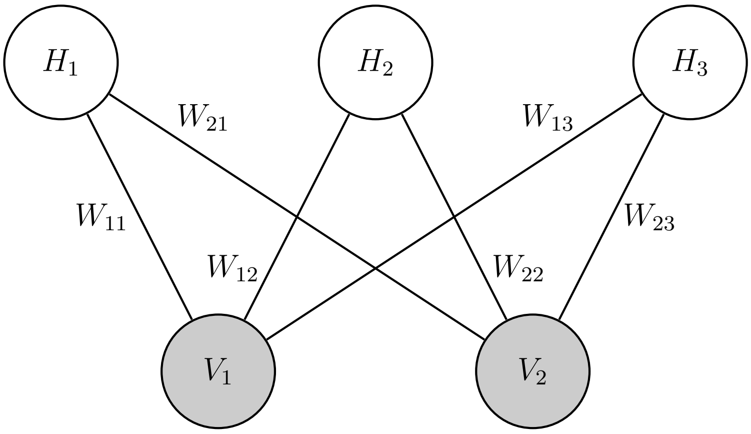


Figure 7.2: Diagram of a simple RBM.

Given the bipartite structure restricting the connections between the hidden and visible random variables, the energy function of an RBM is,

$$\begin{gathered} E(H_1 = h_1, ..., H_M = h_M, V_1 = v_1, ..., V_N = v_N) = \\ -\textstyle\sum_{i=1}^{N} \sum_{j=1}^{M} W_{ij} v_i h_j - \sum_{i=1}^{N} a_i v_i - \sum_{j=1}^{M} b_j h_j. \end{gathered} \tag{7.6}$$

Here, W_{ij} are the connection weight parameters between the ith node in the visible set and the jth node in the hidden set. The visible bias parameters are a_i and the hidden bias parameters are b_j. In summary, the parameters for an RBM are $\boldsymbol{\theta} = (\mathbf{W}, \mathbf{a}, \mathbf{b})$.

Although RBMs are just a particular type of MRF, they are often used as building blocks in neural network models because they can be interpreted as the connections between two layers of neurons. Historically, the visible set of random variables is labeled as such because those are the random variables that are mapped to input features of the data. Then the hidden random variables are trained to turn on and off the connection weights such that correlations between visible feature nodes are learned by the RBM.

In Section 7.2, the RBM is extended to include a notion of temporal history, called a conditional restricted Boltzmann machine (CRBM). In Section 7.3, the restriction to binary random variables is relaxed, allowing the visible layer to represent real-valued input. However, the hidden random variables are still restricted to binary values. In Section 7.4, multiple CRBM layers are combined in a hierarchy to form a conditional deep belief network (CDBN), where the output of the first layer is used to train the second layer.

7.2 Conditional Restricted Boltzmann Machine

The conditional restricted Boltzmann machine (CRBM) is an extension of the RBM that incorporates a temporal history.[68,71] Consider a single input vector $\mathbf{v}_t \in \{0,1\}^N$ at time step t. The vector contains N features, which are mapped to the visible random variables $\mathbf{V}_t$ in the current time slice. There are *undirected* connections between these visible units and the hidden units $\mathbf{H}_t \in \{0,1\}^M$. Alone, these connections form an unaltered RBM at for the input vector at time step t. However, the CRBM incorporates additional *directed* autoregressive connections from the input vectors at the τ previous time steps. This necessitates a temporal sequence to the input data. The variable τ indicates the size of the temporal history buffer used by the CRBM and is refered to as the autoregressive order of the model. Figure 7.3 illustrates the graphical architecture of the model.

The autoregressive directed connections from previous visible time steps to both the current visible units $\mathbf{V}_t$ and the current hidden units $\mathbf{H}_t$ in the CRBM can be thought of as dynamic biases. Given the temporal history $\mathbf{V}_{t-\tau}, ..., \mathbf{V}_{t-1}$, then the

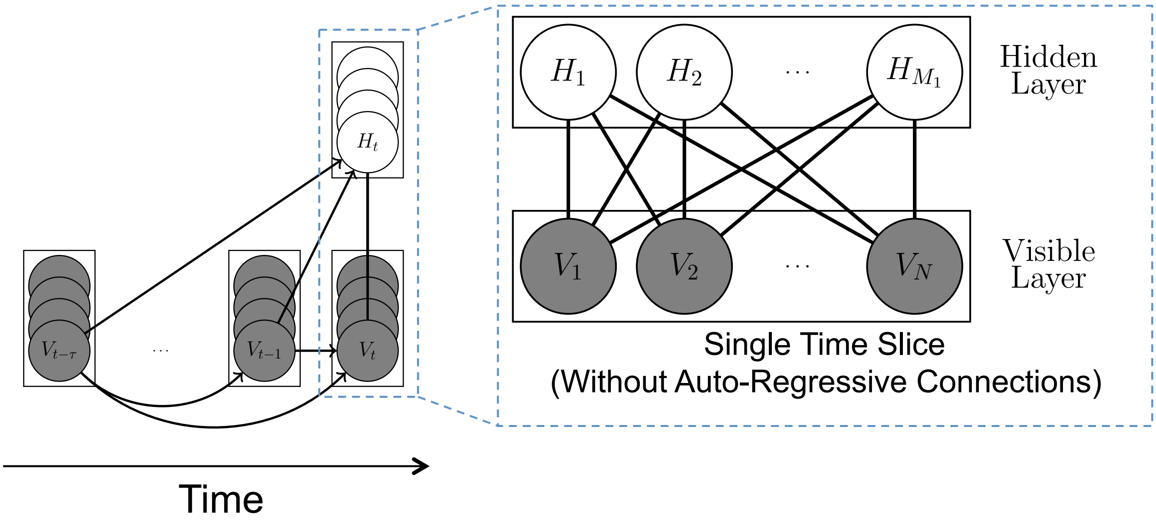


Figure 7.3: Diagram of a CRBM.

energy function of the CRBM is

$$E(\mathbf{V}_t = \mathbf{v}_t, \mathbf{H} = \mathbf{h}_t | \mathbf{V}_{t-1,...,t-\tau} = \mathbf{v}_{t-1,...,t-\tau}) =$$

$$-\underbrace{\sum_{i=1}^{N}\sum_{j=1}^{M} W_{ij} v_i h_j}_{\text{Undirected Connections}} - \underbrace{\sum_{i=1}^{N}\Big(a_i + \overbrace{\sum_{k=1}^{\tau}\mathbf{A}_{ki}\mathbf{v}_{t-k}}^{\text{Visible to Visible Autoregressive Connections}}\Big)v_i}_{\text{Visible Biases}} - \underbrace{\sum_{j=1}^{M}\Big(b_j + \overbrace{\sum_{k=1}^{\tau}\mathbf{B}_{kj}\mathbf{v}_{t-k}}^{\text{Visible to Hidden Autoregressive Connections}}\Big)h_j}_{\text{Hidden Biases}} . \tag{7.7}$$

Note that $\mathbf{v}_t = v_1, ..., v_N$ and $\mathbf{h}_t = h_1, ..., h_M$. As the temporal history is given, the contribution of the directed autoregressive connections can be directly computed and added to the appropriate bias terms. The parameters of a CRBM are $\boldsymbol{\theta} = (\mathbf{W}, \mathbf{a}, \mathbf{b}, \mathbf{A}, \mathbf{B})$, where the autoregressive connection weights are actually sets of weights, one matrix for each time step in the buffer.

7.3 Accommodating Real-Valued Input

One way of accommodating real-valued random variables in the set of visible nodes is the Gaussian CRBM (GCRBM). The energy function of the GCRBM,

$$\begin{aligned} E(\mathbf{V}_t = \mathbf{v}_t, \mathbf{H} = \mathbf{h}_t | \mathbf{V}_{t-1,\ldots,t-\tau} = \mathbf{v}_{t-1,\ldots,t-\tau}) = \; & \tfrac{1}{2} \textstyle\sum_{i=1}^{N} \frac{\left(v_i - a_i - \sum_{k=1}^{\tau} \mathbf{A}_{ki} \mathbf{v}_{t-k}\right)^2}{\sigma_i^2} \\ & - \textstyle\sum_{i=1}^{N} \sum_{j=1}^{M} W_{ij} \frac{v_i}{\sigma_i^2} h_j \\ & - \textstyle\sum_{j=1}^{M} \left(b_j + \sum_{k=1}^{\tau} \mathbf{B}_{kj} \mathbf{v}_{t-k}\right) h_j, \end{aligned} \tag{7.8}$$

is derived in a similar manner to the RBM.[71]

In principle, the parameters σ_i can be learned during training, but it is not a straightforward procedure.[72] In practice, it is preferable to normalize the features of the input data so that they are each zero-mean and have a standard deviation $\sigma_i = 1$.

7.4 Conditional Deep Belief Network

In order to learn more complex distributions, RBMs are often stacked in layers such that the hidden random variables in the first layer are the visible random variables in the second layer. This procedure can include many layers stacked on top of each other and is called a deep belief network (DBN).

The same procedure applies to CRBMs. In this work, two layers, which is the simplest model that can learn hierarchical representations, are used. Figure 7.4 il-

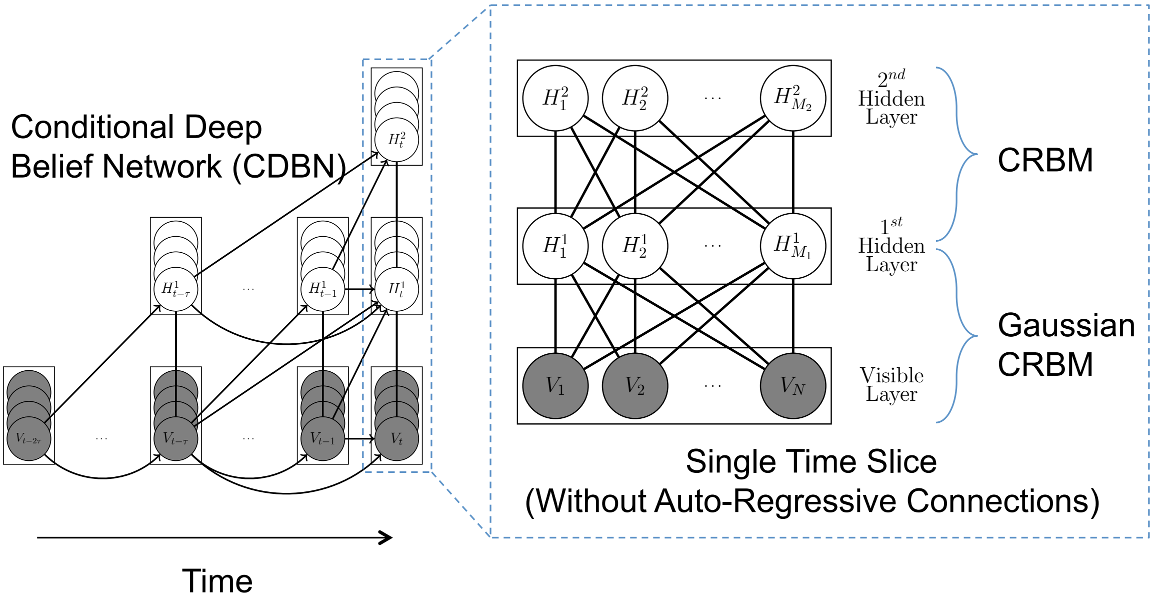


Figure 7.4: Diagram of a CDBN.

lustrates the architecture of the Conditional DBN (CDBN). The bottom layer is a GCRBM, while the second layer is regular binary CRBM. Note that if any more layers were included in the CDBN, they would also be binary CRBMs so that the hidden layer of the lower CRBM is compatible with the visible layer of the CRBM on top of it.

Another aspect of the CDBN is the temporal history buffer required to support multiple layers. In Figure 7.4, each layer has the same autoregressive order, τ. In general, each layer could have a different autoregressive order. However, the overall temporal buffer required to support all of the layers is the sum of the autoregressive orders for each layer. The reason is that a sufficient buffer must exist to support enough time steps in the first hidden layer so that there is enough of a temporal buffer for the second hidden layer. This results in a temporal "fanout" in terms of the number of buffered time steps in each layer. This is of practical concern because most sequential data is collected as a time series. Therefore, the first samples cannot be used to train the model because they do not have enough previous samples to support the required temporal buffer. In the case of Figure 7.4, the first $2 \times \tau$ data points are not eligible to train the model.

7.5 Training Model Parameters

Usually the parameters of a probabilistic graphical model are trained using maximum likelihood estimation (MLE). Unfortunately, there is no analytical MLE solution for the parameters of the Gibbs distribution. In such cases, the usual approach is to fall back on gradient ascent and maximize the likelihood of the observed, or visible, units.

The joint probability of any MRF with observed visible nodes and unobserved hidden nodes is,

$$P_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v},\mathbf{H}=\mathbf{h}) = \frac{1}{Z}\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v},\mathbf{H}=\mathbf{h})\big), \tag{7.9}$$

where the partition function $Z = \sum_{\mathbf{v}'}\sum_{\mathbf{h}'}\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big)$. In order to make the notation more concise, the realizations of the hidden and visible random variables are omitted from the remainder of the calculations in this section. The marginal distribution of the visible nodes is found by summing out the hidden variables in the joint distribution. This results in,

$$\begin{aligned} P_{\boldsymbol{\theta}}(\mathbf{V}) &= \sum_{\mathbf{h}'} P_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H}) \\ &= \frac{1}{Z}\sum_{\mathbf{h}'} \exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big). \end{aligned} \tag{7.10}$$

The conditional distribution of the hidden nodes given the visible nodes is,

$$\begin{aligned} P_{\boldsymbol{\theta}}(\mathbf{H}|\mathbf{V}) &= \frac{P_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})}{P_{\boldsymbol{\theta}}(\mathbf{V})} \\ &= \frac{\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big)}{\sum_{\mathbf{h}'}\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big)} \end{aligned} \tag{7.11}$$

The log-likelihood of the MRF is,

$$\begin{aligned} \mathcal{L}_{\mathbf{V}}(\boldsymbol{\theta}) &= \log\big(P_{\boldsymbol{\theta}}(\mathbf{V})\big) \\ &= \log\Big(\sum_{\mathbf{h}'}\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big)\Big) \\ &\quad -\log\Big(\sum_{\mathbf{v}'}\sum_{\mathbf{h}'}\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big)\Big). \end{aligned} \tag{7.12}$$

Taking the derivative of the log-likelihood yields,

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\theta}}\mathcal{L}_{\mathbf{V}}(\boldsymbol{\theta}) &= \frac{\partial}{\partial \boldsymbol{\theta}}\log\Big(\sum_{\mathbf{h}'}\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big)\Big) \\
&\quad -\frac{\partial}{\partial \boldsymbol{\theta}}\log\Big(\sum_{\mathbf{v}'}\sum_{\mathbf{h}'}\exp\big(-E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\big)\Big) \\
&= -\sum_{\mathbf{h}'}P_{\boldsymbol{\theta}}(\mathbf{H}|\mathbf{V})\frac{\partial}{\partial \boldsymbol{\theta}}E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H}) \\
&\quad +\sum_{\mathbf{v}'}\sum_{\mathbf{h}'}P_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H})\frac{\partial}{\partial \boldsymbol{\theta}}E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H}) \\
&= -\sum_{\mathbf{h}'}P_{\boldsymbol{\theta}}(\mathbf{H}|\mathbf{V})\frac{\partial}{\partial \boldsymbol{\theta}}E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H}) \\
&\quad +\sum_{\mathbf{v}'}P_{\boldsymbol{\theta}}(\mathbf{V})\sum_{\mathbf{h}'}P_{\boldsymbol{\theta}}(\mathbf{H}|\mathbf{V})\frac{\partial}{\partial \boldsymbol{\theta}}E_{\boldsymbol{\theta}}(\mathbf{V},\mathbf{H}). \qquad (7.13)
\end{aligned}$$

Note that both the joint distribution of the MRF and the conditional distribution of the hidden nodes given the visible nodes, both of which were defined earlier, end up being components of the derivative of the log-likelihood. Unfortunately, evaluating the first term of the log-likelihood requires iterating over all possible values of the hidden variables and the second term requires iterating over all possible values of both the hidden and the visible variables, is not tractable. To avoid this computational burden, Gibbs sampling is used to estimate the log-likelihood.

Due to the bipartite structure of the connections between the visible and hidden nodes in a CRBM, each of the hidden nodes are independent from each other given the visible nodes, and each of the visible nodes are independent from each other given the hidden nodes. Therefore, the conditional distributions of the CRBM factorize as,[73]

$$P_{\boldsymbol{\theta}}(\mathbf{H}_t|\mathbf{V}_t, ..., \mathbf{V}_{t-\tau}) = \prod_{j=1}^{M} P(H_j|\mathbf{V}_t, ..., \mathbf{V}_{t-\tau}), \tag{7.14}$$

and

$$P_{\boldsymbol{\theta}}(\mathbf{V}_t|\mathbf{H}_t, \mathbf{V}_{t-1}, ..., \mathbf{V}_{t-\tau}) = \prod_{i=1}^{N} P(V_i|\mathbf{H}_t, \mathbf{V}_{t-1}, ..., \mathbf{V}_{t-\tau}). \tag{7.15}$$

For the CRBM, these single node conditional densities[71] are given by,

$$P(H_j|\mathbf{V}_t, ..., \mathbf{V}_{t-\tau}) = \sigma\Big(\sum_{i=1}^{N} W_{ij} v_{t,i} + b_j + \sum_{k=1}^{\tau} \mathbf{B}_{kj} \mathbf{v}_{t-k}\Big) \tag{7.16}$$

and

$$P(V_j|\mathbf{H}_t, \mathbf{V}_{t-1}, ..., \mathbf{V}_{t-\tau}) = \sigma\Big(\sum_{j=1}^{M} W_{ij} h_j + a_i + \sum_{k=1}^{\tau} \mathbf{A}_{ki} \mathbf{v}_{t-k}\Big), \tag{7.17}$$

where $\sigma(\cdot)$ denotes the sigmoid function.

Using these simplified distributions for the CRBM, the derivative of the log-likelihood with respect to the undirected connection weights is,

$$\begin{aligned}
\frac{\partial \mathcal{L}_{\mathbf{V}=\mathbf{v}}(\boldsymbol{\theta})}{\partial W_{ij}} &= -\sum_{\mathbf{h}'} P_{\boldsymbol{\theta}}(\mathbf{H}=\mathbf{h}'|\mathbf{V}=\mathbf{v}) \frac{\partial E_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v},\mathbf{H}=\mathbf{h}')}{\partial W_{ij}} \\
&\quad + \sum_{\mathbf{v}'}\sum_{\mathbf{h}'} P_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v}',\mathbf{H}=\mathbf{h}') \frac{\partial E_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v}',\mathbf{H}=\mathbf{h}')}{\partial W_{ij}} \\
&= \sum_{\mathbf{h}'} P_{\boldsymbol{\theta}}(\mathbf{H}=\mathbf{h}'|\mathbf{V}=\mathbf{v}) v_j h_i \\
&\quad - \sum_{\mathbf{v}'} P_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v}') \sum_{\mathbf{h}'} P_{\boldsymbol{\theta}}(\mathbf{H}=\mathbf{h}'|\mathbf{V}=\mathbf{v}') v_i h_j \\
&= P_{\boldsymbol{\theta}}(H_j=1|\mathbf{V}=\mathbf{v}) v_i \\
&\quad - \sum_{\mathbf{v}'} P_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v}') P_{\boldsymbol{\theta}}(H_j=1|\mathbf{V}=\mathbf{v}') v_i, \qquad (7.18)
\end{aligned}$$

where the realizations of the random variables have been specified to avoid confusion with respect to the summations. Note that this derivation assumes binary valued random variables. The derivative of the log-likelihood can be computed in a similar manner for the CRBM bias parameters.[71] This is the derivative for a single training example $\mathbf{v}$. Given a full set of training data, $\mathcal{D} = \{\mathbf{v}_1, ..., \mathbf{v}_D\}$, that has D examples, the derivative of the undirected connection weights is,

$$\begin{aligned}
\tfrac{1}{D}\textstyle\sum_{\mathbf{v}'\in\mathcal{D}} \tfrac{\partial \mathcal{L}_{\mathbf{V}=\mathbf{v}'}(\boldsymbol{\theta})}{\partial W_{ij}} = \tfrac{1}{D}\textstyle\sum_{\mathbf{v}'\in\mathcal{D}} \Big(& \mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{H}=\mathbf{h}|\mathbf{V}=\mathbf{v}')Q(\mathbf{V}=\mathbf{v}')}[v_i h_j] \\
- & \mathbb{E}_{P_{\boldsymbol{\theta}}(\mathbf{V}=\mathbf{v},\mathbf{H}=\mathbf{h})}[v_i h_j]\Big),
\end{aligned} \qquad (7.19)$$

where the distribution $Q(\mathbf{V}=\mathbf{v}')$ is the true, but unknown distribution of the training data. Unfortunately, this the second term is still to difficult to sample from because it would require running a Gibbs sampling chain until it converges to the true distribution of the CRBM, $P_{\boldsymbol{\theta}}(\mathbf{V}, \mathbf{H})$.

Fortunately, Hinton[72] developed an approximation for the second term computed using samples from the CRBM after just k iterations of blocked Gibbs sampling.[51] The approximation works well in practice for training the CRBM parameters with gradient ascent. In fact, RBMs are often trained with a single sampling step such that $k = 1$. This approximation for the log-likelihood gradient is known as the k-step contrastive divergence (CD$_k$), defined by

$$\begin{aligned} \mathrm{CD}_k(\boldsymbol{\theta}, \boldsymbol{v}^{(0)}) &= -\sum_{\mathbf{h}} P_{\boldsymbol{\theta}}(\mathbf{H} = \mathbf{h} | \mathbf{V} = \boldsymbol{v}^{(0)}) \frac{\partial E_{\boldsymbol{\theta}}(\mathbf{V} = \boldsymbol{v}^{(0)}, \mathbf{H} = \mathbf{h})}{\partial \boldsymbol{\theta}} \\ &\quad + \sum_{\mathbf{h}} P_{\boldsymbol{\theta}}(\mathbf{H} = \mathbf{h} | \mathbf{V} = \boldsymbol{v}^{(k)}) \frac{\partial E_{\boldsymbol{\theta}}(\mathbf{V} = \boldsymbol{v}^{(k)}, \mathbf{H} = \mathbf{h})}{\partial \boldsymbol{\theta}}. \end{aligned} \tag{7.20}$$

Here the superscript $(0, ..., k)$ denotes the number of sampling steps required to produce a particular sample. Samples are produced by initializing $\boldsymbol{v}^{(0)}$ to a training example $\boldsymbol{v} \in \mathcal{D}$. Then Equations 7.16 and 7.17 can be used iteratively to generate the sequence of samples

$$\boldsymbol{v}^{(0)} \Rightarrow \boldsymbol{h}^{(1)} \Rightarrow \boldsymbol{v}^{(1)} \Rightarrow \boldsymbol{h}^{(2)} \Rightarrow \boldsymbol{v}^{(2)} \Rightarrow \ldots \Rightarrow \boldsymbol{h}^{(k)} \Rightarrow \boldsymbol{v}^{(k)}. \tag{7.21}$$

Using CD$_k$, iterative gradient ascent to update the parameters of the CRBM can be performed according to the following update rule,[71]

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \overbrace{\underbrace{\eta \mathrm{CD}_k(\boldsymbol{\theta}, \boldsymbol{v}^{(0)})}_{\substack{\text{Contrastive Divergence}\\ \text{Gradient Approximation}}} + \underbrace{\lambda \boldsymbol{\theta}^{(t)}}_{\text{Weight Decay}} + \underbrace{\nu \Delta \boldsymbol{\theta}^{(t-1)}}_{\text{Momentum}}}^{\Delta \boldsymbol{\theta}^{(t)}}. \tag{7.22}$$

These parameter updates include a weight decay regularization term and a momentum term that promotes larger step sizes in areas where the gradient is large.

In practice, the standard approach to unsupervised training in a CDBN is to train the parameters of each CRBM layer one at a time in a greedy fashion. The first layer to be trained is the bottom layer that interfaces directly with the features in the training set $\mathcal{D}$. Equation 7.22 is used to update the parameters for a set number of iterations, or epochs, through the training set. Once the parameters for the first layer are trained they are held fixed. The training set is then propagated through the first layer to produce a transformed training set that is used to train the parameters of the second layer in the same way as those of the first. This procedure is repeated until all layers of the CDBN are trained.

7.6 Hallucinating Novel Doppler Modulations from Ultrasound

In order to use the CDBN to generate, or hallucinate, novel examples of motions, a network is trained on data from the JHUMMA dataset for a set of actions. In principle, a network could be trained on the entire dataset, but the real valued visible units in the Gaussian CRBM have difficulty modeling the variation across all twenty-one actions and thirteen trials in the dataset. In order for the real-valued data to be processed by the Gaussian CRBM, each feature is normalized across the training data to a standard normal. Unfortunately, this representation is not the best model for the variability in the data across entire JHUMMA dataset. Therefore, smaller models were trained to demonstrate the capabilities of the system on smaller subsets of actions, where the variability is less extreme.

Once a CDBN is trained, generating a new sequence from the distribution simply requires a seeded history buffer. By buffering with data from a particular action, it is highly likely that the hallucinated sequence will correspond to that action, although there is nothing preventing the model from switching to another action that it was trained on. Once the buffer is fixed, blocked Gibbs sampling is used to generate a novel sequence from the distribution encoded by the CDBN. This procedure uses the same sampling equations described in Section 7.5 to iteratively update the visible and hidden layers. For all of the generated sequences shown in this chapter, 30 iterations of blocked Gibbs sampling were used to hallucinate the visible layer at each time step.

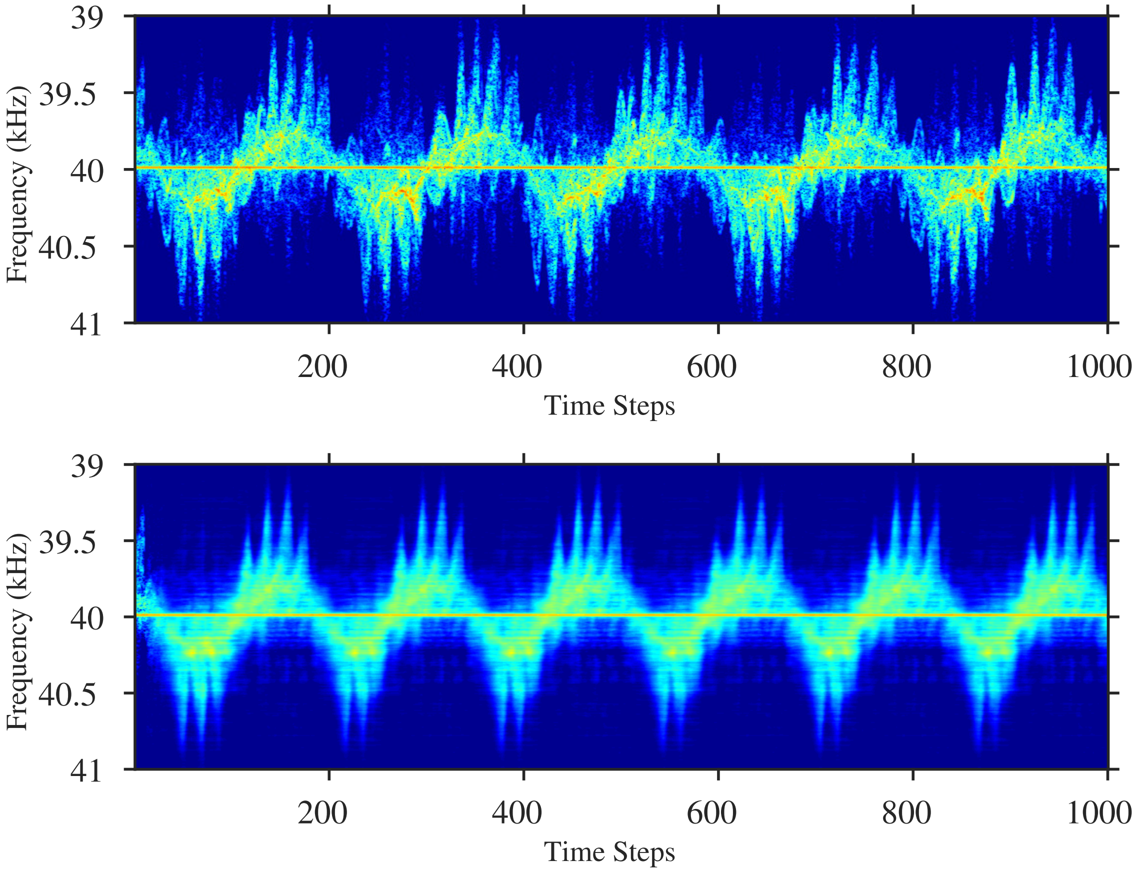


Figure 7.5: The top spectrogram shows a recorded ultrasound sequence of walk and pivot data. The first 14 frames were used to seed the CRBM and sampling the model produced the hallucination shown in the bottom spectrogram.

The lower spectrogram of Figure 7.5 shows a sequence of ultrasound modulations generated from a two layer CDBN trained on 40kHz ultrasound data. The autoregressive order of the CDBN was set to $\tau = 7$ time steps. Both layers shared the same autoregressive order, so the model was seeded with the first 14 spectrogram slices of the 40kHz ultrasound test sequence shown in the upper spectrogram of Figure 7.5. The visible layer was composed of 328 real-valued nodes that map directly to the spectrogram frequency bins. Each of the two hidden layers contained 300 binary nodes. The model parameters were trained on $3,790$ spectrogram slices of the "walk and pivot" action from the JHUMMA dataset. These slices were pulled from two trials of the same actor.

Figure 7.5 demonstrates that the CDBN model does a significantly better job of modeling longterm temporal correlations than the HMM model developed in Chapter 5. For comparison, the HMM generated sequence for the "walk and pivot" action is shown in Figure 5.29. Due to the finite state space limitations of the HMM, the state space would have to grow very large to accommodate the same temporal history and still represent the set of skeletal poses well.

With only a very short window of seed data, the CDBN model recognizes the walking pattern and is able to hallucinate a reasonable set of modulations. In particular it accurately captures the overall cadence of the walking motion and the back and forth movement of the torso. However, it fails to maintain the proper number of sub modulations in each period of the walking motion. This is akin to adding or subtracting limbs from the human. By incorporating a physical notion of the human body, it should be possible to avoid these types of errors.

7.7 Hallucinating Novel Doppler Modulations from Skeletal Sequences

Figure 7.6 shows a sequence of ultrasound modulations generated from a two layer CDBN trained on kinect data for the side arm raise actions from the JHUMMA dataset. The output of the CDBN was then processed by the Doppler-physics model developed in Chapter 6. The autoregressive order of the CDBN was set to $\tau = 7$ time steps. Both layers shared the same autoregressive order, so the model was seeded with 14 skeletal poses. Each of the two hidden layers contained 500 nodes. All 57 rotation features, plus the three translation parameters for the skeleton were learned by the CDBN. The model was trained on $1,961$ skeletal poses derived from both the left and right "arm raises" for a single actor, but seeded with a buffer of left arm raises.

The hallucination produced by the model is remarkable stable over many repetitions. Due to the relative simplicity of the skeletal pose representation, the distribution learned by the CDBN is able to produce a skeletal sequence that approximates the action well. By running the Doppler-physics model three times, where each simulation is configured to mimic one of the three ultrasound sensors used in the JHUMMA dataset, the combination of the two models is able to predict the acoustic modulations for a novel action.

Figure 7.7 shows a sequence of ultrasound modulations generated from a two layer

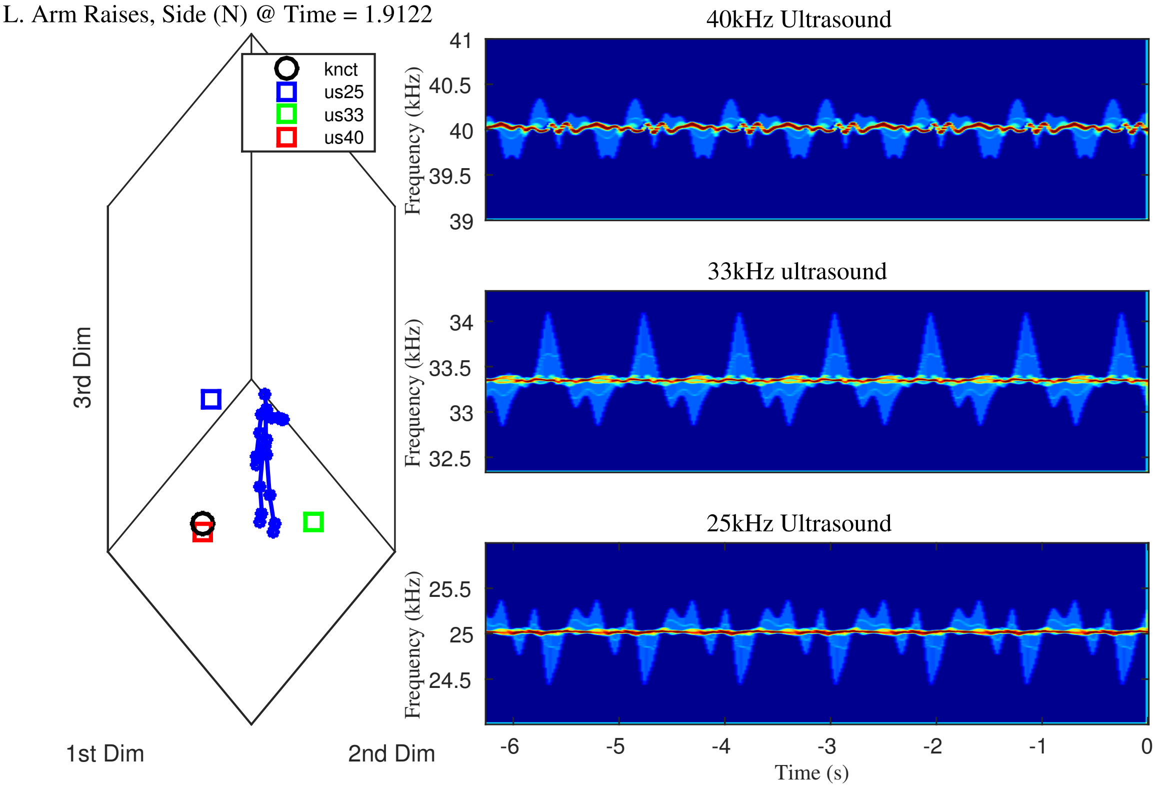


Figure 7.6: Three simulated ultrasound modulations of left arm raises to the side. They were generated by a CDBN trained to model the skeletal motions for these actions and then processed through the Doppler-physics model.

CDBN trained on kinect data for the walk in place action from the JHUMMA dataset. The model parameters were identical to those described for the arm raise model. The model was trained on 5, 150 skeletal poses derived from both "walking in place" and "walking forwards and backwards" for a single actor, but seeded with a buffer of the "walking in place" action.

Figure 7.7 shows a sequence of ultrasound modulations generated from a two layer CDBN trained on kinect data for the walk in place action from the JHUMMA dataset. The model parameters were identical to those described for the arm raise model. The

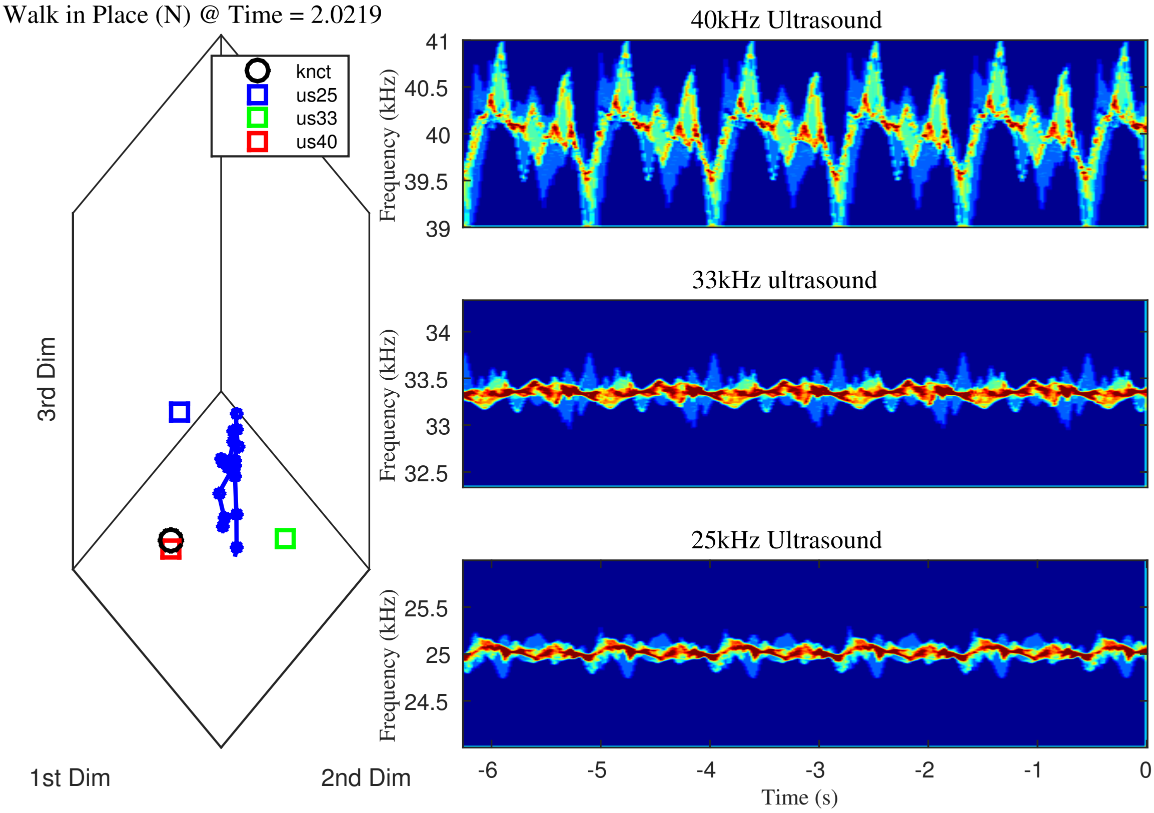


Figure 7.7: Three simulated ultrasound modulations of walking in place. They were generated by a CDBN trained to model the skeletal motions for these actions and then processed through the Doppler-physics model.

model was trained on 3, 790 skeletal poses derived from "walk and pivoting" for a single actor.

The CDBN trained on skeletal frames does a much better job of learning the sequence of individual limb actions for the walking motion. This results in a much more consistent sequence of longterm modulation patterns than the CDBN trained on the ultrasound data.

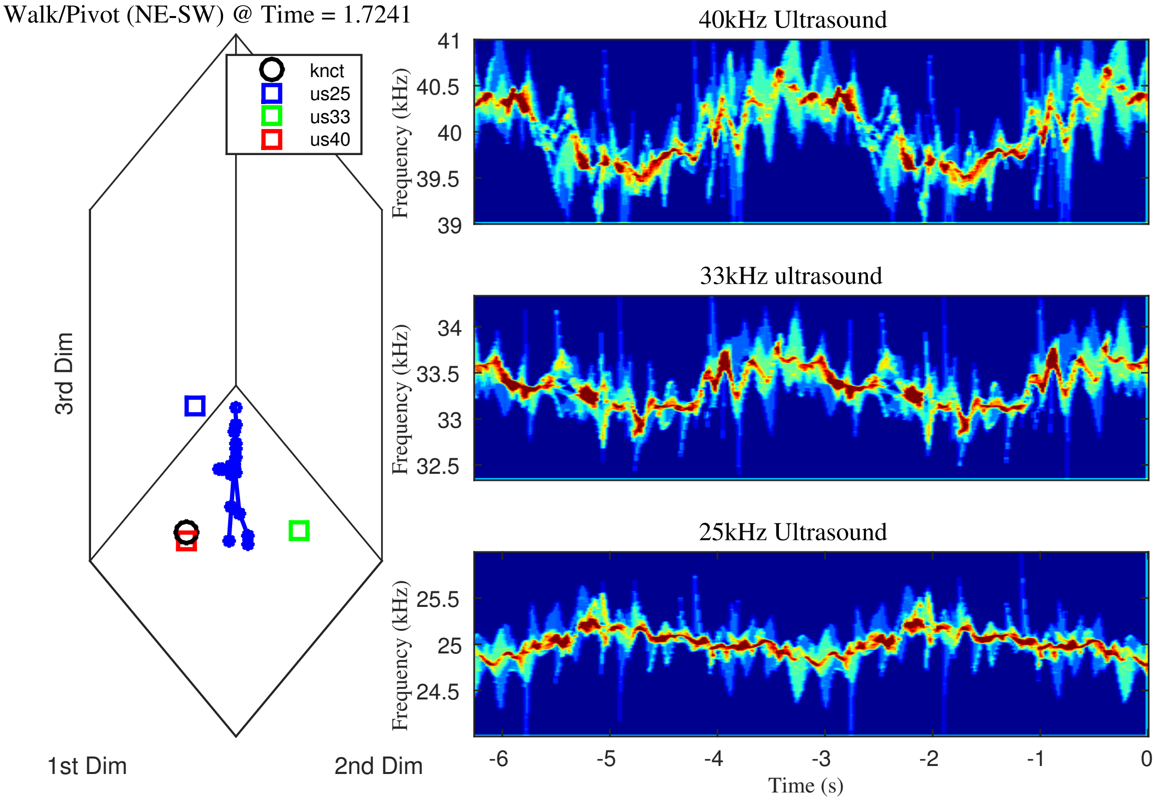


Figure 7.8: Three simulated ultrasound modulations of walking and pivoting. They were generated by a CDBN trained to model the skeletal motions for these actions and then processed through the Doppler-physics model.

7.8 Configuring Limb Lengths

One of the advantages to building a CDBN that learns the underlying skeletal representation is having access to configurable parameters like the human lib lengths. The CDBN has simply learned the relative rotations of the limbs, so adjusting the limb lengths allows for the model to be adapted to mimic people of different sizes. In Figure 7.9, shows the results from doubling and halving the original limb lengths for a walking sequence hallucinated by the CDBN. The associated spectrograms show the output of the Doppler-physics model given each of the three skeleton models. The scaling on the spectrograms is different because it was derived from a demonstration of the Doppler-physics model and a CDBN trained on an older dataset that was not part of the JHUMMA and used a different noise model to scale the background of the Doppler-physics model. However, the implementation of the Doppler-physics model is otherwise identical.

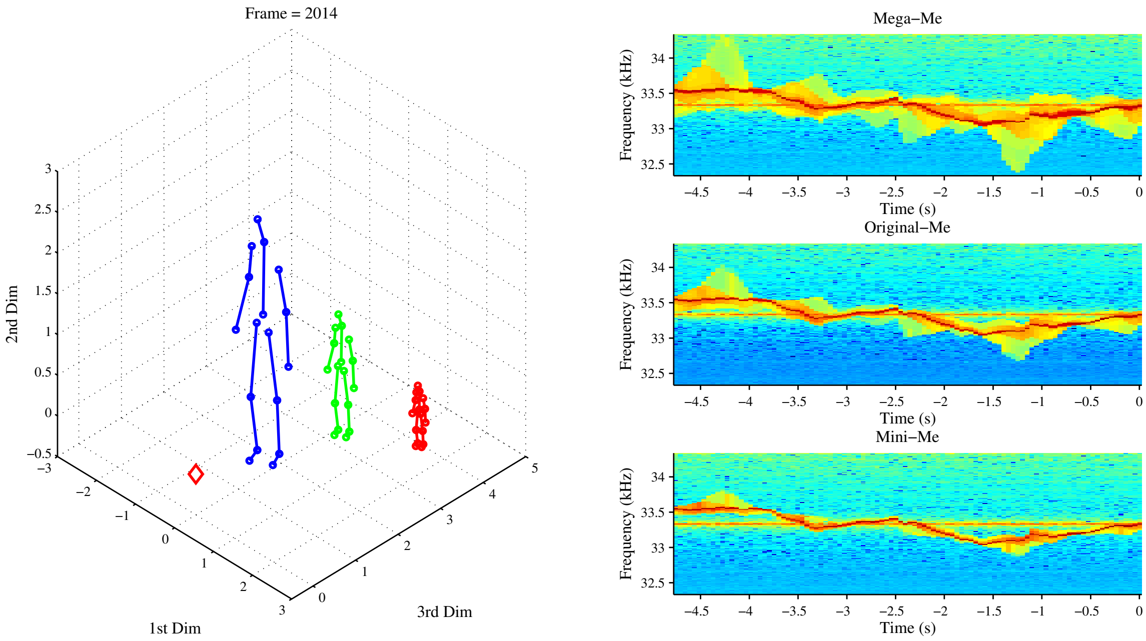


Figure 7.9: Sequence of skeletal walking data hallucinated by a CDBN trained on Kinect skeletal frames and the resulting ultrasound modulations generated by the physics-baed model of a human walking. Two additional sequences were generated by doubling and halving the original limb lengths of the human body model.

The skeleton with the longest limbs generates the fastest limb velocities because it has to move it's limbs a longer distance in the same time period. The Doppler modulations predicted by the physics model have correspondingly larger magnitudes. Conversely, the smaller skeleton has slower limb velocities, which result in smaller Doppler modulations.

Chapter 8

Conclusion

At its core, the framework presented in this dissertation investigates the potential of using low-dimensional impoverished sensor data to make inferences about higher-dimensional natural phenomenon. To overcome the ambiguities associated with making decisions from the incomplete information contained in the low-dimensional data, prior knowledge about the structure and physics of both the environment and the sensor modality is used to constrain and bootstrap the inference procedure.

Specifically, these principles are demonstrated by using low-dimensional active acoustic data to recognize higher-dimensional human actions. In order to facilitate the construction of models, the Johns Hopkins University multimodal action dataset was constructed. It consists of twenty-one actions and focuses on examples of orientational symmetry that a single active ultrasound sensor should have the most difficulty discriminating. The data collection includes recordings from three independent ultrasound sensors, which provides the necessary foundation to explore using data from multiple views to resolve the orientational ambiguity in the dataset. The data collection also includes higher-dimensional RGB-Depth images that are used to track the human movements. This provides a basis for developing the physical models to constrain the inferences made from the acoustic data.

A simple generative probabilistic graphical model is developed and used to classify the actions in the JHUMMA dataset, providing a performance baseline that can be used to guide future work. This action recognizer is based on a set of hidden Markov models (HMM) that are jointly trained on both the low-dimensional acoustics and a skeletal representation of the human body that is extracted from the higher-dimensional RGB-Depth data. The model achieves an action classification rate from 63% to 75% using input from a single ultrasound sensor. When the classifiers built on each individual sensor are combined, the system demonstrates a large gain in performance, reaching over 88% classification accuracy. While the models are well suited to classifying action sequences, they do not perform particularly well when classifying individual skeletal poses. Moreover, they are demonstrated to be unsuitable for generating predictions of novel sequences that could be used in simulations.

The HMM based classifier used the simplest dynamics to capture the sequential nature of the data. An area of future research involves expanding this model to include a more realistic model for the evolution of skeletal poses over time.

One of the primary challenges associated with using acoustic action recognizers is that both the geometry of the sensors, as well as the characteristics of the moving human, greatly affect the modulation patterns recorded in the reflected signals. An important step toward achieving models that are truly adaptable to new situations is the capability to predict what the acoustic modulations will look like at an arbitrary sensor location given a human action. A computationally efficient physics-based simulation, that predicts the acoustic Doppler modulations at configurable sensor locations, was developed.

Although the output of the simulator was demonstrated to be a reasonable prediction of actual action data in the JHUMMA dataset, there are many modifications that can be made to further improve the performance. For instance, accounting for clothes and obstructions would solve several of the models most obvious deficiencies. Moreover, the framework of classification from active can be extended to tasks other than action recognition by building acoustic scattering models for other moving objects.

Finally, conditional deep belief networks (CDBNs), a generative probabilistic deep neural network with autoregressive temporal connections, were developed and used to capture the dynamics of both ultrasound modulations and skeletal motion sequences. The models demonstrate promising capabilities for generating novel sequences in both sensory domains. The CDBN trained on skeletal poses can be coupled with the physics simulation to generate micro-Doppler modulations for novel action sequences that demonstrate realistic long term structure.

The primary contribution of the work presented in this dissertation is a complete implementation of an end-to-end framework for collecting temporally coherent multimodal sensor data and the development of statistical models infused with physical constraints. While each of the components in the system, both hardware and software, can certainly be upgraded to include more advanced techniques, the overall approach demonstrates a consistent methodology for the co-development of highly integrated models for processing novel sensory data.

Bibliography

[1] C. Doppler, "Über das farbige Licht der Doppelsterne und einiger anderer Gestirne des Himmels (English Translation)," *Proceedings of the Royal Bohemian Society of Sciences*, vol. 2, pp. 465–482, 1842.

[2] B. Ballot, "Akustische Versuche auf der Niederländischen Eisenbahn, nebst gelegentlichen Bemerkungen zur Theorie des Hrn. Prof. Doppler," *Annalen der Physik and Chemie*, vol. 11, pp. 321–351, 1845.

[3] V. C. Chen, F. Li, S. S. Ho, and H. Wechsler, "Micro-Doppler effect in radar: phenomenon, model, and simulation study," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 2–21, 2006.

[4] V. Chen, *The Micro-Doppler Effect in Radar*, ser. Artech House Remote Sensing Library. Artech House, Dec. 2010.

[5] J. J. Murray, "A theoretical model of linearly filtered reverberation for pulsed active sonar in shallow water," *The Journal of the Acoustical Society of America*, vol. 136, no. 5, pp. 2523–2531, Nov. 2014.

[6] T. Figliolia, T. S. Murray, and A. G. Andreou, "Acoustic micro-Doppler signal processing with a foveated electronic cochlea," *IET Electronics Letters*, vol. 51, no. 2, pp. 132–134, Jan. 2015.

[7] S. Dura-Bernal, G. Garreau, J. Georgiou, A. G. Andreou, S. L. Denham, and T. Wennekers, "Multimodal integration of micro-Doppler sonar and auditory signals for behavior classification with convolutional networks," *International Journal of Neural Systems*, vol. 23, no. 5, pp. 1 350 021–1 1 350 021–15, 2013.

[8] C. Clemente, A. Balleri, K. Woodbridge, and J. J. Soraghan, "Developments in target micro-Doppler signatures analysis: radar imaging, ultrasound and through-the-wall radar," *EURASIP Journal on Advances in Signal Processing*, vol. 2013, no. 1, pp. 1–18, 2013.

[9] M. Bradley and J. M. Sabatier, "Distinguished between human and equine motion using acoustic micro-Doppler sonar," *The Journal of the Acoustical Society of America -Express Letters-*, pp. 1–17, May 2012.

[10] G. Garreau, N. Nicolaou, and J. Georgiou, "Individual classification through autoregressive modelling of micro-doppler signatures," in *Proceedings of the 2012 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2012, pp. 312–315.

[11] *Human action categorization using ultrasound micro-Doppler signatures*, Nov. 2011.

[12] A. Balleri, K. Woodbridge, and K. Chetty, “Frequency-agile non-coherent ultrasound radar for collection of micro-Doppler signatures,” in *Proceedings of the 2011 IEEE Radar Conference (RADAR)*, 2011, pp. 045–048.

[13] G. Garreau, N. Nicolaou, C. M. Andreou, C. D’Urbal, G. Stuarts, and J. Georgiou, “Computationally efficient classification of human transport mode using micro-doppler signatures,” in *Proceedings of the 45th Annual Conference on Information Sciences and Systems (CISS)*, 2011, pp. 1–4.

[14] A. Balleri, K. Chetty, and K. Woodbridge, “Classification of personnel targets by acoustic micro-Doppler signatures,” *IET Radar, Sonar & Navigation*, vol. 5, no. 9, p. 943, 2011.

[15] G. Garreau, C. M. Andreou, A. G. Andreou, J. Georgiou, S. Dura-Bernal, T. Wennekers, and S. L. Denham, “Gait-based person and gender recognition using micro-doppler signatures,” in *Proceedings of the 2011 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2011, pp. 444–447.

[16] M. Bradley and J. M. Sabatier, “Applications of Fresnel-Kirchhoff diffraction theory in the analysis of human-motion Doppler sonar grams,” *The Journal of the Acoustical Society of America -Express Letters-*, vol. EL248, no. 128 (5), Oct. 2010.

[17] J. Sabatier and A. Ekimov, “A review of human signatures in urban environments

using seismic and acoustic methods," in *IEEE Conference on Technologies for Homeland Security*, 2008, pp. 215–220.

[18] T. Thayaparan, L. Stankovic, and I. Djurovic, "Micro-Doppler-based target detection and feature extraction in indoor and outdoor environments," *Journal of the Franklin Institute*, vol. 345, no. 6, pp. 700–722, 2008.

[19] Z. Zhang and A. G. Andreou, "Human identification experiments using acoustic micro-Doppler signatures," in *Proceedings of the 3rd Argentine School of Micro-Nanoelectronics, Technology and Applications (EAMTA 2008)*, 2008, pp. 81–86.

[20] K. Kalgaonkar and B. Raj, "Ultrasonic Doppler sensor for speaker recognition," in *Proceedings of the 2008 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 4865–4868.

[21] S. S. Ram, Y. Li, A. Lin, and H. Ling, "Doppler-based detection and tracking of humans in indoor environments," *Journal of the Franklin Institute*, vol. 345, no. 6, pp. 679–699, 2008.

[22] K. Kalgaonkar and B. Raj, "Acoustic Doppler sonar for gait recogination," in *IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS 2007)*, Aug. 2007, pp. 27–32.

[23] Z. Zhang, P. O. Pouliquen, A. M. Waxman, and A. G. Andreou, "Acoustic micro-

Doppler radar for human gait imaging," *The Journal of the Acoustical Society of America*, vol. 121, no. 3, pp. EL110–3, Mar. 2007.

[24] ——, "Acoustic micro-Doppler gait signatures of humans and animals," in *Proceedings of the 41st Annual Conference on Information Sciences and Systems (CISS)*, 2007, pp. 627–630.

[25] V. C. Chen, F. Li, S. S. Ho, and H. Wechsler, "Analysis of micro-Doppler signatures," *IEE Proceedings -Radar, Sonar and Navigation-*, vol. 150, no. 4, p. 271, 2003.

[26] V. C. Chen and H. Ling, *Time-frequency transforms for radar imaging.* Artech House, 2002.

[27] P. O. Pouliquen, A. S. Cassidy, G. Garreau, J. Georgiou, and A. G. Andreou, "A wireless architecture for distributed sensing/actuation and pre-processing with microsecond synchronization," in *Proceedings of the 45th Annual Conference on Information Sciences and Systems (CISS)*, Mar. 2011, pp. 1–6.

[28] J. Georgiou, P. O. Pouliquen, A. S. Cassidy, G. Garreau, C. M. Andreou, G. Stuarts, C. d'Urbal, S. L. Denham, T. Wennekers, R. Mill, I. Winkler, T. M. Bohm, O. Szalardy, G. M. Klump, S. Jones, A. Bendixen, and A. G. Andreou, "A multimodal-corpus data collection system for cognitive acoustic scene analysis," in *Proceedings of the 45th Annual Conference on Information Sciences and Systems (CISS)*, Mar. 2011, pp. 1–6.

[29] N. Silberman and R. Fergus, "Indoor scene segmentation using a structured light sensor," in *Proceedings of the 2011 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2011, pp. 601–608.

[30] B. Freedman, A. Spunt, and Y. Ariell, "Distance-varying illumination and imaging technique for depth mapping," Patent, Jun., 2014.

[31] G. Yahav, G. Iddan, and D. Mandelboum, "3D imaging camera for gaming application," in *Consumer Electronics 2007*, 2007, pp. 1–2.

[32] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 1297–1304.

[33] Y. G. Jiang, J. Liu, A. R. Zamir, I. Laptev, M. Piccardi, M. Shah, and R. Sukthankar. (2014, Oct.) THUMOS Challenge 2013. [Online]. Available: http://crcv.ucf.edu/ICCV13-Action-Workshop/

[34] Y. G. Wang, J. Liu, A. R. Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. (2014, Oct.) THUMOS Challenge 2014. [Online]. Available: http://crcv.ucf.edu/THUMOS14/

[35] H. Wang. (2014, Oct.) LEAR- Dense Trajectories Video Description. [Online]. Available: http://lear.inrialpes.fr/people/wang/dense_trajectories

[36] E. Vig, M. Dorr, and D. Cox, "Large-Scale Optimization of Hierarchical Features for Saliency Prediction in Natural Images," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 2798–2805.

[37] S. Mathe and C. Sminchisescu, "Actions in the Eye: Dynamic Gaze Datasets and Learnt Saliency Models for Visual Recognition," *arXiv.org*, Dec. 2013.

[38] K. K. Reddy and M. Shah, "Recognizing 50 human action categories of web videos," *Machine Vision and Applications*, vol. 24, no. 5, Jul. 2013.

[39] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild," *arXiv.org*, Dec. 2012.

[40] S. Mathe and C. Sminchisescu, "Dynamic eye movement datasets and learnt saliency models for visual action recognition," in *Proceedings of the 12th European conference on Computer Vision (ECCV'12)*. Springer-Verlag, Oct. 2012.

[41] C. Wolf, J. Mille, E. Lombardi, O. Celiktutan, M. Jiu, M. Baccouche, E. Dellandréa, C.-E. Bichot, C. Garcia, and B. Sankur, "The LIRIS Human activities dataset and the ICPR 2012 human activities recognition and localization competition," Tech. Rep. LIRIS-RR-2012-004, 2012.

[42] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large

video database for human motion recognition," *IEEE International Conference on Computer Vision. Proceedings*, pp. 2556–2563, Nov. 2011.

[43] M. Marszalek, I. Laptev, and C. Schmid, "Actions in context," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 2929–2936.

[44] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[45] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2247–2253, Dec. 2007.

[46] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, 2004, pp. 32–36.

[47] F. Jelinek, *Statistical methods for speech recognition.* The MIT Press, 1998.

[48] K. Pastra and Y. Aloimonos, "The minimalist grammar of action," *Philosophical Transactions of the Royal Society of London: Series B, Biological Sciences*, vol. 367, no. 1585, pp. 103–117, Nov. 2011.

[49] G. Guerra-Filho and Y. Aloimonos, “A Language for human action,” *IEEE Computer*, vol. 40, no. 5, pp. 42–51, 2007.

[50] ——, “A Sensory-Motor Language for Human Activity Understanding,” in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, 2006, pp. 69–75.

[51] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, 1st ed. The MIT Press, Jul. 2009.

[52] K. P. Murphy, “Dynamic Bayesian Networks: Representation, Inference and Learning,” Ph.D. dissertation, Ph.D. Dissertation, University of California Berkeley, 2002.

[53] K. Murphy, “A brief introduction to graphical models and Bayesian networks,” *http://people.cs.ubc.ca/ murphyk/Bayes/bnintro.html*, 1998.

[54] K. P. Murphy, *Machine Learning: a Probabilistic Perspective.* MIT Press, Sep. 2013.

[55] S. J. Young, G. Evermann, M. J. Gales, T. Hain, D. Kershaw, X. A. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK book.* University of Cambridge, Mar. 2009, vol. ver 3.4.

[56] J. MacQueen, “Some methods for classification and analysis of multivariate

observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability.* Oakland, CA, USA., 1967, pp. 281–297.

[57] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[58] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006.

[59] A. Coates and A. Y. Ng, "Learning Feature Representations with k-means," in *Neural Networks: Tricks of the Trade.* Springer Lectures in Computer Science, 2012, pp. 561–580.

[60] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881–892, 2002.

[61] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms.* Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[62] R. Boulic, N. Thalmann, and D. Thalmann, "A global human walking model with real-time kinematic personification," *The visual computer*, vol. 6, no. 6, pp. 344–358, 1990.

[63] L. V. Blake, “A guide to basic pulse-radar maximum-range calculation,” Tech. Rep. 6930, Dec. 1969.

[64] D. A. Winter, *Biomechanics and Motor Control of Human Movement*, 4th ed. Wiley, Oct. 2009.

[65] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Transactions of the ASME - Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[66] V. Poor, *An Introduction to Signal Detection and Estimation : H. Vincent Poor: 9780387941738: Amazon.com: Books*, 2nd ed., ser. Springer Texts in Electrical Engineering. Springer, Mar. 1998.

[67] Y. Bengio, A. Courville, and P. Vincent, “Representation Learning: A Review and New Perspectives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[68] G. W. Taylor, G. E. Hinton, and S. T. Roweis, “Modeling human motion using binary latent variables,” in *Advances in Neural Information Processing Systems 18 (NIPS-2006)*, 2006, pp. 1345–1352.

[69] E. Ising, “Beitrag zur theorie des ferromagnetismus,” *Zeitschrift für Physik A Hadrons and Nuclei*, vol. 31, no. 1, pp. 253–258, 1925.

[70] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, “A learning algorithm for Boltzmann machines,” *Cognitive science*, vol. 9, no. 1, pp. 147–169, 1985.

[71] G. W. Taylor, G. E. Hinton, and S. T. Roweis, “Two Distributed-State Models For Generating High-Dimensional Time Series,” *Journal of Machine Learning Research*, vol. 12, Feb. 2011.

[72] G. Hinton, “A Practical Guide to Training Restricted Boltzmann Machines,” Tech. Rep. UTML TR 2010-003, Aug. 2010.

[73] Y. Bengio and O. Delalleau, “Justifying and Generalizing Contrastive Divergence,” *Neural Computation*, vol. 21, no. 6, pp. 1601–1621, 2009.

Vita

Thomas Simmons Murray was born on November 1, 1984 in Syracuse, NY. He earned his Bachelor of Science in Engineering, with a minor in statistics, from Swarthmore College in 2007. That fall he began his graduate studies at The Johns Hopkins University, under the mentorship of Andreas G. Andreou. Thomas earned his Masters of Science in Electrical and Computer Engineering from The Johns Hopkins University in 2009.