EXPRESSIVE MOTION

by

Alyssa Lees

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

New York University

September, 2006

Christoph Bregler (co-advisor)

Davi Geiger (co-advisor)

ABSTRACT

Since the advent of motion capture animation, attempts have been made to extract the seemingly nebulously defined attributes of 'content' and 'style' from the motion data. Enabling quick access to highly precise data, the benefits of motion capture for animation purposes are abundant. Yet manipulating the expressive attributes of the motion data in a comprehensive manner has proved elusive. This dissertation poses practical solutions that are based on insights from the dance community and learning attributes from the motion data itself. The culminating project is a system which learns the deformations of the human body and reapplies them in exaggerated form for enhanced expressivity.

While simultaneously developing efficient and usable tools for animators, the result is a three pronged technique to enhance the expressive qualities of motion capture animation. The key aspect is the creation of a deformable skeleton representation of the human body using a unique machine learning approach. The deformable skeleton is modeled by replicating the Utilized by movies, commercials, video games and medical labs, motion capture has become an increasingly popular form of acquiring data for the simulation of human movement. Over the last twenty years, improvements in both equipment and techniques have made animating complex actions trivial. On the other hand, motion capture still has major obstacles to overcome when used

as an animation tool. One review of a current motion capture based movie compared the experience to watching "mannequins act."actual movements of the human spine. The second step relies on exploiting the subtle aspects of motion, such as hand movement to create an emotional effect visually. Both of these approaches involve exaggerating the movements in the same vein as traditional 2-D animation technique of 'squash and stretch'. Finally, a novel technique for the application of style on a baseline motion capture sequence is developed.

All of these approaches are rooted in machine learning techniques. Linear discriminate analysis was initially applied to a single phrase of motion demonstrating various style characteristics in LABAN notation. A variety of methods including nonlinear PCA, and LLE were used to learn the underlying manifold of spine movements. Nonlinear dynamic models were learned in attempts to describe motion segments versus single phrases. In addition, the dissertation focuses on the variety of obstacles in learning with motion data. This includes the correct parameterization of angles, applying statistical analysis to quaternions, and appropriate distance measures between postures.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

A journey through the exhibition hall of the ACM SIGGRAPH 2004 convention readily attested to the increased availability of motion capture data. On the display floor, at least three different competing brands of motion capture systems vied for consumer attention with live demonstrations of suited dancers animating various computer characters in real time. Down the hall at the guerilla studio a temporary motion capture lab was constructed in which SIGGRAPH patrons were allowed to suit up and record data of themselves for personal projects. From the animation theatre to papers, the mainstream use of motion capture data was obvious.

Of course this result is not surprising: the benefits of motion capture are abundant. Due to recent improvement in both equipment and techniques for manipulating data, motion capture has become a means of acquiring instant gratification for character animation. Real time motion capture which maps live recorded motions of an actor directly on to a computer character has become standard. The aspect of time is especially relevant when comparing motion capture to traditional labor intensive animating techniques such as key framing.

Motion capture is defined to be the recording of human or other object for analysis and playback. In the animation community the movement captured is often limited to the position of body parts in space. However, motion capture can be as complex as measuring the deformations of the faces and/or muscles. A limiting attribute to the complexity of recorded motion is the equipment utilized. At the NYU motion capture lab, recording muscle deformations with different intensities of movement was discussed as part of a research project

for creating realistic animations. Unfortunately the experiment was quickly deemed unfeasible with the several millimeter error in the Vicon System. The Vicon system, which is optically based, requires several cameras to record the 2D positions of reflective markers and uses triangulization to find the 3D position of a given marker. Other systems of different design may have greater accuracy and be more prevalent in the future.

Other common problems with acquiring decent data sets with this style of motion capture include occlusion of markers and shifting of markers over time. The problems are often alleviated with post-processing techniques of the data. In fact, research in data-based character animation in the past years has often addressed the problems of "bad data".

This dissertation aims to provide efficient and usable tools for animators, while exploring means of improving the expressive qualties of motion. The methods employed in the thesis include using notation from dance to enhance personality, the use of deformable skeleton to exaggeration nature dynamic changes in the body, and finally the use of sublties or key parts of the body, such as hands, to convey meaning.

The thesis is constructed as follows:

- Chapters 1 and 2 provide a history of motion capture.
- Chapter 3 lists an extensive survey of current research in the motion capture community.
- Chapter 4 details the authors ideas on how to improve the expressivity of motion-based animations and the realtionship to current work.

- Chapter 5 and chapter 6 deal with represntation and preprocessing techniques. Chapter 5 summarizes the representation of motion capture data used in the paper's projects as well as the data acquistion process. Chapter 6 outlines the representation of rotations in quaternions and the special techniques that are needed when working outside of a vector space during learning.

- Chapter 7 describes the projects associated with Laban Personality.

- Chapter 8 outlines learning a deformable skeletal model to simulate squash and strech animation with motion capture.

- Chapter 9 lists additional work with using hands and other subtle aspects of motion for visual emotional impact.

CHAPTER 1
FORMS OF COMPUTER ANIMATION

The traditional and often preferred technique among animators for creating life-like animations is key framing. This method entails creating a set of key configurations of a character. A computer fills gaps between the time steps of the key configurations by using splines of some other form of interpolation. The benefit of such a method is that the animator retains complete control over the actions of the character. An infinite amount of detail can be added. However, the first of numerous difficulties with the method is purely the amount of time and labor involved in creating a single animation. Additional problems include the fact that interpolation between key frames does not replicate how an actual person moves and the resulting animated motion will lack detail if too few key frames are set. An important aspect of this technique is that each frame is set individually and therefore independently.

1.1 Rule-Based Systems

In 1995, Perlin introduced a classic example of rule based systems for animating figures in a paper entitle "Real Time Responsive Animation with Personality." As the name suggests, the paper's animations are guided by a series of rules that defined rhythmic and stochastic noise functions which vary parameters driving the motions [57]. The actual motions are artificially generated by predefined functions. Perlin's preliminary contribution is the concept of a "textural style" of a motion. By varying key visual parameters,

his animated characters appear to have subtle human characteristics such as nervousness over a baseline motion such as walking. In proceeding years, researchers continued to decompose motions into a base action and style. The concept of style is of particular importance since it is the basis of the research detailed in this paper.

The following year, Perlin and Athomas Goldberg generalized the initial concept to create an entire scripted system for procedurally animating characters [58]. The system, called Improv, allowed animators to write simple 'scripts' to form animated characters with layered actions, gestures and personalities. The various commands written by the animator were interactively blended to create seamless gestures and motions.

The surprising aspect of Perlin's early work is that the subtle aspects of the motions conveyed are realistic even though the actions only vary along a few degrees of freedom driven by pseudo-random noise function. Dynamics are not considered in this work and therefore the correlation to actual human motion is most likely minimal. Despite the expressive qualities of the characters, this approach to animation was not adopted by a mainstream following. The lack of control has often been cited as a criticism by animators.

Around the same period, Casselle et al. used rule-based animating techniques to create a system that automatically generated conversations with corresponding intonation, facial expression, and hand gestures between characters [18]. The authors noted that the expressive attributes of facial expression and gesture are psychologically based. Animating this features requires a unconscious intuition of the look of the character. The author's contribution was the automatic generation of such motions by collecting a detailed list of rules based on human observation in advance. A difficulty in this approach

is that the system is based on human observations of 'gestures'. Critics have claimed that this method for obtaining data fails to capture the parameters of a movement that make one gesture transform into another. In addition, the perception of what is and is not a gesture is debatable.

A more recent procedural was developed by Chi et al. to convey the more expressive qualities of motion based on the movement observation science of Laban Movement Analysis [21]. While the authors of the paper never explicitly distinguish between the action and style of a movement, they allude to the notion of a class of movements that fall between the voluntary task-driven actions of, for example, walking and the involuntary movements such as breathing. These secondary movements are often involuntary and occur simultaneously with perhaps more conscious movements.

The system developed by Chi, called the EMOTE model for 'Effort and Shape', relied heavily on the notion that the concepts of effort and shape in Laban notation are essential for the creation of natural gestures. The authors were inspired by the observations of Badler in 1989, that the use of effort parameters would allow for more expressive movement control of figures [4]. It is important to note that the field of Laban Movement Analysis consists of five components: body, space, shape, effort and relationship that together form a complete language for describing movement. The paper authors focused mainly on the components of shape, which describes the changing forms that the body makes in space, and effort, which describes the exertion put forth while the body moves.

The shape and effort components each have several factors. These factors have been hypothesized to be the smallest units into which a motion can be decomposed. The authors built a system in which an animator could modify

these elements on a sliding scale over time. In order to give greater control the authors divided the body into specific sections onto which the Laban factors could be applied. Set rules, based often on joint angles or velocity, were used to animate a figure based on the user input of LABAN parameters.

The results were less than impressive. The authors' animated character appeared robotic and the simulated LABAN factors were difficult to distinguish or even determine, i.e. certified Laban experts were asked to view random samples and notated the actions with Laban factors. The experts had low rates of correct responses. One explanation for the paper's disappointing results is simply the use of a procedural technique. It seems foolish to believe that the nuances of gesture could be captured with a few observational rules. However, this paper is quite important for the contribution of the correspondence between the field of Laban Movement Analysis and computer animation. The theory that effort and space parameters explain the detail oriented 'secondary motion' of human actions is an acute observation that may help to aid in the creation of more expressive animated characters.

Despite the authors' choice of dividing their animated character into distinct sections of arms, legs, torso, etc., the paper makes a very interesting point in the conclusion section that is contrary to this notion. The authors cite,

> 'even if a character moves its arms with appropriate gesture, it will lack conviction and naturalness if the rest of the body is not appropriately engaged.'

This comment is key to the research that is described later in this paper. As with the method given in Chi's paper and in numerous others, many authors

are quick to dissect motion by body part rather than treat motion and the body as one continuous entity.

Rule-based systems never became a mainstream technique for animating computer characters. The lack of user control over the character has often been a major criticism. However, an even more fundamental problem is the questionable motion produced by some of the systems. It is interesting to note that all of the examples of procedural animation listed in this paper focused on animating the more "expressive" qualities of motion (this was not an intentional decision by this paper's author). Perhaps the fact that there are no qualitative measures of such ideas as gesture, a rule-based system is the first natural inclination to tackling such problems. However, the complexity of some aspects of human movement suggest that rule-based techniques are not the method to proceed with in the future endeavors in expressive motion.

1.2 Dynamics

A logical alternative to procedural modeling of motion, is to use physics based algorithms to animate behavior. In 1994, Tu and Terzopoulos demonstrated a realistic and highly celebrated physics-based system with the paper "Artificial Fishes : Physics, Locomotion, Perception, Behavior." [73]. The authors created an aquatic world of fishes whose dynamic motion and behavior were modeled based on physical equations. The bodies of the animated fish were decomposed into structures of masses and springs. The movements of the fish were simulated by hydrodynamics where the animated fish moved by "contracting" its muscles in the same manner as a real fish. The results were impressive. As the authors mentioned, simulating fish posed many control challenges. However, a human skeleton has considerably more parameters

(muscles) to consider and as a result has an added level of difficulty when trying to model with dynamics.

Hodgins and collaborators were in the spotlight around 1995 for creating an animated character that simulated a series of athletic events using dynamic simulation [36]. The simulations contained a rigid body model of a human and a control algorithm for various athletic events such as running, balancing, and bicycling. The position/angles of the joints were determined by the forces exerted upon them (determined by the control algorithm for the particular event). The animations were impressive in that they all performed the given motions accurately. However once again an expressive character remained elusive: the animated characters were notably robotic in nature. The paper's authors handled the human body as a rigid skeleton and voiced no intention of modeling muscle dynamics or simple visual deformations. Therefore, the authors aim of strictly simulating the position of limbs was reflected in the results.

A more recent contribution to physics-based character animation was made by Faloutsos in 2001. At the time, he lamented the unexpected dearth of animated characters that possessed a large repertoire of motor skills. In the paper "Composable Controllers for Physics-Based Character Animation," Faloutsos proposed a framework of autonomous synthesized characters that were subject to such constraints as gravity and contact forces [28]. The crucial contribution of the paper was a model, based on manual data and automatic data supplied by an SVM, defining when motor controllers are expected to function. The vast array of possible motions in humans required a composite controller with broad functionality to micromanage. Since the focus of the paper was primarily on controlling the entire body as an integrated unit with

automatic controls, only a bare skeleton with limited degrees of freedom was animated. The dynamic properties of the skeleton such as mass and moment of inertia were modeled from biomechanics literature. However the model did not posses a built in muscle model and allowed users to implement the model they preferred. The authors decided to model the actions of muscles with rotational spring-and-damper forces and maintained limits on the joints with exponential springs. Individual controllers that managed tasks such as sitting or standing, were treated as black boxes. The final product was integrated system for automatically controlling human movement through machine learning techniques.

An investigation into the employment of dynamical models to compute muscle excitation patterns of the human body for the purposes of animation appeared limited in the computer graphics community. On the other hand, in other motion related fields such simulations are the norm. For example, there is extensive literature written on dynamic simulation of muscle control in various part of the body by biomechanics researchers [71]. The SIMM software developed by MusculoGraphics is of particular interest as a tool for simulating muscle deformations. The SIMM software allows users to construct skeletons, model and analyze the muscle deformations of a moving skeleton. The software is used by both biomechanics researchers and animators. A brief description on the MotionAnalysis (a for profit motion capture company) website suggests that the software was used in conjunction with motion capture to fully realize a human skeleton with proper relationships between muscles. The relationship between motion capture points and the actual skeleton may be a crucial aspect in creating realistic animated motion in characters. Of course, the system is limited by the construction of muscles and bones created by the user and the

animations produced by the system still lack the expressive qualities of motion. None the less, it is important to note that the simulation of muscles through dynamics is a necessary step in creating life-like characters.

A final current example of dynamic simulation used in animation pursuits is especially relevant to this paper because the researcher's aim is to animate a more subtle aspect of motion that adds to the expressive/lifelike quality of a character. Zordan et al., presented an anatomically inspired, physically based model "breathing" throughout the human torso [78]. The authors accurately critique the lack of inclusion of breathing in the graphics community. They suggest that this involuntary movement is important in creating a perceptual visual affect of live motion. Their system achieves realistic results and offers a range of breathing behaviors, (casual, deep, panting) that can be modified for different characters. According to the authors, the benefits of their approach far exceed a data driven approach, which is only capable of recording a single type of breathing for a specific character (no flexibility) and a procedural system which tend to produce stylistic results.

Dynamic simulation has the potential of being an important tool in the field on animation. Due to the shear complexity of the human body, animated characters based purely on a modeled muscle/skeleton system often appear unexpressive and flat. However, as Zordan and collaborators mentioned, the use of dynamics can offer flexibility and accuracy that other competing methods can not provide. Perhaps the future of expressive computer generated animation lies in the use of dynamics in combination with data-driven techniques.

1.3 Motion Capture

The introduction of motion capture systems provided a reliable and efficient means of obtaining highly detailed motion data. Exponentially faster than traditional key frame animation and easily capturing complex and subtle aspects of motion that are lacking in straight dynamics techniques, motion capture seems to provide the best alternative in all but the most expensive projects.

As mentioned earlier, initially the optical motion capture systems were severally restricted by inadequacies in the software and design. However as motion capture systems becomes increasingly accurate the limitations of motion capture are now fundamentally rooted in the application of the data to animation.

Obtaining motion capture data is now a trivial task; crafting realistic and expressive animations from this data is a more difficult task. Once the data is acquired manipulating the data (or motion editing) and the representation of the data become primary issues. Traditional motion capture data is represented in hierarchical format. For human bodies, the skeleton is usually rooted at the pelvis and has a tree like structure of points extended through the core of the body and down the joints of the limbs. A proper representation of the movement of the motion capture data is another obstacle. Before proceeding it may be beneficial to review the standard motion capture pipeline for animating a character from start to completion. In optical based motion capture systems, a subject (usually human), wears a bodysuit with reflective markers attached to key positions in the body. These positions include the major limb joints (elbow, wrist, knee, ankle) as well as positions in the pelvis,

torso and head. The subject is recorded by the motion capture system and the markers are tracked and labeled. In most cases, a human model is constructed in which set of several markers correspond to key angles in the body. A rigid body kinematic chain is constructed and optimized to fit the marker data. In the process, the chain solves for the rotations of the angle positions to obtain marker positions. The data from the kinematic chain is exported to an animation program where a computer generated character is linked to the motion points.

The rigid body kinematic chain is an extremely advantageous tool. Motion capture data is plagued with occlusions, lost markers, uneven tracking and shaky points. As a result, raw marker data is often uneven and sparse at best. There are many ways of cleaning up the raw data by smoothing rough sections and employing splines to clean gaps. These techniques are labor extensive and are not always adequate to provide viable results. Another common phenomena in motion capture is for markers to slide over the skin in the suit or appear to change 'length' purely because of limitations of the material. The fitting of a rigid body kinematic chain to a set skeleton ensures smooth transitions along the joint angles as well fixed bone lengths. The principal reason for the use of a kinematic chain is that markers are often intentionally placed on uneven intervals on the body. The right shoulder marker may not be directly on the shoulder bone and it might be an inch higher than the left shoulder marker. The markers have little corresponding meaning to the actual skeletal positions and/or rotations.

Obviously, a kinematic chain to a skeleton enables rapid translation of motion capture data to an animated figure. However, in the study of motion

a rigid body model may not always be appropriate. The appearance of deformation in bone length may reveal changes in muscle patterns. The ignoring of this information may be at the expense of the details and expressiveness revealed in the original motion capture data

It is now the norm for the described motion capture pipeline to be accomplished in real time. The speed of the endeavor is undisputable. However, the usefulness of raw motion capture data is often minimal. In order for motion capture data be useful to animators, tools must exist to edit the motion. Motivated by this need, researchers have investigated a vast array of areas in their pursuit. The various types of motion editing will be divided as follows : initial motion editing techniques, constraint based methods, large database manipulation methods, statistical synthesizing and style/expressive based techniques. These tools will be discussed in greater detail in Chapter 3.

CHAPTER 2
HISTORY OF MOTION CAPTURE RESEARCH

Despite increasing popularity, motion capture is relatively new. Motion capture was not used for computer character animation until the late 1970's. However, the concept of recording human motion dates back to the late 1800's when Eadweard Muybridge studied the dynamics of horse and human movement with a sequence of time released cameras. Muybridge constructed a shed with 24 sequential cameras that were triggered by threads on the ground. As an animal passed through the shed shutters were triggered and the motion was recorded, Figure 2.1. A French physiologist named Etienne-Jules Marey, inspired by the work of Muybridge, was frustrated that the glass plates used at the time for photographs could not be changed quickly enough to record fast movements. Marey's first attempted to alleviate the problem by recording multiple images on the same photographic plate. With this method all movement could be analyzed on the same print, but it only was successful in recording large movements that extended along the length of the plate. In order to study more subtle movements, Marey invented the 'photographic gun'. The device held a glass plate that rotated once a second. When the trigger was pulled, the shutter opened 12 times in a second yielding 12 different images around the edge of the plate.

In 1915, the idea of directly copying human motion for animated characters was developed by Max Fleisher. Fleicher developed a method called rotoscoping which involved tracing around film footage of live actors playing

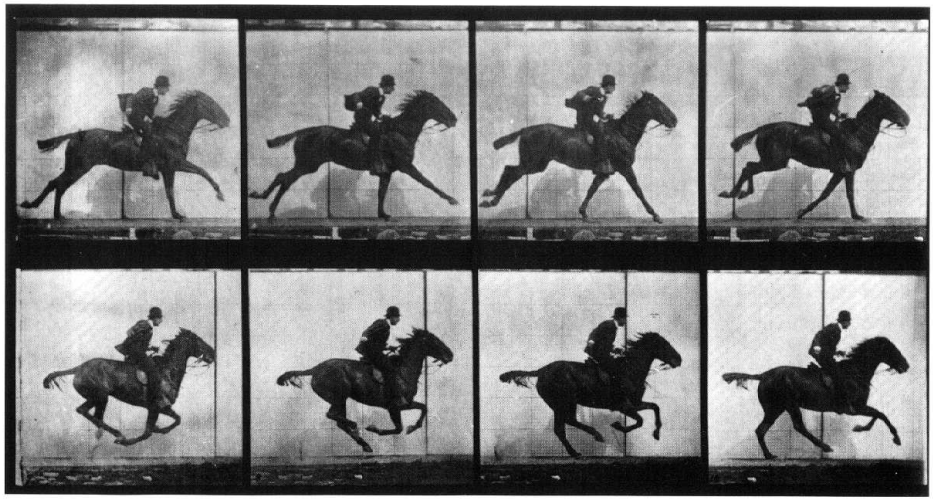

Figure 2.1: Muybridge Recordings

out scenes to obtain convincing motion for animated characters. This labor intensive process was used for the characters in Snow White and is still used to some extent today. It should be noted that this technique is often dismissed by traditional animators. The animated results often appear stiff and robotic - lacking the "essence" of the original movement. Similarly, motion capture has often been rejected by computer animators who claim that the results are unrealistic.

In the early 1980's, Tom Calvert formed computer animated figures driven by potentiometers that were attached to an actual body. Calvert's research was conducted for choreographic studies and assessment of movement abnormalities. The device used to extract information from the potentiometers was an electromechanical suit. The suit consisted of an exoskeleton that was strapped to each relevant section of the body. Potentiometers were positioned on the suit along side each joint so that they were triggered when the corresponding joints were flexed. Analog output was converted to digital

output which was subsequently fed into a computer animation system. It is important to note that the computer animation system used Laban notation (which will be discussed in detail later) and kinematics specifications to drive the actual character motion. Besides the obvious problem of restricted motion in such an apparatus, the system was unable to detect the subtle attributes of motion. Therefore data from the system was coarse.

Soon after Calvert began his project, MIT and the New York Institute of Technology Computer Graphics Lab commenced optical tracking of the body. Optical tracking systems often are driven either by flashing LEDS or small reflective dots used in conjunction with a series of cameras focused on a performance space, Figure 2.2. As mentioned before, the quality of the data from such system is limited to the speed at which the markers can be captured, occlusion of markers by other parts of the body (or other objects) and the resolution of the cameras. Initial optical systems were only capable of tracking a dozen markers at a time. However, more recent optical systems are capable of tracking dozens of markers at a time and have resolved many of the other optical system problems by including high resolution cameras and fast frame rates. For example the cameras in the VICON motion capture lab are capable of capturing 1000 frames a second. Occlusion can be alleviated to some extent by increasing the number of cameras in a performance space, but still can be a major set back.

Optical tracking systems have become the norm for most readily available or 'off-the-shelf' motion capture systems. The intrinsic limitations of design, especially the problems of occlusion and accurate marker position have led to numerous post-processing techniques and research in to varying procedures.

Since the early 1980's, several different systems of obtaining motion data

have been created. Various systems have included an optical fiber system which used fiber-optic sensors based on transmitted light. Unfortunately, the optical fiber system yielded unreliable data that was difficult to work with. Another system placed electromagnetic sensors on joints of moving objects. The orientation and position of the sensors was measured with respect to an electromagnetic field generated by a transmitter. Acoustic systems, where acoustic sensors transmitted sound waves suffered from low accuracy and other sounds in the same frequency range disrupting capture. Finally an optical strobing LED system partly solved the occlusion problem found in optical tracking, but could only handle a limit number of sensors.

Optical motion capture systems have become the standard and since the mid 90's have been developed commercially. However, the general availability of motion capture systems for researchers has been a recent phenomenon. As a result, the past twenty plus years has resulted in numerous research advances in strictly video-based motion analysis. The investigation of motion from video is plagued with a variety of vastly different problems than those of optical motion capture. Changing environments in video often cause difficulty in segmenting objects and lead to severe occlusion problems. For example, the inability to label markers that are close together, markers that touch and occluded markers. Perhaps not coincidentally, a primary area of research in video motion analysis is tracking. Tracking tools range from the standard Kalman Filter and early Lucas Kanade algorithm to recent developments such as the particle based system of 'Condensation'. Unlike the Kalman Filter, condensation supports multiple hypotheseses in tracking to be considered simultaneously. As a result condensation related techniques can handle occlusions, multiple markers and misplaced trajectories in tracking and produce accurate results.

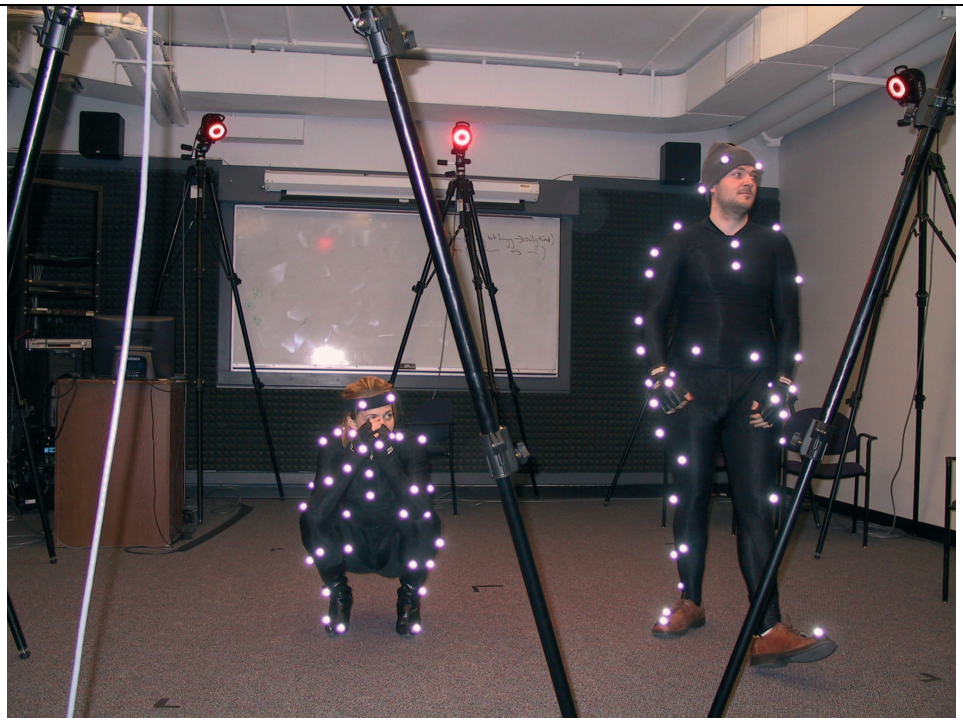

Figure 2.2: Optical Motion Capture System

The recent push of real time motion capture to character animation is partly a result of these improvements. Other relevant areas of video research include 3-D reconstruction of objects form video (or video-based motion capture) and object recognition.

Improved technology in optical motion capture systems and research into post-processing techniques (more to be discussed in detail later) are changing the face of motion capture. Like rotoscoping, motion capture has often been attributed with creating robotic animation. Technology once limited motion capture to a niche tool for projects that did not require the subtle and expressive aspects of motion. For example, motion capture is the perfect tool for sports based video games where large amounts of motion animation represented in a low dimensional space are required. In the past, the use of motion

capture in realistic animation required meticulously processing and repositioning by animators. Recent research suggests broader applications for motion capture and a place as a basic tool for animators. Current research trends in data-driven animation span many different areas, but the research discussed here is restricted to two main often overlapping themes. The concepts include creating more expressive animations using the idea of a motion 'style' and using existing motion capture data to recreate completely different movements that were never originally performed by an actor.

CHAPTER 3
SURVEY OF MOTION CAPTURE RESEARCH

3.1 Initial Editing Techniques

In 1995, motion capture systems arrived for commercial consumption. With the advent of the new technology, Andrew Witkin and Zoran Popovic immediately recognized the vital need for motion editing techniques and predicted a future of highly realistic animation generated from libraries of motion capture data [77]. They introduced a methodology for editing motion capture data, entitled 'Motion Warping,' which respected keyframe-like constraints, but still retained the fine details of motion. The authors initially stated a hypothesis, which has been incorporated into much of the proceeding motion capture related research, that the distinguishing factor between motion capture and traditional key frame animation is the expressive quality (or 'aliveness') of the data which stems from the preservation of high frequency details.

The authors use 'warping' to derive a fresh set of motion capture curves which are set on a set of constraints provided by the animator. A motion curve is defined as a value of one of the body's parameters (usually represented as a position and a rotation hierarchy) as a function of the time. Each curve is warped separately and is modified by constraints for time and position. The

method is as follows :

(θ_i, t_i) : constraint of value that must be assumed at time 't'

$(t_j^{'}, t_j)$: time warp constraint

$\theta^{'}(t^{'})$: warped motion curve defined by $\theta^{'}(t) = f(\theta, t)$ and $t = g(t^{'})$

An interpolating spline is used for the function $g(t)$. The values are warped by using a transformation

$$\theta^{'}(t) = a(t)\theta(t) + b(t) \text{ which satisfies } \theta_i^{'} = a(t_i)\theta(t_i) + b(t_i)\forall i \tag{3.1}$$

and where $a(t)$ is a scaling function and $b(t)$ is an offset function.

Blending of motion capture clips was accomplished by overlapping clips over an interval and progressively performing a linear interpolation over the time frame. The blending of two curves is defined as :

$$\theta_{blend}(t) = w(t)\theta_1(t) + (1 - w(t))\theta_2(t) \tag{3.2}$$

During the same year, Armin Bruderlin and Lance Williams compiled a list of techniques already used in the image and signal processing domain that were applicable to designing, modifying and adapting motion capture data [15]. The authors suggested that of the methods reviewed of multiresolution filtering, multitarget interpolation, dynamic time warping, waveshaping, and motion displacement mapping were appropriate for manipulating joint angles or joint coordinates in motion data.

The first technique, multiresolution filtering, had been well established in the image processing community for a number of years as a method of image encoding based on sampling an image with Laplacian operators of different

scales [16]. The general premise of the technique is that a range of filters can be recursively applying a lowpass filter to the dataset and subsampling by a factor of 2 at each iteration. This successive process of convolving an image (or data set with a filter) and then subtracting the results of previous iterations, produces a Laplacian pyramid of output which can be reconstructed to form the original signal. With this representation, high frequencies are separated from low level frequencies and can be utilized for such applications as merging two different signals by blending bands of the same frequency [32].

When applying the Laplacian pyramid technique to motion data, Bruderlin defined the number of frequency bands to be computed when $n = m$, where

$$\begin{aligned} m &: \text{the number of frames} \\ n &: 2^n \leq m \leq 2^{n+1} \end{aligned}$$

Each successive band of the pyramid is calculated by performing a convolution with a filter where the data is reduced by a factor of 2 in each iteration. Bruderlin suggests instead expanding the length of the kernel by 2 at each iteration. The algorithm then becomes straight forward :

1. Calculate the lowpass frequency bands of all signals by performing a convolution between the signal and expanded kernels. Let G be the motion signal and w be the filter.

$$G_{k+1} = w_{k+1} \times G_k \tag{3.3}$$

2. Obtain the bandpass filter bands :

$$L_k = G_k - G_{k+1} \tag{3.4}$$

3. The calculated frequency bands, L_k can now be modified or blended.

4. The original or modified motion signal is reconstructed by recombining the frequency bands.

$$G_0 = G_n + \sum_{k=0}^{n-1} L_k \tag{3.5}$$

Bruderlin suggests that the procedure of multitarget motion interpolation can be accomplished by blending the frequency bands obtained by the proceeding algorithm and reconstructing the signal. An example is the blending together of a waving and swaying motion by modifying high level frequencies of both motions. However, Bruderlin points out that such an operation can only occur if the motions have corresponding points over time.

The issue of timing is solved by the next technique of dynamic timewarping, which has a long history in the field of speech recognition [25]. Dynamic timewarping reduce to simply finding the optimal sample correspondence between two signals and warping them to match. The solution is obtained by calculating a global cost function measuring the difference between the two signals and finding the minimum through dynamic programming. A motion is warped to the optimal solution by applying the optimal vertex correspondences through substitution, deletion and insertion. For example, if there are two motion signals A and B, B can map to multiple frames of A.

When combined with multiresolution targeting, dynamic programming can enable seamless blending of various parameters of different animations. These two techniques are key in several motion editing tools. Later in the

paper, the described formulation, Eq. 3.5, will be discussed as a means of applying different styles and/or emotion qualities to preexisting motion capture data [61].

Finally, Brudelin introduced the concepts of motion waveshaping and motion displacement mapping. Waveshaping is the idea of using cubic splines to change the shape of a motion over time. Motion displacement mapping enables a user to change the shape of a signal locally, while maintaining continuity and preserving the global shape of the signal. The process involves an animator changing the pose of an articulated figure at a select few key frames. A spline is fit through the displacements and then added to the original movement. The idea is the basically the same concept as presented by Witkin in the same year. An iterative application of the displacement procedure can yield fine-tuned results.

These initial contributions have been expanded upon and combined in varying forms over the last 19 years. Motion warping, time warping and the Laplacian frequency decomposition enabled motion blending, motion deformation and the ability to make smooth transitions between motion segments from different sequences. However, researchers quickly noticed that these techniques were not sufficient for reproducing physically accurate transitions in motion. As a result, a proliferation of advances in constraint-based motion editing techniques occurred in the following few years.

3.2 Constraint Based Editing

In 2001, Michael Gleicher, inspired by the sheer number of unrelated constraint based motion editing techniques wrote a survey of the field comparing and contrasting the effectiveness of the differing methods [33]. Gleicher

observed that constraint based methods generally focus on spatial constraints, which provide features of the pose at given instances. While they must be respected, temporal properties have been less formally examined. However, he reiterates the observation of Witkins that high frequency bands of data must be preserved. In addition, the constraint based motion editing operations primarily are applied to tasks where motion is adapted for new settings. Gleicher points out that the signal processing approaches recommended by Bruderlin often destroy the spatial constraints. The author emphasized the necessity of a constraint based method to reestablish the validity of the motion.

The general approach for handling spatial or geometric constraints is called 'Inverse Kinematics' in the motion capture community. Often, the process for calculating the configuration of a character's parameters is based on specifications such as the position of ends limbs. All constraint-based methods include an IK solver of some form. However, IK methods pose several new problems, since the relationships are often nonlinear and for some goals there are simply no solutions. Analytic closed form solutions exist for some IK problems in carefully specified cases. Tolani, Goswani and Badler presented a closed form IK solution suitable for real-time applications of a 7 degree of freedom limb [72]. In most cases, the lack of flexibility in both under-constrained cases and types of problems that can be handled make closed form solution inappropriate. Despite the large computational expense, most researchers utilize both constrained and unconstrained numerical solution methods.

Before outlining the various advances in spatial constraint oriented IK motion editing techniques, it is important to recognize Gleicher's 1998 paper "Retargetting Motion to New Characters." [31]. At the time that most researchers were focusing primarily on transforming the spatial qualities of

motion, Gleicher investigating the problem of retargeting motion capture data to a character of differing proportions than the original body. A naive approach to such a problem might include simply mapping the joint angles from one character to another. An example was illustrated of motion capture data of a walk cycle being mapped to a character of 60% of the size of the original person. The animated character appears to be sliding along the floor because the constraints that we expect in a walk cycle are not being maintained.

Gleicher argued that inverse kinematics (IK) does not provide a satisfactory solution for this particular problem. Due to the fact that the IK solver considers each frame independently, the new constraints will cause jerkiness and a lack of consistency (i.e. an addition of high frequency to the motion). The process he proposed involves defining the new constraints which may include such commands as not letting the elbows bend, or the character may not go below the level of the floor. The next step is to find an initial estimate of the solution. The estimate is computed by scaling the translational parameters of the motion and adding a translation to define the center of the scaling, which is found by calculating the displacements of the scaled motion, interpolating and smoothing. The author uses the motion frequency decomposition recommended by Bruederlin and obtains an appropriate spacing for control points for the motion by specifying the key spacing as the points where the highest frequency band exceeds a threshold. From this formulation a representation for the motion-displacement curve is formed. Finally a non-linear constraint problem is solved with a numerical solver that finds a displacement such that when it is added to the original estimate of the solution it will provide a motion that satisfies all the stated constraints [31].

As described in the background of animation, traditional computer animation systems provide a user with the ability to control a set of key poses, which rule the spatial aspects of motion. In motion editing operations involving keys, the solver examines each pose individually. A significant disadvantage of this method is that the key frames may not be placed at positions that are convenient for editing. However, motion warping combined with inverse kinematics is a solution to this problem. This solution is somewhat flawed because this is no control over geometric constraints except at key frames.

Most motion constraint-based motion editing techniques fall on the category defined by Gleicher as "Per-Frame Methods." The idea is similar to the idea of key-frame methods, except that the keys are required to be regularly spaced. Since the data in most cases is the output of motion capture machines, the per-frame methods are dealing with densely sampled data with minimal change in motion between frames. Therefore, the consistency between poses must be enforced.

In 1999, Jehee Lee and Sung Yong Shin presented the paper, "A Hierarchical Approach to Interactive Motion Editing for Human-like Figures," in which frames altered by an IK solver are modified to maintain smoothness [48]. In order to obtain a smooth displacement map, the authors suggested implementing a motion decomposition technique and finding appropriate B-splines to act as low-level filters to remove spikes and discontinuities. The formulation is not technically correct because the smoothing operation and IK operation are conducted separately. Since they are not conducted simultaneous, one operation may reverse the changes of the other.

Independently filtering each parameter of an Euler angle has little meaning in relation to the actual rotation itself. Using the domain of displacements

is often justified because the displacements are typically small and with small angles, the orientation representations are linear. The following year, 2000, Lee and Shin presented a report in which they showed that using a multi-resolution (laplacian decomposition of frequencies) in combination with an exponential representation of coordinates has a theoretical meaning [47]. The discovery of exponential maps provides a natural and non-singular parameterization for performing filtering with small angular displacements that can be mathematically supported. As mentioned previously, filtering Euler angles has no mathematical basis and furthermore, unit quaternion space is inherently non-linear and therefore difficult to perform simple operations such as interpolation in.

The previous constraint-based IK methods all considered frames individually. Spacetime IK methods differ in that they analyze all frames involved in the duration of a motion simultaneously. The "Retargeting Motion" paper by Gleicher described earlier considered this approach to motion transformation. Gleicher also proposed a 'Motion Editing Approach with Space Time Constraints' in 1997 [30]. The author turned the problem into a numerical constrained optimization problem with a set of constrains that maintain the existing motion and specify desired changes. The optimization problem is solved by quadratic programming. However, the solution had ignored basic dynamics and physics (including Newton's law so that an easy mistake is for the character to fly off into space) .

A year earlier, Rose et al. also proposed an IK technique for motion editing using spacetime constraints [64]. Rose's method was specifically for the generation of motion transitions. The solution was calculated using a fast recursive dynamics formulation. Utilizing dynamics as well as preserving

spacetime constraints produced realistic transitions. The authors gave an example of a straight joint angle interpolation which produced unnatural linear results. Their spacetime constraint optimization yielded natural limb angle deformations.

While Gleicher chose to ignore Newton's Laws in his spacetime solution, he noted that in some situations the inclusion of kinematics is crucial. Zoran Popovic and Andrew Witkin noted the absence of dynamics in constraint-based editing techniques (as well as in other motion capture techniques) and responded with a spacetime constraints approach that simplified the geometry of the skeleton in order to include Newton's laws [59]. Their approach can be broken into four stages. First the skeleton is projected onto a simplified model with a minimal number of degrees of freedom. A spacetime motion fitting is performed to find a solution that matches the simplified character. A spacetime edit is performed that introduces new pose constraints, changes the kinematics and objective function. Finally the motion is reconstructed by mapping the change in motion introduced in the spacetime edit onto the original motion. The approach is limited by the simplification of the figure. In highly dynamic and highly periodic motion sequences such as running, the methodology performs well. However, other more lethargic motions will not yield satisfactory results without a more complicated skeletal model - which requires an unacceptable amount of additional computation.

Obviously a large amount of initial research in the fields of motion capture was dedicated to finding optimal methods for IK constraint problems. All methods include varying tradeoffs. The choice of methods is based on the problem formulation.

3.3 Current Motion Capture Research

A vast majority of current research in motion capture can be categorized under the broad heading of motion synthesis. Using various motion editing techniques, all the approaches described in this section seek to aid the formulation of larger more flexible databases of motion from limited input data. Some techniques extend the previously described editing techniques such as methods that seek to broaden the expanse of motions that can be feasibly blended together with smooth transitions. The remainder of the procedures described will be characterized as motion synthesized models and methods of generating large databases from existing motion capture data - including automatic segmentation, and motion compression.

3.3.1 Editing

In 2003, Anthony Fang and Nancy Pollard proposed a solution to the problems of constraint-based motion editing techniques lack of inclusion of dynamics or conversely the sacrifice of model complexity for dynamic models [29]. They describe a set of objective functions that calculate the first derivatives of dynamics parameters such as joint torque in linear time instead of the standard polynomial time. The paper investigates a restricted class of functions where physics constraints are included. The formulation produces highly dynamic results that preserve "squash and stretch" characteristics of physically realistic motion, though it is limited to specific cases.

The paper relates to research by Liu and Popovic in 2002, in which the authors suggest that desirable animation effects can be extracted from utilizing dynamic patterns [50]. The paper relies on momentum patterns to synthesize

dynamic character motion from simple animations. Fang and Pollard are excited by utilizing momentum patterns, but criticize the competing research for lack of inclusion of interaction constraints. They suggest that the future of highly dynamic and accurate motion lies in a combination of correct physics and knowledge of the natural dynamic patterns of motion such as movement or roll of the foot, etc.

In 2004, the tradeoff between simplification of the model and sacrifice of dynamics in optimization methods was addressed with a technique for simplifying complex characters in motion specific spaces [66]. Based on the intuition of motion dynamic patterns, Safonoa presented a method for accurately synthesizing realistic motion from low-dimensional subspaces. The authors showed that highly periodic motions can be drastically reduced by the simple method of Principle Component Analysis (PCA). Optimization is then used to define linear coefficients that relate vectors to desired motion. Finally constraints are maintained through inverse kinematics solvers. The key intuition behind the paper is that the optimization of human motion is more effectively calculated in a lower dimensional space. However, this technique is limited by the need for a motion basis of similar related motions.

Kovar and Gleicher exalted the significance of traditional blending techniques for creating new motions from limited motion capture data in the paper "Flexible Automatic Motion Blending with Registration Curves," [38]. With traditional blending techniques, if there is no additionl information about the motion data other than the raw parameters of root position and joint angles, then only the blending of extremely similar motions will produce visually satisfying results. The authors contributed a method of expanded the range of motions that can be blended by the use of registration curves. Construction of

a registration curve is a simple application of time warping to each curve, and calculation of a distance function to find a 2D rigid transformation that effectively changes a motion's local coordinate frame to match the other. While the algorithm is not appropriate for the blending of all motions it does broaden the possibilities. The authors note that the method does not enforce physical constraints such as balance, but propose that a post-process could be applied to constrain the motion.

In an entirely new vein of motion editing, Michael Gleicher mused that the concept of 'path' is an abstract of motion in 2001 [33]. He noticed that most motion editing tools being produced offered numerous ways to adapt motion to set constraints, but did not easily adapt to different paths over space. Gleicher's creation of an interactive tool to modify the path of locomotion was not novel, but the idea of adapting to motion capture data and preserving the expressive qualities of personality preserved in the data was a great advance. The author points out that an alternative method to producing large ranges of motions is to capture a large database of motion and blend motions together for desired needs. However, this approach, to be discussed further later, is limited in that motion capture libraries can not possibly predict every motion. The method utilized is a trivial extension to displacement mapping. Given a path and an initial motion, the motion can be configured into the path by placing the motion in a moving coordinate system along the path. An initial path is defined for the characters motion along with a time varying rotation matrix for the path. A new path is chosen by the user through curve editing tools that are defined by the animator. Apply the following equation to the data:

$$P(t)R(t)R_0(t)^{-1}P_0(t)^{-1} \tag{3.6}$$

where

$P(t)$: the path

$P_0(t)$: original path

$R(t)$: rotation matrix

$R_0(t)$: original rotation matrix

Finally apply a constraint solution technique to preserve geometric features of the motion. The method is equivalent to using a motion displacement map on the translation of the map with the inclusion of a rotation frame.

3.3.2 Motion Databases

From traditional animation techniques to automatic generation of motions in video games, large motion databases used to synthesis complicated motions have long been the projected future of motion capture. Kovar, Gleicher and Pighin added a significant contribution to this vision with the introduction of 'Motion Graphs' [40]. A motion graph a method for navigating a motion database, that consist of pieces of original motion and automatically generated transitions. The nodes of the graph are choice 'points' in which the small segments of motion are joined together. Since the transitions are automatic the user does not need to search for motions that specifically connect with each other. A key aspect of the motion graph system is user control for construction of new motions. The automatic transitions in the graphs are generated by calculating distance functions between potential frames of fixed window lengths. Local minimums in the calculated distance functions

are used as transitions points. The transitions are then generated by spherical interpolations between joint angles (represented presumably in quaternions).

Along similar lines, Arikan, Forsyth and O'Brien presented a parallel framework for synthesizing motion from large databases using user selected annotation such as "walk, run or jump" [2]. According to the authors, motion graphs do not allow for global planning of motion. The lack of global planning may cause disturbances if for example a jump is needed at some point in the animation and requires a long preparation. The system involves the defining of a vocabulary to match segments of motion. The user is allowed to manipulate geometric constraints directly, place frame constraints on the time line and choose motions. Dynamic programming is used to select optimal sequence of motion from a database. The search is recursively applied to obtain finer levels of detail. The system is limited by it's inability to synthesize frames that are not included in the database.

Motion graphs and annotation based motion systems offer flexible systems for user navigation of motion data. Such systems appear to be the most likely direction for motion capture synthesis. However, both systems were focused on user defined motion editing. It is important to recognize that realtime navigation of motion databases for video game applications is another area of related research. Lee and collaborators presented a system of efficient searching and identification of plausible motion transitions through clustering in the paper, "Interactive Control of Avatars Animated with Human Motion Data." [46]. The authors take a dramatically different approach then previously described systems and model motion data as a first order Markov process. The process is represented as a matrix of probabilities of transitioning from frame i to frame j. Transitions are formed by blending operations and constraints

are maintained with the heirachrchical motion fitting algorithm previously described [48]. This lower layer Markov process captures motion details. The authors define a higher layer statistical process the captures the distribution of frames and transitions. 'Gaussian Mixture Models' are used to cluster motions with similar states. The effectiveness of the clustering was directly influenced by the distribution of the motion capture data collected. As motion capture bases increase in size, statically based techniques become integrated further to control/navigate motion capture synthesis.

In parallel to research to navigating and synthesizing motions from motion database, are methods to segment and parameterize data. The large scale collection of large motion capture sequences demands techniques to automatically divide input into distinct motion. In 2004, Barbic and collaborators presented several straightforward approaches to easily segment motion data into distinct behaviors [6]. The first method uses Principle Component Analysis. In this technique, a motion is divided into a segments when the dimensionality of a local motion increases suddenly. It is based on the intuition that the derivative of a motion rises sharply above a constant value when a transition occurs. The second method, which was reported to yield superior performance to the other two suggested algorithms, creates segments using a probabilistic model of motion obtained from Probabilistic PCA. The final method uses Gaussian mixture models to generate segments. All three techniques are valid alternatives for automatic motion segmentation.

Kovar and Gleicher observed that large motion databases invariably contain several versions of the same type of motion. However exploiting this fact is difficult without appropriate tools, which motivated the collaborators to develop automated methods for extracting and parameterizing motion based on

characterizing motions by a novel similarity measure [39]. Kovar's similarity measure is based on the intuitions that :

1. logically similar motions can have differing skeletal poses
2. numerical similarity between motions can be defined by distance functions of corresponding frames
3. a match web can be pre-computed for a compact efficient representation.

Therefore, related motions can be automatically extracted, registered and blending to create a continuous space of motion.

3.3.3 Statistical Methods

An alternative to generating complicated motions from large databases and/or solving optimization constraints, is synthesizing completely new motion that is statistically similar to an original motion capture data sequence. Pullen and Bregler proposed generating realistic motion data that resembles motion capture data but contains random variations that maintain the statistical properties of the original motion [60]. The methodology involves initially decomposing the motion into frequency bands. The authors recognize that it is inappropriate to treat the joint angle motion and frequency band separately because the information is dependent. The dependencies are modeled with multivariate densities using kernel-based representations. An optimization procedure was used to randomly sample from the distribution and synthesize similar. The paper only applied the technique to 2D data and acknowledged that an application to 3D motion capture data is significantly more difficult.

Li et al. presented the concept of Motion Texture, a characterization of the stochastic and dynamic nature of motion, in 2002 [49]. A motion texture

is defined to be a two-level statistical model that utilizes a Linear Dynamic System to capture local linear dynamics and a transition matrix to retain the global non-linear dynamics of the system. While the paper was able to synthesize motion from a variety of examples, the technique is limited to motions that demonstrate periodic or repeated behavior. In addition the synthesized motion may lack global variations when the training data is limited. The authors attempted to incorporate an interface allowing users to edit a motion texture, but it is was determinate that the edited pose can not deviate significantly from the original motion. The many limitations of the system suggest that much development will be needed before statistically synthesized motion becomes a mainstream approach.

3.3.4 Motion Style

So far in this paper the descriptions of previous background work and current research involving motion capture data have conspicuously ignored techniques developed explicitly to enhance the expressive nature of motion and/or methods for modifying the 'style'. The topic is intimately connected to the research described in this paper. The separate concept of style in relation to a motion was distinguished by Perlin in his papers for generating procedural animations [57]. In a general sense the idea of style is the parameter that varies when different people perform the same physical motion in space. The Emote System for procedural animating the various parameters of LABAN notation offered another alternative for modifying the expressive aspects of motion [21]. While the results of the procedural system were disappointing, future techniques involving the principles of LABAN Motion Analysis may be integrated with the realistic data of motion capture systems for

created modifiable and expressive motion sequences. The natural inclination for formulating a system is using machine learning techniques to "capture" the LABAN parameters that can be reapplied to motion segments of differing styles. Background for the separation style parameters in a motion exists in past motion capture research as well as in related fields.

In 1995, Unuma, Anjyo and Takeuchi presented a method using Fourier expansions of human motion data to interpolate and extrapolate motions to create new emotionally based animations [74]. The ultimate goal of the early work was to create cartoon-like exaggerations or expressions of movement. The simplistic technique blended together motions of different emotional contents to create different emotional level content using a Fourier representation. The method did not differentiate the style and content of the data and is obviously limited by the need for motion capture data of the same motions with different emotion content. It also requires that the motion be periodic in nature. While this does not offer a generalized approach to expressive motion, it does allow interactive generation of motion with different moods.

Rose, Bodenheimer and Cohen from Microsoft Research suggested creating character that exhibit complex and subtle expressive behavior by utilizing a combination of radial basis functions and low order polynomials to create interpolations between motions [63]. The technique was developed with the goal of control of emotional expressiveness and behavior. The work does not attempt to "apply" characteristics of one motion to another, but instead assumes that a library of initial motions will have emotional information embedded. Unlike, the method proposed by Unuma, Rose's technique was not based on the frequency domain and therefore was capable of handling non-periodic motions. The system assumes that a large repertoire of sample motions exists

in a motion library. The motions should have a variety of parameterization such as walking, jogging, idling, etc. as well as different examples of emotions such as happy, sad, afraid, etc. With the system, different parameterization of motions were combined with varying degrees of emotional values. The result was a system capable of a substantial variety of behaviors and emotions. However, the system is substantially limited by the input to the system and the user defined emotions. The emotions are specified by user and may have little relation to the underlying 'style' . It is not clear that the system has a general method for modifying personality.

A short sketch at Siggraph in 2002, proposed exaggerating the expressive features of motion using a 'Singular Value Decomposition' of a matrix of motions performed at different intensities [23]. The sketch theorized that the "expressiveness" of a motion is determined by the magnitude and predictability of changes over joint angle trajectories. From a SVD decomposition,

$$A = U \sum V^T \tag{3.7}$$

of a motion matrix, A, of similar motions performed at varying intensities, U is extracted as a representative basis. The authors defined an expressiveness, E, for a joint-angle trajectory across the varying intensities in the equation

$$E = O \bullet P^2 \tag{3.8}$$

where P is the predictability calculated by an eigenvalue weighted sum and O is the observability measured by Euclidean distance. Selected motion trajectory projects are extracted and project onto U to create exaggerated movements. The authors offer a novel method of creating exaggerated emotion content in motion. However, the method offers no clear advantages over straight interpolation/exterpolation of motion segments. In addition, the method is application specific and offers no generalization for broad consumption.

All of the preceding techniques offer differing methods of modifying the expressive qualities, but none offer clear distinguishing factors between the content and style of the motion. The separation of style and content has been a prevalent theme in papers in a variety of vision related fields in the past 10 years. The decomposition of style and content has been applied to such problems as recognizing font, recognizing faces, and analyzing motion. In the context of motion, content is the actual base body placement and style is defined as the component that varies over individuals. These definitions are not fixed and have been interpreted in different ways and varying approaches.

Brand and Herzman approached the idea of style as a learned hidden Markov model [11]. The beneficial aspects of a statistical model included the synthesis of new motions in differing styles which can be adjusted with stylistic knob (parameters). The authors sought a model representation that captured the essential structure of an example motion capture sequence, while discarding the accidental properties such as noise or bias particular to the sample. An learning objective function was constructed such that the cross-entropy between the distribution of the model and statistics extracted from the data is minimized. The objective function also minimized the cross entropy between generic and specific models so that models behave similarly and their "hidden" states contain comparable "meanings". Finally the objective function minimizes the predictive nature of the model in order to construct a generic model that produced the most concise representation of the data. Optimization was conducted by expectation-maximization. The entire learning process was as follows :

1. A generic model and a style specific model were initialized for each motion sequence

2. Expectation maximization was iterated until convergence

   (a) E step : computed statistics of each motion sequence in relation to its model

   (b) Calculate parameter values related to the minimum entropy prior and the minimum -cross entropy priors

3. Use PCA to find a subspace that spans the parameter variations between the generic and style specific models

New styles can be created by interpolating and extrapolating between style specific spaces, which are specified as a vector concatenating the learned state means, square-root covariances and state dwell times. The dimensionality of the space is reduced by Principle Component Analysis. Synthetic motion data can be generated to apply a specific "style" on to existing motion by calculating the maximum-likelihood path of the new path in the desired style given the style parameters and the state sequence of the original motion content. The path produced is an inherently smooth curve.

Style machines are conceptually interesting because they offer a generalized notion of style. In addition, the method is extremely flexible in that motion capture data need not be segmented, annotated, aligned and includes no kinematic or dynamic model. All information is self contained within the data. Unfortunately, the results of style machines were not visually convincing and not adopted as a valid method of analyzing/creating style among motions. The unsatisfactory results produced by style machines and other statistically

based means on synthesizing motion data, prompted many researchers to focus primarily on techniques that kept the information within motion capture data intact.

In response to the statistically driven approach to style in which many of the subtle details of the motion capture data were sacrificed, Pullen presented a data based characterization of style in 2002 [61]. The premise of Pullen's method was the extraction and direct mapping of a style from one motion capture onto another of different content through matching similar frequency bands. Pullen suggests an application in which a style from motion captured data can be applied to key framed animation. The data from both sequences is first decomposed into frequency bands. A frequency band that provides information on the content of the entire motion for a particular angle is selected. The frequency band is segmented according to the locations in which the first derivatives of the angle changes sign. The bands for the real and key framed data are fragmented. Every motion fragment from key framed frequency bands is compared to the all the motion capture fragments and the 'K' closest matches are reserved. A optimal path through the K closest matches for each segment is used to reconstruct the entire motion. A blend function is used to seamlessly blend together the frequency band components. The method seemingly seems to directly apply motion onto new characters. Disadvantages of the data driven approach to apply style are the lack of hard constraints and that certain angles must be specified as the angles "match" by an outside user. As a result the procedure can not be fully automated and has limited applications.

In the paper, 'Separating Style and Content on a Nonlinear Manifold,' Elgammel and Lee suggest that the content of a motion is actually embedded

in nonlinear manifolds [27]. Furthermore, they define style as a linear combination of these manifolds. The paper offers a signification contribution in the observation that motion should be modeled in a nonlinear space, but fails as a valid method of generally separating style and content of motion. The length and involved approach suggested in the paper is extensive and does not offer novel results. Instead the paper suggests many manipulations of data to get results that can be obtained by simple linear interpolation with time warping.

Elgamal and Lee recall the use of bilinear models in the work of Tenenbaum and Freeman where multi-linear and linear analysis was successfully utilized in the decomposition of static images (i.e. face recognition) [69]. Linear dimensionality reduction methods have a long history of being used in simple image (face) recognition experiments [7]. Tenebaum extends this work by fitting models that discover explicit parameterization that characterize the dependencies between similar styles and similar contents. A bilinear model is a two factor model that is separable when one factor is held constant. The authors demonstrated extremely promising results in classifying, extrapolation and translating styles and content of various images. For example, they successfully learned particular fonts from a limited alphabet of letters and accurately predicted the shape of an unseen letter in the new font.

Lee and Engamuld formulated their technique on the observation that a simple bilinear model is not appropriate to decompose style and content in motion because with dynamic objects the visual space is nonlinear. The author's cite an example of a simple walk cycle. The authors point out that two points in the walk cycle which should be the farthest points in the kinematic cycle are very similar in the Euclidean visual input space. The advent of new nonlinear reduction methods are allowing scientists in a variety of field to find

meaningful low-dimensional structures embedded in complex high-dimensional manifolds [70]. Unlike tradition techniques such as PCA, the new approaches claim to computer globally optimal solutions by discovering nonlinear degrees of freedom in natural observations.

The inclusion of a brief overview of linear and nonlinear unsupervised learning methods serves as an introduction to a direction that may yield a generalized solution for separating the style and content of motion. Several interesting approaches have been examined over the years with the aim of creating more expressive and flexible animations from motion capture data. However, the effectiveness of the various techniques has not been promising. The procedural approaches to style, including Perlin's IMPROV system and the EMOTE model were limited by lack of user control and the ability to recreate the subtle detailed aspects of motion seen in motion capture data. The concept of a learned statistical model of style also failed in the creation of visually accurate human motion. Other methods such as Pullen's motion texture produce visually satisfying results, but are severely restricted the flexibility of use. A generalized solution to the separation of content and style in motion data has yet to be developed. Success may lie in nonlinear learning methods in the vein of Engamuld and Lee.

CHAPTER 4
EXPRESSIVE MOTION

4.1 Unresolved Problems in Motion Capture

Motion capture technology has been a relatively recent phenomenon. It was not until 1995 that commercially availably motion capture systems were produced. The initially proposed motion capture systems were plagued with difficulties, e.g. the realism of the motions proposed, that limited their use. On the other hand, with improved technology current systems produce extremely accurate results with incredibly high frame rates (a standard is 1000 frames per second). Access to highly detailed and accurate motion data from such systems prompted an entirely new field of data driven character animation.

Over the past nine years, numerous techniques have been developed to aid the creation of realistic animated motions through manipulations of motion data. The most likely direction for data driven computer animation lies, as originally predicted by Witkin in 1995, in large databases of motions. While the background sections of this paper have demonstrated the numerous developments to aid and ease the animation process through motion capture data, an integrated tool to assist the mainstream animator does not exist. The standard tools used by animators include Motion Builder, Maya, Poser or 3D Studio Max. The motion capture specific software of Motion Builder offers tools to aid in the construction of poses and blending of motions, but is far from a motion equivalent of Photoshop. The future of motion capture

based animation lies in the development of tools analogous to Photoshop that provide easy, clear tools for any user to modify and reconstruct motion.

A framework for the construction of an easy user interface for the construction of motion sequences was presented in the paper "Motion with Annotations." Similar ideas were expressed by papers on motion graphs, path editing, and the numerous solutions to IK constraints. Tools need to be developed for simple navigation through libraries of motion capture data to create the desired scripted animations. In addition, processes to modify the style of motions are useful. Animators should be able to easily change the personality expressed through the given motion. In order for these techniques to mimic key-framed animation, the ability to exaggerate actions is a necessity. Traditional animation has relied on physically infeasible exaggerations such as "squashing" and "stretching" figures to convey emotions. A comparable tool for applying these concepts to motion capture characters is needed in order to create dynamic motion instead of the flat movements often associated with motion capture.

Despite the numerous advances, motion capture has many obstacles to overcome; the most obvious of which is the unconvincing robotic animations that are often produced. After working in a motion capture lab for over a year, it is always surprising to witness the horrible animated results produced by motion data. The raw data points retrieved from the motion capture system itself are amazingly expressive and highly detailed. Often the essence of the motion, which is so clearly expressed in the data, is lost in the translation of the points to an animated character.

One could argue that the poor translation of motion capture data to

computer characters is simply due to lack of resources. For example, the Motion Analysis Company's website shows animations of highly realistic human characters performing motions derived from motion capture data. The description of human motions of football players created for a Nike commercial involved a motion capture lab with 50 cameras running and an extensive body marker set. The characters' skin deformations were based on 3D body scans performed on actual humans. The company often fits motion capture data to a full skeleton and references the use of the SIMM software to calculate accurate muscles movements based on skeletal patterns. Correct muscle deformations can be produced with the use of dynamics. In addition, large scale companies with lots of money have access to complex animated character skeletons which can include many blended shapes and parameters for muscle deformations and accurate modeling of the human body. The numerous procedures used to create the highly realistic animations are extremely expensive as well as time consuming.

In order for motion capture to be an accessible tool, methods need to be adapted to allow realistic movements to be easily mapped to characters for the average animator. Tackling this problem may involve dealing with several areas. One difficulty lies in the capabilities and degrees of freedom of the character to be animated. The complexity of animated characters is beyond the scope of this paper, but is always an important consideration when striving for visually accurate animations. Another solution is improvements in the model for the body skeleton and perhaps for the muscle deformations. The use of kinematic solvers to model the movements of the body could possibly be incorporated into motion tool kits to allow automatic generation of muscle deformations from motion capture data. A final possible explanation for the

robotic appearance of many data driven animations is that the motion editing techniques modify the motion to such an extent that they create unrealistic movements. This suggests further research into constraint based techniques along the lines of Fang and Pollard to create IK solvers that include the full complexity of the human body as well as complete dynamic constraints.

Of course, implementing an alternative viewpoint to creating realistic motion is not only to strive to develop more accurate dynamics of human motion, but also to focus on perceptually key aspects that make movement life-like. These aspects may include subtleties such as breathing, eye movements connected with motion and other involuntary aspects of motion.

As suggested previously, if the goal of animation, whether computer generated or otherwise, is to effectively convey emotional impact through character movements then tools are needed to effectively modify the expressive qualities of motion. In particular, the decomposition of a motion into style and content may aid in the grand scheme of adapting motion databases. The ability to transform the personality targeted to a specific motion will give animators greater flexibility in developing characters and story lines. While several techniques have been outlined for the translation and generation of particular styles, no general solution has been found.

In many situations motion capture data may be recorded for a specific application. Even though the acting may be tailored for the specific animation, the emotional impact of the movements may not be as expressive as the animator would desire. Tools in the Photoshop tradition, for "exaggerating and/or diminishing" the emotional qualities of movement may be a useful addition.

4.2 Objectives and Goals

This paper presents an integrated approach to improving the expressive qualities of character animation driven by motion capture data with an emphasis on efficient techniques that can easily be accessed. Ultimately the thesis strives to create visually convincing animated characters with minimal additional effort on the part of the user. The proposal involves three separate steps to achieve these goals.

The first step is motivated by observations by collaborators in the dance community that the skeletal representation often used in the motion capture pipeline is not sufficient to capture the essence of movement. The initial approach is to improve the skeletal interpretation of the body defined by motion capture. Specifically, the paper seeks to derive data about the motions of the spine, scapula, and clavicles from basic motion capture data. The construction of a more complete skeleton will lead to a better motion for the body. Abstractly, the concept is to constrain the body to maintain a smooth line of motion throughout the body at all times. If the body is aligned correctly and with a correct skeletal representation, the model will visually exhibit characteristics such as weight and presence (as well correct deformations of the body corresponding to movements) that are often absent in motion capture animation.

The second stage focuses on the inclusion of subtle aspects of motion which are often neglected but are visually important. In previous sections, this paper has suggested the addition of certain involuntary motions such as breathing and eye movement. However, this proposal seeks to expand the definition further. Many motion editing techniques assume that bone lengths

are fixed rigid bodies. This assumption may be inaccurate. In places such as the spine, the bone lengths do appear to "lengthen" and "shorten" as the spin contorts. In addition, changes in the length between model markers in motion capture sequences can correspond to muscle deformations. We seek methods that correctly extract the deformations in bone length and hope to use this to add to the information provided from the motion data. The ultimate goal of such a pursuit is a natural extraction of "squash and stretch" within a motion capture animation for higher emotional value.

Finally, this proposal seeks robust methods of applying different "personalities" or styles onto motion capture data. While working in the motion capture lab, a frequent problem was that an actor did not display in their motion the personality characteristics desired for a particular animation. These 'personality' characteristic are not mood, such as happy or sad, nor are they broad motion styles in the vein of the motion style papers of Pullen and Brand. Instead, this proposal suggest that motion personality is the general sequence of effort parameters, derived from LABAN movement analysis, that an individual's motion tends to obey. A method to broadly apply meaningful 'personality parameters' with tunable amounts of exaggeration will be presented.

4.3 Expressive Motion

Recently, a review in the New Yorker magazine of the major motion picture release 'Polar Express' reveals a fundamental problem with motion capture based animation. The experience of watching the movie was described in the article as paramount to viewing "mannequins" act. As argued several times in this paper, motion capture based animation without extensive user input is often characterized by viewers as robotic and inexpressive. In the

current state, primary capture is most useful in niche animation applications such as video games.

The main thesis of this paper argues that motion capture is not flawed. Instead, the current methods utilized need to be modified. In the past, motion capture was limited by many factors including lack of precision, occlusion etc. However, recent techniques have yielded motion capture systems that are increasingly precise and more advanced tracking techniques such as particle based algorithms like' Condensation' have rendered missing data points less of a problem. In addition, the large repertoire of motion editing techniques discussed in the survey portion offer numerous opportunities for generating and editing motions.

The limiting factor in motion capture is the demands of the user. A motion capture system can only return what the user is searching for. This is a simple point that is often overlooked. It should be of little surprise that a character that is based solely on the motion of arm and leg joints appears "robotic". The human body is not a stiff cylinder with limbs that move. Instead, we have a spine and core that expands and contracts, lengthens and shortens. In addition, the motion of the arms and legs, torso, hips and head are all related. For example, rarely does one move ones arm completely independently from the rest of the body. When the arm is raised, the spine often lengthens. Sometimes the weight of the body is inadvertently shifted to the leg on the opposing side of the body and the opposing shoulder falls. The clavicle of the arm being lifted completely shifts its position and changes the shape of the back. All these changes are visible and related. Simply registering that an arm moved is insufficient.

Viewing the body as separate rigid body elements is a fundamental problem that may in fact inhibit the expressive components of the motion. A series of perceptual experiments suggest that translation of motion capture data to a skeleton may sacrifice the "essence" of the motion. Over the several years, this paper's author has had much experience with dancers from various training backgrounds. When the motion capture system in the lab is initially demonstrated, the dancers are shown animations of unmodified motion capture data on stick figures, motion capture data mapped to an animated figure and the raw motion capture points. In most cases the dancers have voted that the raw points capture the motion most accurately. A theory for this result is that the addition of a stiff torso (i.e. the line in the stick figure or the body of the Maya animated character) adds an unnatural stiffness.

A proposed hypothesis to fix this particular perceptual problem is that the points actually captured need to be rethought. In particular, the LABAN approach to interpreting movement, as described by the dancer and author Peggy Hackney, view most movements of the body as extensions generated from the core of the body. Therefore, in order to capture the natural essence of a motion, one must capture the initiation of a movement from the root, which in many cases is the pelvis. Along these lines, it is also important to capture and understand the contortions of the body. To fully realize the impact of the distinction between motions generated from the core versus independent motions, consider a simple animated character raising an arm. If the arm is unrelated to the rest of the body, the back will not lengthen, the clavicle will not modify the shape of the back and the weight of the body will not be shifted. Instead a stick like arm separate from the rest of the body will simply rise in a manner that would be expected, let's say, on a mannequin.

The above description may suggest that modeling the dynamics of the human body needs to be included in order to accurately synthesize the interactions between skeletal parts. An alternative is to modify what is being motion captured. If an animation is to contain an "entire motion" including all body interactions, the spine, the scapula, the clavicles, the ribs, i.e. essentially the entire skeleton needs to be captured.

4.3.1 Deformable Skeleton

The basic premise of the first step in the proposal is to use machine learning on a smooth line of motion from the motion capture data that can be mapped into animations. The idea of a line of motion is a bit abstract. It is basically an imaginary line from foot to end extension of the arms that possesses the weight of the body as well as the twist through the spine and orientation limbs dominate to the motion.

The justification for this concept stems from training in visual arts. In formal life drawing classes, students are often instructed that the first step in drawing a life like human form is to draw a free form line that represents the "motion" in the body. The line captures the twist in the body, the orientation of the head and the general flow of the body. This line is vital to a successful drawing because it establishes the general essence of the pose. Once the line is in place and the orientation of the thorax and hips is established, then the exact placement of limbs and head are determined. When a full skeleton is in place, muscle deformations and other details are added.

A common occurrence in artists with a skilled eye, but lacking in formal training is the drawing of a human body one limb or part a time. These drawings often display beautifully rendered parts of the body with, for example,

highly detailed shadows and muscle. However, the entire drawing as a whole appears out of proportion and awkward. On the other hand, a less talented artist who has been trained to follow a line of motion for the body and construct the skeleton as a whole, rather than individual parts, most likely will produce a work that is wholly successful. Even if the details are not perfect, the picture will demonstrate a more accurate and expressive version of the original pose than the first described drawing.

An analogous conclusion can be drawn about the current skeletal hierarchy that is used to drive motion capture based animations. As soon as the body is viewed as individual and separate limbs and disregards the general line and weight driving a motion, the results will not be successful as a whole.

The concept of using a line of motion in the motion capture domain is not novel. In 2002, Bregler suggested manually defining shapes or drawing 'lines of motion' through cartoons to capture the general motion of the cartoon [13]. The method was used to retarget the motion of cartoons on different models. The methodology, which used a combination of affine transformation and key-shape interpolation, was distinct in that it tracked non-rigid changes in the cartoon motion. The new animations that were produced by the technique were extremely expressive and displayed the desirable exaggerations of squash and stretch.

The cartoon retargeting paper was conducted in 2-D space and required manual intervention. How does one automatically capture a line of motion in 3D space from motion capture data?

The first step is to capture the entire human skeleton accurately including the vitally important spine, scapula and clavicles. The deformations or change of length between markers must also be preserved. Unfortunately, for

many reasons, extensive use of markers and accurate tracking of the spine is impractical. It is difficult to track and use data with multiple markers that are closely spaced. The skeleton is much more difficult to work with. In addition, accurate capture of the spine is impractical and essentially impossible with the standard motion capture suit.

As a result a simple idea was constructed and now is part of a work in progress. The idea is that highly detailed motion data that includes spine deformations and integrated information about the body can be captured. This data, which takes a substantial length of time to collect, can be stored into a database. Machine learning can then be used to learn a parameterization of the spine (and body) that corresponds to the standard motion capture hierarchal skeleton. Once a parameterization is learned for a normal skeletal input, motion capture data can be processed to produce an accurate representation of the spine. This now allows any user to have a full model of the body that can be easily mapped to a character that allows flexible and deformable parts.

In order to avoid markers sliding over the skin through the suit or not being close enough to the skin to show the full deformations of the spine, highly detailed motion capture of markers glued to raw skin was performed. The markers were carefully placed on the key positions on the vertebrae of the spine, at both ends of the clavicles, around the ribs, on the clavicles and up and down the heads and limbs. Since numerous markers down the spine are used, the data is manually labeled to avoid problems with the motion capture system.

No inverse kinematics solver or skeleton should be imposed on the data. The natural deformations of bone length must be preserved. Therefore raw data, that has been cleaned to eliminate gaps or inconsistencies is exported.

A separate kinematic skeleton is fit to the data points in the capture data that correspond to points traditionally used in motion capture such as shoulders, elbows, pelvis, knee (this does not include the spine).

A neural net is used to train a system in which the input is the kinematic chain and the output is the raw motion data including the spine, scapula, etc. In addition, the output vector includes a parameterization that records deformation in marker distance. It is important that a loss function is used that favors solutions close to the existing raw motion data, but allows, with a probability, any motion in space. The learning process (1) retains the deformations in marker length seen in the raw data, preserving the natural squash and stretch parameters. The learning process also (2) maintains the interactions of body/leg/scapula/spine orientation and configuration that is naturally contained in the motion capture data. Essentially, the process is learning a perceptual version of the dynamics of the human skeleton. Hard constraints may not be preserved, but the visual aspects will appear correct.

With this system, any standard rigid body motion capture data in a hierarchical skeleton can be used as input to the system and a learned deformable body parameterization of the body including spine and scapula orientation will result. Finally, the deformable and more complete skeleton can be mapped onto a flexible model that allows for body deformations and includes handles for points down the spine and scapula.

Before details of the algorithm are discussed, it is important to note current works related to the proposed paper. The idea of mapping skin deformations onto character animations was proposed by Sand and collaborators in 2002 [67]. The method involved combining motion capture data with camera

output to produce deformable human models that changed according to movement. Similarly, this thesis strives to deform the skin by accurately learning and training the full skeleton. We seek animated characters whose backs deform and reshape accurately with motion. A criticism of the Sand paper is that the lack of accurate skeletal input led to body deformations that appear unnatural. A more complete and integrated skeletal model may alleviate this inconsistency.

In 2004, Grochow, Martin, Hertzmann and Popovic presented a paper on Style-Based Inverse Kinematics [35]. The paper presented an inverse kinematics system that was based on a learned model of human poses. The system produces the most likely pose satisfying user constraints. The learned model is a probability distribution over the space of all possible poses. The notation of style comes in to play because training the model on different input data leads to different styles of Inverse Kinematics. The fact that the model is a learned distribution and that the system favors poses that exist in the training set even though any pose is possible were motivating factors for the techniques adopted in the thesis proposal. The Grochow paper used Scaled Gausssian Process Latent Variable Models (SGPLVM) to represent the data sets [43]. This paper approaches the problem with different learning procedures, but produces analogous results.

4.4 Exaggerating Subtleties

The procedure outlined in step 1, has aims of including the expressive qualities of a deformable skeleton in character animation. However, there are other fundamental visual cues that the eye subconsciously needs to see to be

convinced of a realistic motion. These visual cues include muscle deformations that correspond to changes in the skeleton. This paper's author notes that many systems exist for solving the kinematic equations for muscles according to bone movements. Perhaps such a tools can be integrated with the deformable skeleton procedure to create a post processing method that constructs a fully deformable body.

If the learning procedures for learning more detailed aspects of motion such as spine deformations are successful, than data driven learning of other involuntary motions is a natural extension of the work. This learning can be extended to learning breathing and eye movement. Post process tools to layer these subtle phenomena on top of existing motion capture data will enable a more pipelined and realistic approach to data driven animation.

4.5 LABAN

The motion capture group suggested that the concept of applying the effort parameters from LABAN movement analysis onto character animations as demonstrated in the 2000 paper on the EMOTE system was well suited to motion capture applications. By motion capturing a set motion phrase several times with varying effort parameters, it was suggested that the individual LABAN parameters could be learned and applied in a meaningful manner. The group proposed a system of knobs with tunable LABAN effort parameters for motion that can be applied to motion capture data. A drawback to this method may be that LABAN parameters have little meaning to an animator and that key framing parameters over time could be extremely time consuming. While research continues and will hopefully be successful, initial experiments suggested that isolating individual LABAN parameters is more difficult than

original anticipated.

On the other hand, discussions with the LABAN expert Peggy Hackney resulted in the discovery that most almost all people have a 'motion signature.' LABAN effort parameters almost never appear in isolation. A motion signature is a combination of 2 or 3 LABAN effort parameters in which the majority of a person's actions fall. Peggy reviewed numerous film clips with the author and showed time and time again that different individuals consistently fell within one of the 10 states or drives that define motion signature. For example, one person may have particularly strong and bound movements in every action they perform. Another may very free and direct.

The basic concept of how to apply this information is still murky. One alternative is to use machine learning to classify motion capture data into one of the 10 personality categories. Any motion entered into a motion library can automatically be classified using the learned machine. Using current motion separation techniques the data can be subdivide into meaningful segments. In order to construct new motions, only segments classified as a particular personality are used to blend together new motions. The 'personality' can be exaggerated or diminished using linear interpolation/extrapolation techniques.

CHAPTER 5
POSTURE REPRESENTATION

5.1 The Motion Capture System

In previous sections a crucial aspect of motion capture was discussed: the mapping of randomly placed marker positions in 3-D space to a fixed rigid body skeleton or kinematic chain. However, it is first relevant to discuss the motion capture system and setup used in all subsequent section's projects. All research was conducted with a 10 camera motion capture system. Motions of subjects were recorded at a frame rate of 120 HZ. It is important to note that the VICON system is capable of a much higher frame rate (up to 1000 frames per second) in ideal conditions. The given frame rate was selected because it was a standard for many applications and little no motion information in subjects was sacrificed.

The motion capture markers are made of a retro-reflective material. The VICON cameras shine a visible red light, which reflects off the markers. Each of the 10 cameras record the 2-D location of the reflecting markers. The 3-D position of the markers is calculated by triangulating between the 10 camera images. A calibration phase is first performed before subjects are recorded. During calibration, including a static and dynamic phase the focal lengths of the cameras and the actual size of the recording space are determined.

In a typical motion capture session a body suit of suitably skin tight fabric is worn by the subject. The reflective markers are attached to the suit

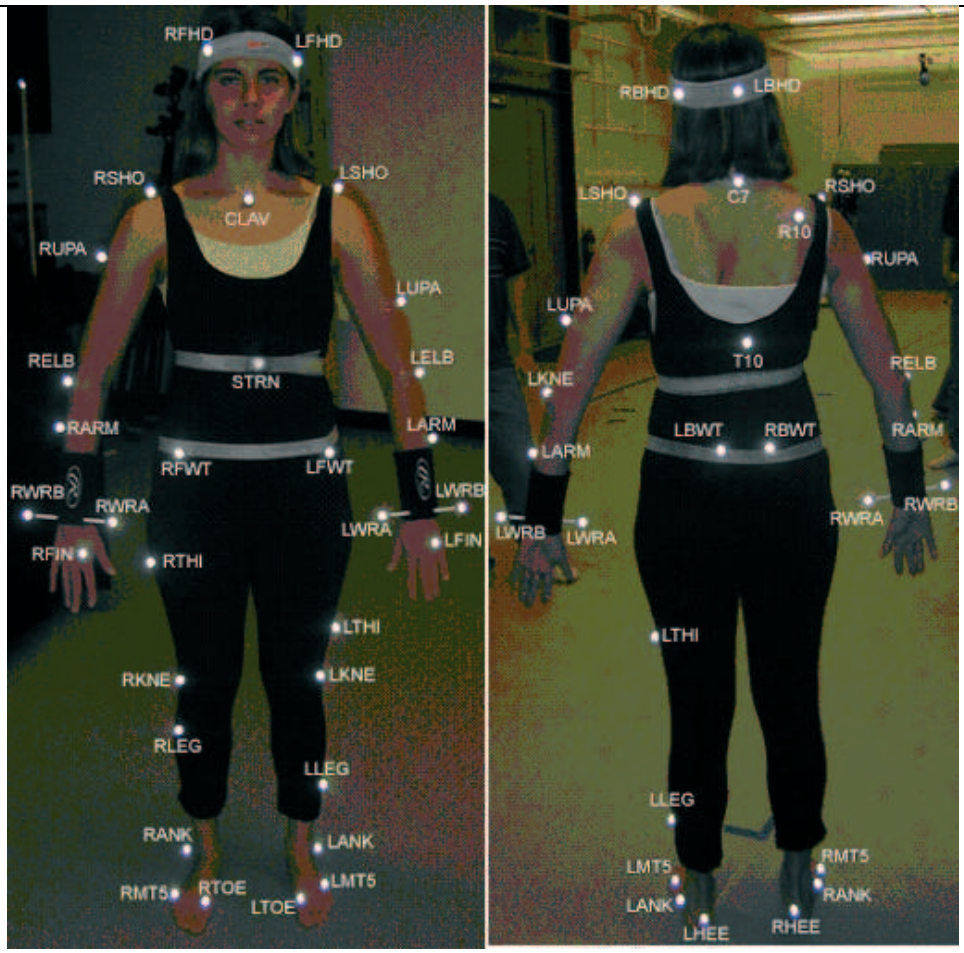

Figure 5.1: Placement of a Standard Marker Set on Subject

(usually Velcro). An image of the names and placement of a standard motion capture marker set is provided in figure 5.1.

In conjunction with the VICON camera system, the VICON IQ software was used for recording motion capture and post-processing the data. The 3-D reconstructions produced by the camera system need to be attached to the corresponding marker labels. This requires two procedures : 1. a user manually labeling markers in a given pose for a single frame and 2. tracking the points across all frames. Usually, the points in a marker set are labeled

when the subject is in *T-Pose*. In T-Pose, the subject is standing with arms stretched out and legs together as if in a T-shape.

Ideally we would like the hand labeled markers to line up with the corresponding 3-D point reconstructions for all frames in a motion recording. However, this can be a complicated task for a variety of reasons. Since the points, marker positions, are moving from frame to frame (sometimes dramatically), it is easy to loose track of where a point is going. In complicated movements, several markers may align in the same position, and movements of the body can easily occlude marker positions for long periods of time. As a result a tracking method is needed that calculates the path of a given reconstruction over time and assigns the reconstruction the appropriate label.

5.1.1 Tracking

The data used in all projects presented in this paper is initially tracked using the VICON IQ software. The actual algorithm used is not available. However, it is most likely based on one of the two schemes presented below. Additional tracking is used both for smoothing and as part of a learning scheme in one of the projects. As such, it is important to review the techniques here.

KALMAN FILTER

A standard and fairly simplistic tracking scheme that is frequently used by the vision community is the Kalman filter. The Kalman filter is an extension of a Bayesian dynamical system. The basic premise of a linear Kalman filter is that a stochastic difference equation is used to estimate the state of a discrete-time controlled process by updating in two separate stages. The process is updated specifically with an autoregressive or AR(1) model. A measurement of a state vector (in our case the physical location of a reconstruction) is used

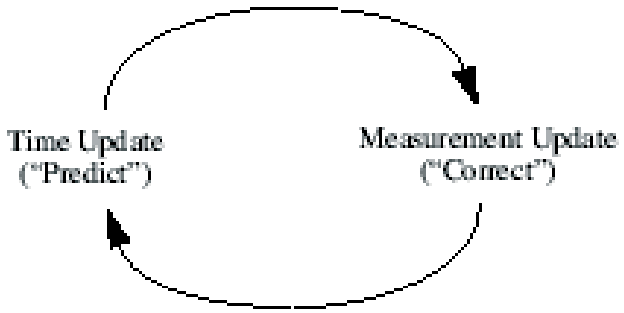

Figure 5.2: The discrete Kalman filter cycle. The time update step projects the current state estimate ahead in time. The measurement update modifies the projected estimate by an actual measurement.

to calculate the *Kalman gain* function and then update the state vector and error covariance. The proceeding time update step involves predicting the next time step value using the AR(1) model and then predicting the error covariance ahead. The filter cycles between these states updating each state with the new input measurement [76] as illustrated in Fig. 5.2.

Given a state $x \in \mathbb{R}^n$, the process is updated in time by the AR(1) equation :

$$x(t) = Ax(t-1) + Bu(t-1) + w(t-1)$$

with a measurement $z \in (R)^m$ such that

$$z(t) = Hz(t) + v(t)$$

The variables $w(t)$ and $v(t)$ represent the process and measurement of noise. The two variables are assumed to be gaussian in nature, independent from each other and have zero mean.

$$p(q) \sim N(0, Q)$$

$$p(v) \sim N(0, R)$$

The $n \times n$ matrix A is the difference equation which relates the current state to the previous time step. For our purpose, this is an estimation of position based on the previous state position and velocity. In certain applications, the value of A can be updated with each time step, but for our purposes it is assumed to be constant. The $m \times n$ matrix H transforms the current state to the measurement. Finally, Q is the process noise covariance matrix and similarly R is the measurement noise covariance matrix.

The time update equations are essentially the predictors and likewise the measurement update equations act as the correctors. The initial estimates for state and error covariance, $\hat{x}(t-1)$ and $P(t-1)$ are fed into the time update equations. The time update equations, are as follows.

1. Project the next state in time

$$\hat{x}^-(t) = A\hat{x}(t-1) + Bu(t-1)$$

2. Project the error covariance ahead

$$P^-(t) = AP(t-1)A^T + Q$$

Similarly the measurement update equations in the prediction step are listed below.

1. Calculate the Kalman gain function

$$K(t) = P^-(t)H^T(HP^-(t)H^T + R)^{-1}$$

2. Use the measurement to update estimate

$$\hat{x}(t) = \hat{x}^-(t) + K(t)(z(t) - H\hat{x}^-(t))$$

3. Update the error covariances

$$P(t) = (I - K(t)H)P^-(t)$$

In the Kalman filter equations outlined above, an assumption of linearity was assumed. However, the Kalman filter can be extended to nonlinear cases by changing the state update equation. It should also be noted that the covariances and Kalman gain function quickly stabilize and are relatively constant throughout the process.

MULTIPLE HYPOTHESIS TRACKING(CONDENSATION)

A drawback to the Kalman filter is that it only allows for a single hypothesis for both the measurements and actual location at each time step. When there are occlusions or several moving bodies in a similar orientation, the Kalman filter will fail. Another scheme, the *Condensation* Algorithm, or Conditional Density Propagation, allows general representations of probability distributions.

Condensation uses a particle filter, a well known technique in Monte-Carlo methods, to sample the conditional state-density distribution which is normally computationally infeasible. The process uses simple Bayesian statistics to predict the future time step based on the current measurement and previous time step.

A general assumption is made that the object dynamics in the system form a temporal Markov chain. With a state vector x, the probability distribution is :

$$p(x(t)|X(t-1)) = p(x(t)|x(t-1))$$

The observations z are assumed to be independent . The probability is calculated as :

$$p(Z(t-1), x(t)|X(t-1)) = p(x(t)|X(t-1)) \prod_{i=1}^{t-1} p(z(i)|x(i))$$

Given this Markov chain with independent observations, the conditional state density at time t is defined by $p_t(x_t) = p(x(t)|Z(t))$. Using Bayes rule, we can

calculate the propagation of the state density over time as:

$$p(x(t)|Z(t)) = k(t)p(z(t)|x(t))p(x(t)|Z(t-1))$$

where k is a constant representing the inverse of the probability of the observation. We update the conditional state density with the equation:

$$p(x(t)|Z(t-1)) = \int_{x(t-1)} p(x(t)|x(t-1))p(x(t-1)|Z(t-1))dt$$

Since the algorithm allows a general probability framework, the observation state density $p(x(t-1)|Z(t-1))$ is non-gaussian. There are several means of estimating this probability distribution. Some tracking procedures employ various forms of 'multiple hypotheses tracking'. However, the condensation algorithm uses factor sampling to estimate the non-gaussian distribution.

Factor sampling works by randomly sampling a series of points, $s(i)$, from a prior density function, $p(x)$. The samples are each assigned a weight that is in proportion to the value of the observation density, $p(z|x = s(i))$. The weighted point set is used as a representation of the posterior density $p(x|z)$.

5.1.2 Cleaning

The tracking procedure serves two purposes: it enables the user to track a single point across time frames and it smooths the data, eliminating some of the measurement noise. Unfortunately, certain camera angles and poses can cause severe occlusions of markers. Therefore, manually cleaning of the data is often a post-processing necessity.

Cleaning of the data usually entails filling in segments of missing motion from a marker. A common technique includes interpolating between the motion frequency bands of neighboring points. If the missing motion constitutes a minimal number of frames, another alternative is to fill the space with a spline matching the chosen endpoints.

Finally, various issues including camera angles, the number of markers placed on the subject, camera frequency, etc., can produce extremely jittery motion in certain markers. The cleaning process often includes removing or smoothing these high frequency motion anomalies. One technique is to predict the actual state of the marker and do smoothing based on the predicted true position similar to the Kalman filter. A less optimal alternative is to run a moving average across the entire time series. Another method is to simply remove the high frequencies by running a low bandwidth filter.

5.1.3 Skeleton Model

After the data is tracked, labeled and cleaned, it can be fit to a human skeleton. A skeleton is constructed with rigid bone lengths that represent the actual degrees of freedom in the underlying motion model. It is important to make a distinction between the motion capture marker labels and the actual skeletal points. The motion markers are grouped to approximate the actual joint positions and rotations. This is done using an optimization procedure that assumes fixed bone length.

An example of a standard marker set used with the VICON motion capture system and a constructed skeleton that has been fit to the data is in figure 5.3. The skeleton used has been constructed manually using VICON software. The actual bone lengths are determined during a calibration procedure that optimizes over a set of motion capture data (the motion capture set used for calibration is usually called 'range of motion' and includes full movement of the limbs).

The existence of the skeleton serves many purposes. Once the skeleton is calibrated, it can be quickly fit to other motion capture data of the same

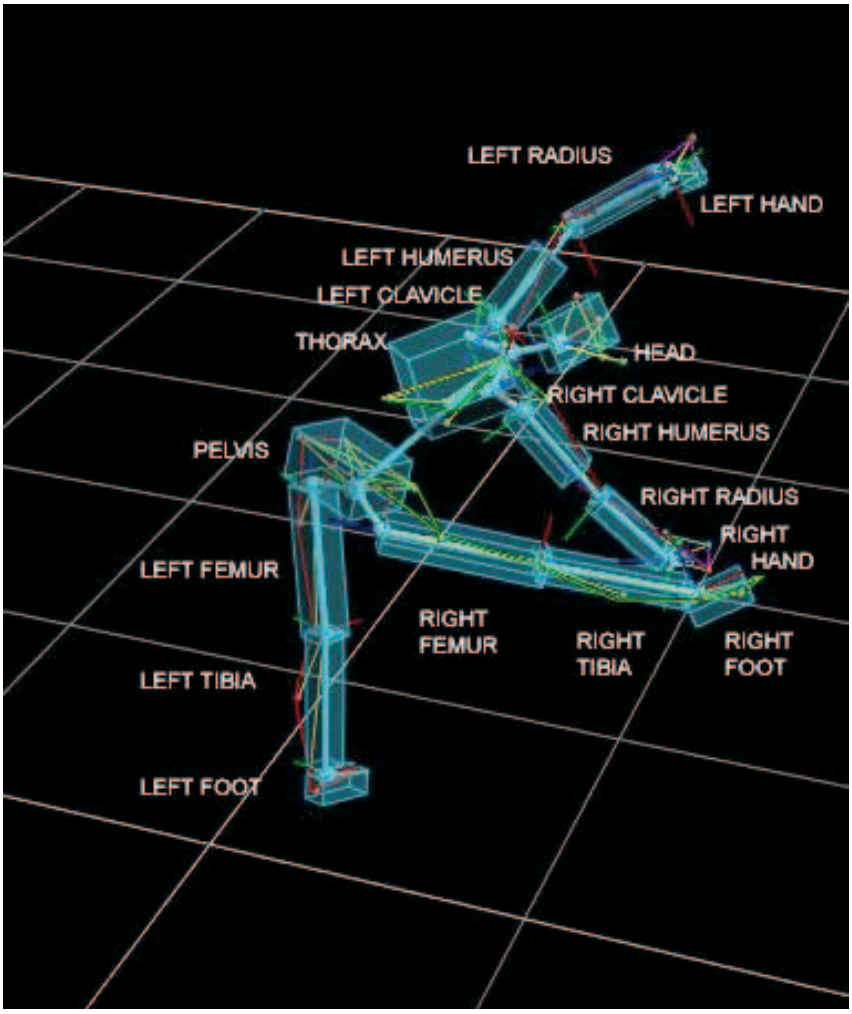

Figure 5.3: A Standard Skeleton with Labels fit to a Subject in VICON IQ

subject. The skeleton allows firm placement of joints so that the motions executed by a performer can be mapped onto arbitrary animated characters.

5.2 Kinematic Chain

When manipulating the motions of a skeleton, the user is often not interested in the motion of individual joints or segments, but rather in the collective motion of the entire subject. The assumption used is that the skeleton is a *rigid body* meaning that the distance between any two joints remains constant despite forces exerted about the body or various segments. Formally, this means for any two joint positions $p = [p_x, p_y, p_z]$ and $q = [q_x, q_y, q_z]$:

$$||p(t) - q(t)|| = ||p(0) - q(0)|| = c$$

where $c = constant$.

In dealing with human motion data, we want to allow the joints and segments in the body to rotate relative to each other but still maintain the rigid motion constraint. A common technique is to represent the body as a *kinematic chain*. All translations are performed in world coordinates and applied to the entire rigid body. The subject has a specific orientation to the world coordinate system. This orientation includes the translation and a rotation of all segments/joints.

Figure 5.4 shows the kinematic chains used for a human subject in the projects in this paper. The chains are based on the joints in the standard skeleton. This particular construction contains a total of five separate chains. In each chain, the rotation of the parent joint is translated to the child. Note that in this formulation the 'PELVIS' segment acts as the root and is the starting point for each chain. The chains, reading from root to child, are :

```
CHAIN(1) = (PELVIS, THORAX, HEAD, headend)
```

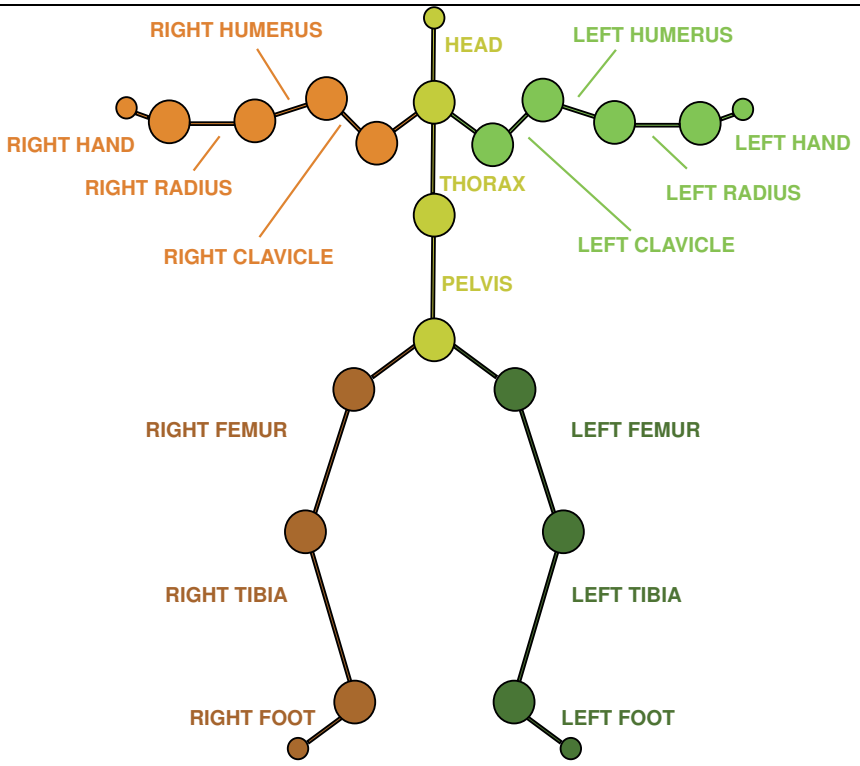

Figure 5.4: Kinematic Chains in Standard Skeleton

```
CHAIN(2) = (PELVIS, THORAX, LEFT CLAVICLE, LEFT RADIUS, LEFT
   HAND, lefthandend)
CHAIN(3) = (PELVIS, THORAX, RIGHT CLAVICLE, RIGHT RADIUS, RIGHT
   HAND, righthandend)
CHAIN(4) = (PELVIS, LEFT FEMUR, LEFT TIBIA, LEFT FOOT,
   leftfootend)
CHAIN(5) = (PELVIS, RIGHT FEMUR, RIGHT TIBIA, RIGHT FOOT,
   rightfootend)
```

The lower case names in the chain components represent the chain terminations and do not have a corresponding rotation associated with the joint.

The benefit of using this representation is that each joint is represented

as a local rotation off of the parent joint in a given frame. This eliminates redundancy in the representation. A rotation about the right femur will automatically be applied to the entire leg. This coarse to fine approach enables flexibility in managing the data and animating new characters. It should also be noted that this parameterization is a necessity when applying learning procedures later in the paper. When applying procedures to reduce the dimensionality of the dataset, we want all correlations between rotations to be minimized.

The local rotations of each of the joints are calculated based on the global rotations (in world coordinates) determined by the skeletal model. Each rotation (local) in the chain is a product of the parent rotations. As an example, we can formulate the global (world rotation) of the RIGHT FOOT as a product of the local rotations in the right leg chain :

$$R^G_{rightfoot} = R^L_{rightfoot} * R^L_{righttibia} * R^L_{rightfemur} * R^G_{pelvis}$$

The physical position of the joints can be calculated using the fixed bone segment lengths determined in fitting a skeleton. The position of a particular element in the body is determined by summing the translation on the root of the kinematic chain (in this case the pelvis) and applying the global rotation to a vector representing the bone segment in the root position.

$$[x(t), y(t), z(t)]_{rightfoot} = [x, y, z]_{pelvis} + R^G_{rightfoot} * [x(0), y(0), z(0)]^T_{rightfoot}$$

5.3 Rotation

A crucial decision in our kinematic chain representation is how to parameterize the joint rotations. Numerous parameterizations exist and which is used is dependent primarily on the application of interest. The applications utilized in this thesis seek to encode orientations and describe rigid body

motion through kinematic chain hierarchies. In addition, learning using the chosen rotation parameterization will be performed. However, rotations are non-Euclidean and therefore the parameterization can yield difficulties when using standard algorithms such as linear interpolation. For example, Euler angles are among the most popular methods for parameterizing rotation and give an intuitive representation of a 3-D dimensional movement. Yet, they suffer from severe disadvantages including *gimbal lock*, which make them undesirable candidates for learning algorithms.

This section will enumerate the advantage and disadvantage of representing 3-dimension rotations in the following formats :

1. Coordinate Matrix
2. Euler Angles
3. Axis-Angle (Exponential/Log Map)
4. Quaternions

Rotations in the 3-dimensional Euclidean space ($\mathbb{R}$) form a group $SO(3)$ of *special orthogonal* 3×3 matrices. An orthogonal matrix is always invertible and has column vectors of unit magnitude which form an *orthonormal basis* for the rotated space. Orthogonal matrices are defined mathematically by the following property, where $A = 3 \times 3$ orthogonal matrix :

$$AA^T = I$$

where I = the identity matrix. Therefore, the inverse of an orthogonal matrix is equivalent to the transpose :

$$A^{-1} = A^T$$

$$det(A) = \pm 1$$

The subset of orthogonal matrices that have exclusively a determinant equal to 1 ($det(A) = 1$) are labeled *special* orthogonal matrices. Therefore a coordinate matrix encoding a rotation has 9 components, is invertible (with an inverse equal to its transpose) and has a determinant of 1.

In the previous section, we alluded to the fact that rotations must be read from right to left and that multiplying two rotation matrices produces a new rotation matrix. This also implies that rotations do not demonstrate the *commutative* property. Generally, applying rotations in a different order produce different results.

$$R_a R_b = R_{ab}$$

$$R_a R_b \neq R_b R_a$$

In fact, reversing the order in applying rotations in graphics applications can be the difference between applying a rotation in the local axis frame versus the global axis frame.

However, a matrix representation of rotations includes a total of 9 parameters (values in a 3×3 matrix). Euler's theorem (below) establishes that a 3-dimensional rotation can be described with a minimum of 3 parameters.

> *Euler's Theorem* Every displacement (or orientation with respect to a fixed frame) of a rigid body can be described as a rotation by some angle θ around some fixed axis $\hat{n}$.

This theorem is also interpreted as stating that a rotation can be represented by only (a minimum of) three parameters. Two parameters represent the rotation axis and one parameter represents the planar rotation perpendicular to the defined axis. *Euler's Rotation Theorem* also states that in 3 space, for any two coordinate systems with a common origin, there exists a single

vector which is the same in either system. This single vector by definition is the eigenvector of a rotation matrix.

Recall that if A is a linear transformation in matrix form, there exists a vector $x \in \mathbb{R} \neq 0$ such that :

$$Ax = \lambda x$$

where the scalar λ is defined as the *eigenvalue* of A and x is the corresponding *eigenvector*. The SO(3) rotation matrices have rank 3 and therefore 3 eigenvalues with one real eigenvalue equal to one. The other two eigenvalues are complex numbers with values of $e^{\pm i\theta}$. The eigenvector corresponding to the eigenvalue of one is the axis of rotation for the matrix (or the set of points that remain static under the rotation). The complex eigenvectors form a plane orthogonal to this axis.

5.3.1 Coordinate Matrix

Representing a joint rotation as a 3×3 coordinate matrix initially appears ideal. The matrix representation of SO(3) is used to define rotations. This presents a strong advantage in that a matrix in SO(3) maps 1-to-1 onto the angular displacements of rigid bodies. The other rotation parameterizations discussed here do NOT have this 1-1 property.

The disadvantages of the coordinate matrix include the nine parameters of the matrix. Euler's theorem demonstrates that the rotations inherently involve only 3 degrees of freedom. Therefore we have to enforce 6 constraints, the orthonormal nature of the columns and the determinant being positive one. In addition, the group of SO(3) is not a vector space. The group of SO(3) forms a *Lie group* (the importance of the Lie group is made clear with the quaternion representation). A vector space requires that linear combinations

and operations be closed under the space. Therefore, standard algorithms, such as linear interpolation can not be performed with coordinate matrices. Combining two matrices with the addition operation will yield a matrix that violates the definition of SO(3). Finally, the extra computation involved in combining matrices increases the numerical issues. Computational error means that the matrices may drift away from the special orthogonal group and re-normalizing the matrix is computationally expensive.

5.3.2 Euler Angles

An extremely popular parameterization of a rotation in graphics involves dividing the rotation into three sequential and separate rotations about the standard orthogonal axes (x, y, z). The complete rotation is then represented by 3 *Euler* angles, such as (ψ, θ, ϕ), which each represent the angle of rotation about one of the fixed axes. The notation is based on Euler's Rotation Theorem : since any rotation has three degrees of freedom, these 3 Euler angles can specify any rotation in 3 space as shown in figure 5.5. A complete rotation is accomplished by multiplying 3 rotation matrices :

$$R = R_x(\psi)R_y(\theta)R_z(\phi)$$

In this particular example, the order of rotations is called Z-Y-X . The above equation is literally a rotation around the z-axis by the angle ϕ, followed by a rotation $\theta \in [0, \pi]$ around the y-axis, finally followed by a rotation about the x-axis with angle ψ. As was discussed earlier, rotations do not commute, so that applying the Euler angle rotation matrices in a different order would yield a different final 3-dimensional rotation. The form of the matrices used

in the Euler angle parameterization are as follows:

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos(\psi) & sin(\psi) \\ 0 & -sin(\psi) & cos(\psi) \end{pmatrix}$$

$$R_y = \begin{pmatrix} cos(\theta) & sin(\theta) & 0 \\ -sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R_z = \begin{pmatrix} cos(\phi) & -sin(\phi) & 0 \\ sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

An Euler angle set consists of any product of these 3 matrices conditioned on the fact that no consecutive matrices have the same principle axis of rotation. Unfortunately, this result implies that there are a total of twelve possible Euler angle orders. The possible rotation orders include Z-Y-X, X-Y-Z, Y-X-Z, Z-X-Y, X-Z-Y, Y-Z-X, Z-Y-Z, Z-X-Z, X-Y-X, X-Z-X, Y-Z-Y, Y-X-Y. It should be noted that the Euler Angle convention stipulates that each rotation is around the x, y, or z axes in the world orientation. A corresponding *moving axes* representation would rotate around the local axes of the body. However, once again the difference between a local or world rotation is only a matter of the order in applying the rotations. A rotation in the world coordinate frame is equivalent to the reverse order or rotation operation in the local or body coordinate frame.

$$(R_z R_y R_x)_{world} = (R_x R_y R_z)_{body}$$

The intuitive nature of Euler angles make them a desirable representation

for rotation. An orientation can quickly be visualized and understood using Euler angles. This property allows an easy interface for animators where each angle can be plotted as a function of time or be represented as 3 independent sliders. Changing an angle from 40 to 45 degrees can easily be understood. As such, Euler angles are the preferred method of representing angles in animation packages. Unlike coordinate matrices, Euler angles only contain 3 parameters.

Despite the stated advantages, Euler angles are always a poor choice for applications that involve combining rotations. The Euler angle representation contains an inherent singularity, which results from a loss of a degree of freedom by concatenating three fixed single-axis rotations. Whenever 2 of the 3 rotation axes align, a degree of freedom is lost and the result is called *Gimbal Lock*. Consider the following example of how the axes could align: suppose one first rotates 90 degrees (or $\frac{\pi}{2}$) around the z axis followed by a rotation of 90 degrees around the y axis. At this point the x axis has aligned with the original z axis. It is impossible to find any new orientation from this position that could have not been produced by simply rotating around the z axis in the first place.

A seeming straightforward solution to this problem is to place limits on the range of motion of the Euler angles. However, gimbal lock will occur when the second rotation in a Euler angle sequence has either value 0 or $\frac{\pi}{2}$ depending on the choice of Euler angles. As such, even with limits, gimbal lock is impossible to avoid. In fact, gimbal lock will occur with any choice of fixed axes - the only way to avoid such an occurrence is to add another degree of freedom.

Finally, the Euclidean nature of Euler angles is appealing but yields poor results since they do not accurately reflect the non-Euclidean nature of SO(3).

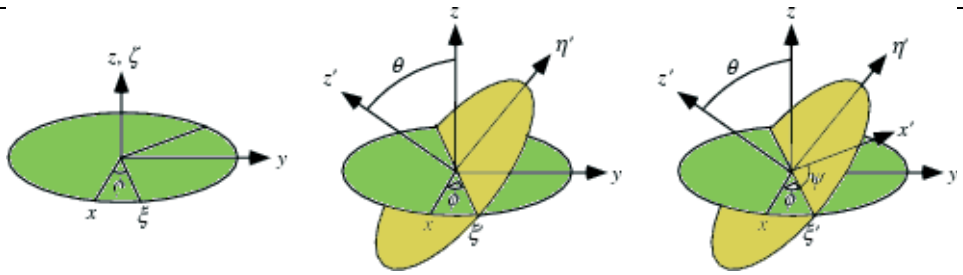

Figure 5.5: Euler Rotations around x, y, and z axes (image from Wolfram MathWorld)

Interpolation of Euler angles produces jittering movements because Euler angles interpolate around each of the axes independently and ignore the inherent interaction between axes. While Euler angles are suitable for Ordinary Differential Equations, the singularity of the representation is undesired in most applications.

5.3.3 Quaternions

In 1843, Sir William Rowan Hamilton discovered quaternions, a group in mathematics that enables triples of real numbers, for example (x, y, z) to be multiplied and divided by using three imaginary units and one real number. The underlying set in quaternions is $\mathbb{R}(4)$ and the group operates with a multiplication operator $\circ$ that combines the dot product and cross product of vectors. The set of quaternions with magnitude equal to one, or rather of unit length, form a subgroup whose underlying set is $S(3)$. $S(3)$ possesses the same local geometry and topology as $SO(3)$ and therefore is an extremely desirable space to interpolate rotations.

A quaternion can be written as a sum of real and imaginary components.

$$Q = w + xi + yj + zk$$

where $w, x, y, z \in \mathbb{R}$ and i, j, k are distinct imaginary numbers such that $i^2 + j^2 + k^2 = ijk = -1$. A quaternion is often decomposed into a scalar for the real component and a 3-vector for the imaginary parts :

$$Q = w + \hat{v} = w + x\hat{i} + y\hat{j} + z\hat{k}$$

In this paper, a quaternion will often be written as (q_w, q) or $q = [q_w, q_x, q_y, q_z]$. Unit quaternions have the useful property, as in $SO(3)$, that the inverse is equivalent to the conjugate :

$$Q^{-1} = Q^*$$

Note, that the conjugate of a quaternion is $Q^* = w - v$.

An interesting property of the quaternion group is that a $3 - vector$ can be rotated simply by using quaternion multiplication. If we have a vector $x \in \mathbb{R}(3)$ and a unit quaternion q representing a rotation in 3 space, the vector is rotated by the following operation :

$$x' = Rotate(x) = q \circ x \circ q^*$$

In addition, computing a rotation matrix and partial derivatives from a unit quaternion is trivial. The four partial derivatives of the unit quaternion are also linearly independent. This is crucial because it implies that unit quaternions *are free from gimbal lock*.

Unit quaternions are almost an ideal parameterization for 3 dimensional rotations. The nature of the space makes them the best candidate for performing interpolations. The results are pleasing to the eye and free from gimbal lock. However, there are drawbacks because only a subspace of the full quaternion space actually represent rotations. The extra degree of freedom can lead to certain difficulties during calculations. In particular, the nice formulation of derivatives depends on the quaternions remaining in $S(3)$, which means that unit length must be preserved during all calculations with quaternions.

Given that a quaternion has four degrees of freedom, but only 3 rotational degrees of freedom, algorithms manipulating quaternions can easily move the quaternion off the unit sphere, which no longer results in a rotation. In applications involving ordinary differential equations or dynamics, calculating the instantaneous rotations can be hazardous because the derivative lies in the tangent plane to $S(3)$ and movements in the plan will push the quaternion off the sphere.

These situations are targeted by renormalizing the quaternion after every integration step. Similarly, the extra degree of freedom is addressed by imposing strict constraints of unit length in all algorithms. However, these solutions come at the cost of increased complexity and decreased efficiency.

Finally, quaternions form a *group* and not a vector space. The nature of the space is nonlinear and leads to difficulties when performing operations that require a linear space. Despite, the limitations quaternions are the representation chosen for the majority of work in this paper.

5.3.4 Exponential Map (Axis Angle Representation)

The exponential map maps a vector from $\mathbb{R}(3)$ to $S(3)$ and is formulated from the quaternion representation of rotations. First note that quaternions can be written in polar form :

$$q = re^{\hat{n}\frac{\theta}{2}} = r(cos(\frac{\theta}{2}) + \hat{n}sin(\frac{\theta}{2}))$$

The value for r is the magnitude and hence 1 when dealing with quaternions in $S(3)$. The value $\frac{\theta}{2}$ is the angle of the quaternion and $\hat{n}$ is the axis of rotation.

The exponential function is the same as with a matrix exponential and is a formal power series :

$$e^{\frac{\theta}{2}\hat{n}} = 1 + \frac{\theta}{2}\hat{n} + \frac{(\frac{\theta}{2}\hat{n})^2}{2!} + \frac{(\frac{\theta}{2}\hat{n})^3}{3!} + ...$$

The exponential map is the physical quantity

$$ln(q) = ln(e^{\frac{\theta}{2}\hat{n}}) = v = \frac{\theta}{2}\hat{n}$$

where

$$q = (1, 0, 0, 0) = e^{(0,0,0)}$$

and for $v \neq 0$

$$e^v = (cos(\frac{\theta}{2}), sin(\frac{\theta}{2})\hat{n})$$

The problem with this formulation is that as θ goes to zero the calculation of the axis $\hat{n}$ becomes numerically unstable. The exponential map has singularities on the spheres of radius $2n\pi$, since a rotation of 2π about any axis is equivalent to no rotation at all.

Therefore, when dealing with the exponential map representation of a rotation we must restrict our parameterization to inside a ball of radius 2π. Inside this ball, each member of $SO(3)$ has two parameterizations. The first is a rotation of θ about the axis v and the second is a rotation of $2\pi - \theta$ about the axis $-v$.

If the magnitude of the mapping is 1, the exponential mapping has many desirable properties. It contains 3 degrees of freedom and exists in $\mathbb{R}(3)$. The mapping represents a ball of radius 2π where each point inside the sphere is a unique rotation. However, as mentioned above, there are duplicate representations of the mapping on the surface of the ball and the exponential mapping representation must be used with care because of these singularities.

Another difficulty is that the straight line between two orientations in the exponential map space of $\mathbb{R}(3)$ is not equivalent to the geodesic distance between orientations in $S(3)$ [34]. The main impediment is that each orientation of $SO(3)$ maps to antipodal solutions, which in the case of quaternions, is the quaternion and equivalent conjugate. However, with the exponential map

formulation, each rotation can map to an infinite number of points. While we can restrict the range of the mapping, the procedure does not guarantee that the geodesic approximation picks the mapping that minimizes the euclidean distance. This may make the exponential map approximation unsuitable for interpolation procedures.

On the other hand, the desirable linear properties of the space make it ideal for certain learning applications. As such, this paper uses mainly a quaternion parameterization for rotations, but converts the data to an exponential map representation when applicable.

CHAPTER 6
LEARNING WITH QUATERNIONS

In the previous chapter, the advantages of using quaternions or exponential maps versus an Euler angle representation of rotations were enumerated. When performing dynamic simulations, utilizing statistical analysis or undertaking machine learning applications, the risk of combining these techniques with Euler angles can be perilous. Gimbal lock is almost assured in a situation where for example a 'mean' representation of a rotation is required. The process of adding together multiple angles to from rotations which most likely sum to way over 360 degrees in each direction yields meaningless results.

In certain applications quaternions are always the preferred representation. When interpolating smoothly between joint angles in a certain range, spherically interpolating between quaternions produces the most visually satisfactory transitions. Unfortunately, quaternions present certain difficulties. A quaternion has four degrees of freedom, but maps to the space $SO(3)$ which only contains 3 degrees a freedom. The extra degree of freedom implies that each rotation has multiple representations. Unit quaternions, where $||q|| = 1$, are points that exist on a unit hypersphere. Each quaternion has a corresponding representation at the opposite pole or other hemisphere of the sphere. For clarification purposes, this 4 dimension hypersphere, is called $S(3)$, because the surface has 3 degrees of freedom even though it is embedded in a 4 dimensional space.

Since exponential maps have only 3 degrees of freedom, the representation circumvents some of the problems encountered when using pure quaternions. However, exponential maps also have disadvantages. Strict boundaries must be placed on the representation to ensure that it is inside the sphere and not near the surface. Evidence demonstrates that while the approximations may suffice in most application, interpolating with exponential maps is a dangerous practice. Even when boundaries are strictly enforced, the results often are not the shortest euclidean path between rotations

The majority of work in this thesis relies on the quaternion representation of angles. In certain applications, the nonlinear nature of the quaternion space and additional degree of freedom requires alternative forms of representation to perform calculations. In these cases, an exponential mapping or log mapping is performed so that the angle data is both in a linear space and contains only 3 degrees of freedom. This representation is particularly apt because the log map is a mapping to the tangent space of the quaternion.

Recall that a quaternion can be written as the exponential of a pure vector and similarly we can define a mapping with this vector by using the logarithm, ln, function.

$$ln(q) = ln(e^{\frac{\theta}{2}\hat{n}}) = \frac{\theta}{2}\hat{n}$$

The logarithmic map (or exponential map - depending on naming convention) maps a point on the quaternion hypersphere into the 3-dimensional tangent space at the identity quaternion. This tangent space, as mentioned earlier, is in fact a vector space $\mathbb{R}(3)$. Mapping a quaternion into the tangent space at locations other than the identity, $q = [1, 0, 0, 0]$, involves first rotating the quaternion to the desired point on the sphere. To rotate a quaternion q to align with a quaternion point p on the sphere, we simply perform the operation

$P^* \circ Q$. Therefore the formula for mapping a quaternion into the tangent space at point p is :

$$ln(P^* \circ Q)$$

Since quaternions are intrinsically non-linear, special consideration has to be taken when approaching standard learning and dynamics applications. A first step in dimensionality reduction is forming a mean of clusters of data. However, determining a 'mean' configuration with quaternions is more difficult than linearly adding together items and dividing by the number of components. This chapter addresses the methods and sometimes differing approaches in accomplishing these basic learning and dynamics operations when using quaternions. These operations include necessary algorithms for interpolation, performing statistical analysis and time derivatives for dynamics.

6.1 Quaternion Operations

A few operations with quaternions should be addressed before proceeding with algorithms for animating and learning applications. Quaternions are members of a Lie group, which is defined as a group that is also a smooth manifold. A group is a set that has a binary operation, in the case of quaternions, $\circ$ or quaternion multiplication, such that the binary operation maps any two members of the group into the group. In addition the operation of quaternion multiplication must obey the axioms of associativity and closure(mentioned above). Finally, the group must have an inverse for each element and an identity element.

The quaternion group is closed under the operation of quaternion multiplication which is defined as follows :

$$q_1 \circ q_2 = w_1 * w_2 - \hat{v}_1 \bullet \hat{v}_2 + w_1\hat{v}_2 + w_2\hat{v}_1 + \hat{v}_1 \times \hat{v}_2$$

The identity element is simply :

$$q = [1, 0, 0, 0]$$

In the case of unit quaternions, which are used exclusively for representing rotations, the inverse is simply the conjugate q^*

$$q^* = w - \hat{v}$$

$$q \circ q^* = [1, 0, 0, 0]$$

The magnitude of a quaternions is defined as :

$$|q| = \sqrt{z \circ z^*} = \sqrt{q_w^2 + q_x^2 + q_y^2 + q_z^2}$$

The magnitude of unit quaternions is always equal to 1 by definition.

6.1.1 Distance

Calculating the distance between two quaternions can be accomplished by exploiting the exponential map parameterization. The spherical distance of any point on the sphere is preserved under the exponential map. Therefore, the magnitude of the log mapping of the exponential is equivalent to the spherical distance.

$$d(q(1), q(2)) = |ln(q(1)^* \circ q(2))|$$

From the definition of the exponential map in the previous section, it is known that the magnitude of the log map is θ. Mathematically, we can calculate the exponential(log) mapping for a quaternion $q = e^{\frac{\theta}{2}\hat{n}}$ as follows :

$$\frac{\theta}{2} = arccos(q_w)$$

$$\hat{n} = \frac{(q_x, q_y, q_z)}{sin(\frac{\theta}{2})}$$

Performing quaternion multiplication between the conjugate of $q(1)$ and $q(2)$ yields a scalar element in the resulting quaternion that is equivalent to the norm of the dot product between two quaternions (see definition of quaternion

multiplication). Therefore we can simplify our distance function to :

$$d(q(1), q(2)) = 2 * arccos(|q(1) \bullet q(2)|)$$

Equivalently, we can define the euclidean distance function as the norm or magnitude between the difference of two quaternions.

$$d(q(1), q(2)) = |q(1) - q(2)| = \sqrt{(q(1) - q(2)) \circ (q(1) - q(2))^*}$$

6.1.2 Spherical Interpolation

Quaternions are the ideal parameterization of rotations for performing interpolation. In a vector space, linear interpolation between two components is performed with the familiar formula :

$$x = (1 - \alpha)x_0 + \alpha x_1$$

For very small angles, the linear formulation may suffice, if the norm of the unit quaternion is maintained. However, the spherical nature of quaternions requires a procedure that is not linear. Many spherical interpolation techniques exist, but Shoemaker introduced *Slerp* or spherical linear interpolation for the explicit purposes of quaternion interpolation for animating 3D graphics. Similar to linear interpolation, the Slerp algorithm refers to constant speed motion along a unit radius arc given the end points and an interpolation parameter $0 \leq \alpha \leq 1$.

The geometric formula for Slerp is :

$$Slerp(p_0, p_1; \alpha) = \frac{sin((1 - \alpha)\frac{\theta}{2})}{sin(\frac{\theta}{2})} p_0 + \frac{sin(\alpha\frac{\theta}{2})}{sin(\frac{\theta}{2})} p_1$$

where $\frac{\theta}{2}$ can be interpreted as the angle subtended by the arc between p_0 and p_1. Recall that the exponential map equation for a quaternion is $q = e^{\frac{\theta}{2}\hat{n}}$ and a quaternion exponentiated by a value α is $q^\alpha = e^{\alpha\frac{\theta}{2}\hat{n}} = cos(\alpha\frac{\theta}{2}) + \hat{n}sin(\alpha\frac{\theta}{2})$. The equivalent quaternion expression for Slerp is:

$$Slerp(q_0, q_1; \alpha) = q_0(q_0^* q_1)^\alpha$$

Raising the quaternion to a power α is accomplished by taking the exponential map of the quaternion product and then applying α with θ in the polar coordinate formula. The result is converted back to a quaternion and multiplied by the original starting quaternion.

The one disadvantage of the quaternion Slerp scheme is that one must check for antipodal symmetry of the unit quaternions before interpolating. Since, there are two equivalent representations for each quaternion, the user must check that the two quaternions that are to be interpolated are located on the same side of the sphere. Otherwise, the geodesic path would not be the shortest path.

Checking whether two quaternions are on the same side of a sphere involves a simple heuristic. Simply take the dot product of the two quaternions. If the result is negative, the two quaternions are in opposite hemispheres and one quaternion needs to be moved by taking the conjugate to the opposing hemisphere.

$$q_0 \bullet q_1 < 0$$

The interpolation scheme for 2 quaternions can be extended for multiple quaternions. However there is no standard procedure for accomplishing this task. Patrick Johnson wrote a thesis examining the different methods. A simplistic approach is to extend the premise of Slerp and use a vector in fixed tangent space (exponential map) as the basis for interpolating between quaternions. An alternative method that is less dependent on the single quaternion which will be the basis of interpolation, is to perform an iterative procedure for interpolation that loops until convergence.[37].

For learning purposes the iterative interpolation scheme is computationally intractable. Therefore, we will only discuss the Slime procedure for interpolating multiple quaternions. The Slime algorithm has the added advantage of also being capable of performing extrapolation. Since Slerp can be rewritten in the following format :

$$Slerp(q_0, q_1, \alpha) = q_0 e^{\alpha ln(q_0^* \circ q_1)},$$

we can expand the formula for the multiple quaternions in a vector Q with corresponding weights a_i and a reference quaternion p as follows :

$$slime(a, p, Q) = p e^{\sum_{i=1}^{N} a_i ln(p^* \circ Q_i)}$$

One disadvantage of the preceding formulation is that the choice of reference quaternion p can severely affect the results. Another concern is that since it relies on a fixed tangent space, a singularity is introduced. The singularity will occur orthogonal to the reference quaternion p. However, this translates to not being able to blend quaternion joints that are 360 degrees apart. The algorithm can not be used as a substitute for taking the average of random quaternions, but it is suitable for interpolation schema since no joint on the human body should be allowed to rotate 360 degrees. Finally suitable choices for the reference quaternion include the identity quaternion, one of the samples or the mean of the samples (to be defined in the next section).

6.2 Statistical Models

Learning information through the use of quaternions (or the exponential map representation), implies that we need means of representing the distribution of the data. A standard gaussian density, requires representations of both the mean and variance in a group of quaternions.

6.2.1 Quaternion Average

The expected value or mean of a function with euclidean vectors is the standard formula :

$$\bar{x} = \frac{1}{n}\sum^{n} x_i.$$

Attempting to use this formulation is plagued with problems. As with the interpolating procedure, combining spherical elements in a linear fashion does not yield an accurate result. Even if the result is renormalized to ensure a unit quaternion, the result would be inaccurate. In addition, the antipodal symmetry of quaternions means that the average is dependent on the representation of the same quaternion q or $-q$.

The notation for average rotation is well defined, but there are various definitions of mean. Moakher demonstrates that a mean rotation derived from the Frobenius inner product produces a rotation that is the closest to normal arithmetic mean or rotation matrices. Moakher defines a Euclidean mean rotation and a Riemannian mean rotation. The two rotations are based on the definition of a distance function between quaternions. The Euclidean distance formulation between 2 rotations is :

$$Dist_F(R_1, R_2) = ||R_1 - R_2||_F^2$$

where the F denotes the Frobius norm. Similarly the Riemannnian distance between two rotations is defined as

$$Dist_R(R_1, R_2) = \frac{1}{\sqrt{2}}||log(R_1^T R_2)||_F$$

The mean rotation R is then defined as the rotation that minimizes the sum of distances over all examples

$$argmin_{R\in SO(3)} \sum_{i=1}^{N} dist(R_i - R)$$

The Riemannian mean shares properties with the geometric mean.

The first algorithm computing the mean with quaternions is an iterative procedure using the Riemannian definition of distance [17]. Before proceeding, it is important to ensure that none of the points being averaged are antipodal to any other in the group. The algorithms can be modified for different convergence rates. Given a vector or weights, w, and a set of quaternions, Q, first initialize q as the sum of weights divided by the norm (to force the result back on the unit sphere).

$$q = \frac{\sum_{i=1}^{n} w_i Q_i}{||\sum_{i=1}^{n} w_i p_i||}$$

Then iterate until u becomes sufficiently small with the following two steps :

1.
$$u = \sum_{i=1}^{n} w_i (ln(Q_i) - ln(q))$$

2.
$$q = e^{ln(q)+u}$$

The main shortcoming of this method is the computational cost and time to iterate through the algorithm. If we are interested solely in the mean value of the quaternion and not a random average, the mean has a closed form solution to a maximization equation. Minimization of the Euclidean distance function as a sum over the elements contains non-linear elements, which makes analytic minimization difficult. This is equivalent to using the distance function defined earlier :

$$dist(q_1, q_2) = ||ln(q_1^* \circ q_2)|| = 2arccos(|q_1 \bullet q_2|)$$

$$q = argmin_P \sum_{i=1}^{N} dist(P, Q_i)^2$$

Simplifying this solution involves minimizing over $1 - cos(\theta)$ instead of the physical distance between elements θ (θ is the result of the logmap operation on the quaternion product). The procedure can be simplified further by

maximizing over $cos(\theta)$ which is the equivalent of minimizing over $1 - cos(\theta)$.

However, $cos(\theta)$ is simply the dot product of the two quaternions. So our final optimization problem is reduced to :

$$q = argmax_{P \in S(3)} \sum_{i=1}^{N} (P \bullet Q_i)^2$$

In addition, we want to constrain the quaternion solution to be of unit magnitude. Therefore, we add a Lagrange multiplier to the equation.

$$E(P) = \sum_{i=1}^{N} (P \bullet Q_i)^2 + \lambda((P \bullet P)^2 - 1)$$

Through a series of operations this reduces to :

$$E(p) = p^T Q Q^T p + \lambda(p^T p - 1)$$

Setting the derivative to zero, we arrive at,

$$\frac{dE(p)}{dp} = 2QQ^T p + 2\lambda p = 0$$

This problem is the eigenvector problem $Ax = \lambda x$. Therefore, we have a closed form solution for the quaternion mean [37]. The procedure is simply to multiply the data matrix of quaternions Q by its transpose, $S = QQ^T$. Then find the eigenvector of S that has the largest eigenvalue. Since it is a quaternion representation there are two antipodal solutions.

Both methods of solving for an average quaternion were utilized in determining the mean value of a cluster. In cases with minimal variation within a cluster of quaternions, the two solutions yield identical results. However, in situations of quaternions varying over a large range of motions, the results from the two methods tended to vary. After testing and examining results, it was decided to use the closed form solution simply because of the increased efficiency of the method. Figure 6.6 demonstrates the results of running the latter algorithm on a group of quaternion data. The green vector represents

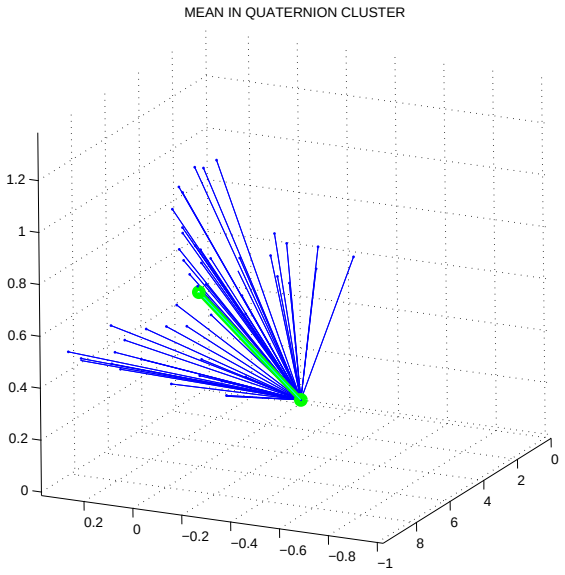

Figure 6.6: The Mean Quaternion (green) in a Cluster of Thorax Quaternions

the mean quaternion which does appear to be the central angle of the cluster.

6.2.2 Quaternion Gaussian Density

Here, Patrick Johnson's parameterization of a quaternion density for learning purposes is used. Recall that the normal euclidean density function in N dimensions is written as :

$$N(x, m, C) = \frac{1}{(2\pi)^{\frac{n}{2}}|C|^{\frac{1}{2}}} e^{\frac{1}{2}(x-m)^T C^{-1}(x-m)}$$

where m is the mean and C is the covariance matrix. When modified for quaternions, the quantity $(x - m)$ is replaced with the log map representation of the vector q rotated to the position of the mean. Therefore, we obtain $ln(M^* \circ Q)$. Johnson defines the quaternion covariance as:

$$C = R^T V R$$

where R is the rotation matrix associated with a quaternion r and V is a diagonal matrix v. The final density function is written as :

$$p(q, m, r, V, \rho) = ce^{-\frac{1}{2}(ln(R^* \circ M^* \circ Q \circ R))^* V^{-1} (ln(R^* \circ M^* \circ Q \circ R))}$$

The extra parameter ρ is used to force the density function to zero if the value strays outside of a determined range. The density equation for the gaussian reduces to the exponent of the squared mahalanobis distance of a vector q from the cluster of quaternions defined by mean m and variance v. The value of ρ is defined to be the maximum mahalanobis distance of any quaternion in the cluster to the mean:

$$\rho = max_i(dist_{mahalanobis}(Q_i, m))$$

where the mahalanobis distance is :

$$d_{mahalanobis} = ||V^{\frac{1}{2}} ln(R^* \circ M^* \circ Q \circ R)||.$$

Finally, the variances need to be obtained from a cluster of quaternions. The first step is to arrange all the data in a cluster so that it is in the same hemisphere as the mean (use the heuristic presented before of taking the dot product and checking whether it is positive). Take the tangent of all the data in the cluster that has been aligned with the mean :

$$w_i = ln(M^* \circ Q_i)$$

Obtain the matrix K from the data matrix of the w_i, W :

$$K = \frac{1}{N-1} W W^T$$

Perform an eigenvalue decomposition on K (use the svd).

$$u \cdot s \cdot v^T = svd(K).$$

Apply the exponential map function to convert the vector values of U into quaternions and store as the matrix R. Store the diagonal of s as the eigenvalues in the matrix V. If the matrix V is singular, it implies that one or more of the eigenvalues is zero. Therefore the inverse can not be calculated;

however, this is revealing the true, underlying degrees of freedom of the joint. Instead of the quaternion model, we can use Euler angles to constrain the joint to the proper degrees of freedom.

6.3 Angular Velocity

In both dynamic simulations and learning applications, the position and pose of the skeleton are not sufficient. The linear and angular velocity must be included to form a coherent model. In our situation, angular velocity is a 3-dimensional vector or 3×3 matrix in the form of a skew adjoint linear transformation that encodes the rate of rotation of a given joint. It is defined by the derivative of orientation with respect to time.

In particular, if the matrix $A(t)$ is a SO(3) or special orthogonal linear transformation in 3 dimensions that describes the orientation of a joint, then the angular velocity is defined as follows :

$$\omega = A(t)^{-1} \frac{d}{dt} A(t)$$

The magnitude of the angular velocity is the *angular speed* or rate of rotation around the given axis. The total velocity for a point on the skeleton which is moving with the same translational velocity as the rest of the body and rotating independently within the skeleton is :

$$v = v_t + \omega \times (r - r_c)$$

where v is the total velocity of the particle, v_t is the translational velocity, r is the position of the particle and r_c is the position of the center of the body.

Two coordinate frames are used in rigid body dynamics. One axis is fixed in space (the world view) and the other is associated with the principle axes of the rotating body. Using Euler angles in the 'x-convention' (note that the x-convention is a XYZ rotation where a body is first rotated around the z

axis, then the y axis and finally the x axis with corresponding angles $[\phi, \theta, \psi]$), the angular velocity ω can be calculated in the body axes as follows :

$$\begin{pmatrix} w_x \\ w_y \\ w_z \end{pmatrix} = \begin{pmatrix} sin(\theta)sin(\psi) & cos(\psi) & 0 \\ sin(\theta)cos(\psi) & -sin(\psi) & 0 \\ cos(\theta) & 0 & 1 \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \tag{6.1}$$

However, a major disadvantage of using Euler Angles to calculate the angular velocity is that the matrix in this equation is singular when $sin(\theta) = 0$.

Quaternions present an elegant solution and avoid the numerical problems encountered when using Euler angles. If $\dot{q}$ is the derivative of a quaternion then the instantaneous angular velocity can be mapped into the tangent space with the following relationship :

$$\begin{pmatrix} \dot{q_w} \\ \dot{q_x} \\ \dot{q_y} \\ \dot{q_z} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega z \end{pmatrix} \circ \begin{pmatrix} q_w \\ q_x \\ q_y \\ q_z \end{pmatrix}$$

where the multiplication indicated is quaternion multiplication.

Allow q_ω to be the quaternion representation of the angular velocity by adding a 0 to the w portion of the quaternion vector to yield a total of 4 dimensions. Using this notation, we can write the formula to explicitly update quaternion vectors in time, based on the angular velocity. If t represents increments in time, the equation is :

$$q(t) = e^{\frac{1}{2}q_\omega \cdot t} q(0) = \begin{pmatrix} cos(\frac{1}{2}||\overrightarrow{\omega}||t) \\ sin(\frac{1}{2}||\overrightarrow{\omega}||t)\frac{\overrightarrow{\omega}}{||\overrightarrow{\omega}||} \end{pmatrix} \circ q(0)$$

In calculating future time steps, the absence of a parameter for angular acceleration implies that $\alpha = 0$ and therefore, the body demonstrates constant velocity. Such an assumption may be incorrect, but still yields a feasible approximation to the correct values.

Calculating the angular velocity given two quaternions and a time step of n is straightforward. First consider the difference between two quaternions:

$$q_1 = q_d \circ q_0$$

$$q_d = q_1 \circ q_0^*$$

Now suppose we view a quaternion at time step t as a fixed unit quaternion, say q_d, exponentiated by time, so:

$$q(t) = e^{t*ln(q_d)}.$$

Note that the log of a quaternion is the same as multiplying the angle and axis, in the angle axis representation of a rotation, together.

$$ln(q) = \frac{\theta}{2}\hat{v}$$

Using the exponentiation of time representation, the derivative of the quaternion is as follows :

$$\frac{d}{dt}q(t) = q(t) \circ ln(q_d)$$

We can then solve for the angular acceleration

$$\omega = 2 * \dot{q}(t) \circ q(t)^*$$

$$\begin{pmatrix} 0 \\ \omega \end{pmatrix} = 2 * \begin{pmatrix} q_w(t) \\ \hat{q} \end{pmatrix} \circ \begin{pmatrix} 0 \\ \frac{\theta}{2}\hat{v} \end{pmatrix} \circ q(t)^*$$

$$\omega = \theta\hat{v}$$

6.3.1 Angular Acceleration

In dynamic simulations and learning procedures, the angular acceleration is a necessary component in addition to the angular velocity. Recall that for the linear translation of a particle $a = \frac{dv}{dt} = \frac{d^2(p)}{d^2t}$. Similarly, in the realm of rotations, the angular acceleration is the change in rate of angular velocity. The angular accelerations are directly related to the second derivatives of the quaternion in time. The angular acceleration is found by differentiating the expressions for ω.

If we extend the angular velocity to form a quaternion

$$\omega = \begin{pmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}$$

$$\dot{q} = \frac{1}{2}\omega \circ q$$

$$\omega = 2\dot{q} \circ q^*$$

Differentiate the expression for ω with respect to time :

$$\dot{\omega} = 2\left(\ddot{q} \circ q^* + \dot{q} \circ \dot{q}^*\right)$$

$$\begin{pmatrix} 0 \\ \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{pmatrix} = 2\left(\begin{pmatrix} \ddot{q}_w \\ \ddot{q}_x \\ \ddot{q}_y \\ \ddot{q}_z \end{pmatrix} \circ \begin{pmatrix} q_w \\ -q_x \\ -q_y \\ -q_z \end{pmatrix} + \begin{pmatrix} |\dot{q}|^2 \\ 0 \\ 0 \\ 0 \end{pmatrix}\right)$$

In the angular velocity section, two coordinate frames are used in rigid body dynamics. In the space-fixed coordinate frame, *torque* is the time derivative of *angular momentum*. Similarly, force is the time derivative of linear momentum. Note that the angular momentum of a rigid body is the product of the moment of inertia and the angular velocity. Torque is the force that produces rotational motion. Therefore :

$$\tau = \frac{dL}{dt} = I\frac{d\omega}{dt} = I\dot{\omega} = I\alpha$$

However, the relationship between the time derivative in the space and body frames is as follows:

$$\left(\frac{dL}{dt}\right)_{space} = \left(\frac{dL}{dt}\right)_{body} + \omega \times L$$

Therefore, we can rewrite the equation for torque in the body frame by substituting into the above equation.

$$\frac{dL}{dt} = N$$
$$\dot{L}_x + \omega_y L_z - \omega_z L_y = N_x$$
$$\dot{L}_y + \omega_z L_x - \omega_x L_z = N_y$$
$$\dot{L}_z + \omega_x L_y - \omega_y L_x = N_z$$

Using as the principal axes the axes around x, $L_x = I_x\omega_x$, the Euler equations of a rigid body rotation can be derived.

$$I_x\dot{\omega}_x = N_x + (I_y - I_z)\omega_y\omega_z$$

$$I_y\dot{\omega}_y = N_y + (I_z - I_x)\omega_z\omega_x$$
$$I_z\dot{\omega}_z = N_z + (I_x - I_y)\omega_x\omega_y$$

Rotations expressed in quaternions over even time steps can be evaluated using our calculated values for angular velocity and angular acceleration.

$$q(t) \approx e^{t\omega + \frac{1}{2}t^2\dot{\omega}} \circ q(0)$$

However, the accuracy of this equation is dependent upon the angular acceleration having a constant value. Without this assumption, the equations of motion are far more complex and involve solving differential equations. Therefore, the above equations only present an estimate of the quaternion values in time.

Similar to the methods used for angular velocity, we can approximate values for the second derivatives of the time series data and hence the angular acceleration. These approximations derive the *average* angular velocity and angular acceleration and not instantaneous values. As such, our equation for updating future time steps, even with the assumption of constant angular velocity is only a rough estimate. In application where we are tracking over time, we use the calculated values from past time steps as an estimate of the model and adjust with values found for current input parameters.

Given quaternions q_0, q_1, q_2, at time steps $t = 0, 1, 2$ we can approximate the average angular velocity at time $t = 1$ and $t = 2$ as shown in the previous section.

$$\omega(1) = 2 * ln(q_1 \circ q_0^*)$$
$$\omega(2) = 2 * ln(q_2 \circ q_1^*)$$

Therefore we can approximate the angular velocity as the difference in angular accelerations divided by the difference in time :

$$\dot{\omega}(2) = \omega(2) - \omega(1) = 2 * (ln(q_2 \circ q_1^*) - ln(q_1 \circ q_0^*))$$
$$= 2 * ln(q_2 \circ q_1^* \circ q_0 \circ q_1^*)$$

CHAPTER 7
LABAN PERSONALITY

Chris Bregler identified the lack of expressive control in motion capture based animation and turned to the dance community for inspiration. In a collaboration with an expert in the field, Peggy Hackney, Bregler proposed using notation from dance as the basis of a filter to process motion capture data. He envisioned being able to twist knobs to continuously add or subtract certain 'style'-based parameters.

As mentioned in the background sections, exaggerating or minimizing aspects of motion are well known techniques in the realm of 2D animation. A common phenomenon is that video 'flattens' a performance. For example, a dynamic and energetic live dance performance will often appear to have reduced intensity when recorded and viewed on a screen. It was hypothesized that motion capture produces a similar effect. The proposed system would 'boost' all the efforts in motion capture data so that perceptually it simulates the effects of watching a live person move.

7.1 Laban Notation

Developed by Rudolf Laban, Laban notation is a means of decomposing and classifying movement into core components. Used extensively by choreographers, Laban notation enables the written documentation of a movement. The Laban theory contains a substantial subsection devoted specifically to the expressive qualities of movement. Entitled *Effort Theory*, these concepts

are based on the inner impulse of a movement or, rather, where a movement originates from. The theory specifies that every motion has the potential to include effort parameters or factors *Space*, *Weight*, *Time*, and *Flow* [53].

Laban suggested that the Effort factors resulted from bipolar interactions, either yielding or resisting. Therefore each effort factor has two opposing poles. As a historical note, Laban first developed his theories after studying the repetitive movements of workers in factories. His teachings emphasized practicing reciprocal movements that balanced the stress put upon the body by performing narrow repeated tasks.

The four effort parameters and their bipolar elements are described below.

SPACE

1. **Indirect** Defined by movements that are multi-focused and encompassing. Movements that fall in the indirect portion of the weight space show a tendency toward a combination of twisting, bending-extending actions in several parts of the body. As such, the movements in the body appear more successive and sequential.

2. **Direct** Defined by movements that are narrowed and focused. Direct movements align the joints and the flow of movements tends to be simultaneous.

WEIGHT

1. **Light** The Light pole is defined by a delicate or sensitive touch. As such the movements demonstrate decreased muscular tension and often

engage the full extension of the upper body. In space, the movements are often directed upward.

2. **Strong** Defined by firm and forceful movements that are resisting the pull of gravity. Strong actions show a tendency towards muscular tension and are often focused in the lower body for support. As such the movements are often downward directed.

TIME

1. **Sustained** Defined by calm and prolonged actions. The movements in this state are large, total body movements that tend to move forward and outward.

2. **Sudden** Defined by immediate and unexpected actions. The movements are often isolated gestures and appear visually backward and inward.

FLOW

1. **Free** Defined by continuity of movement; the motions are fluent. The movements tend to initiate from the torso and spread in successive flow out to the extremities.

2. **Bound** Defined by controlled or restrained movements. Movements start from the extremities toward the center of the body.

The combinations of effort parameters create new factors. The combination of factors often create certain moods. While it is rare to see a person perform a movement and demonstrate only one of the four effort qualities, only

in the most extreme and dire circumstance will all four efforts be observed (at some pole) in the same action. Individuals tend to have effort (or expressive) personalities that tend to combine a certain 2 or 3 effort factors across all movements.

Given the continuous 4 dimensional nature of this decomposition of the expressive factors of motion, the effort space is the ideal candidate for parameterizing as a filter for animation. The proposed course of action was to gather motion capture data of a variety of movements and have LMA (Laban) experts annotate the data with the correct effort parameters from the video and motion capture performance. Every movement sequence would be assigned a numerical value for each of the four factors, where 0 corresponds to the factor not being exhibited at all. The space would then be learned and decomposed. New movements could be modified by 'amplifying' the knobs on certain Laban parameters.

Annotating the motion and subdividing according to the effort parameters is straightforward using Laban notation. The notation subdivides the movement in a vertical format into individual phrases that clearly list each action of the various parts of the body. The effort parameters are included in the side of each of these phrases. The documentation of a short phrase is demonstrated in 7.7.

The idea of applying Laban theory to animation is not novel. As discussed in the survey section Chi et al. introduced Laban parameters as a tool for creating emotional effect in their 2000 paper on 'EMOTE' [21]. However, this work focused on procedural techniques to produce the desired effect. While the results were not impressive, the paper contained key insights on how joint movements effected the visual impact of a motion. The approach

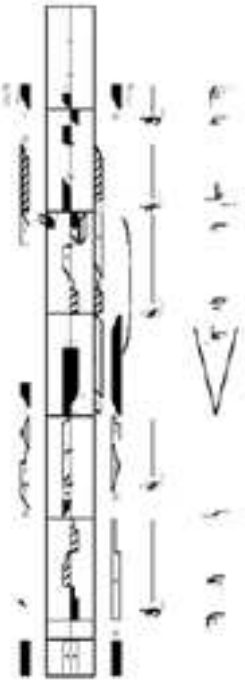

Figure 7.7: Laban Notation for Neutral Phrase

Figure 7.8: Dancer Performs Neutral Phrase to be Motion Captured

proposed by Bregler differed in using the actual motion capture data as the means of acquiring information about the space and as a basis for reanimating in exaggerated form.

7.2 Neutral Phrase

After several initial experiments, the team under Bregler found it difficult to isolate and learn parameters from random movements demonstrating a variety of Laban effort factors. Instead, Peggy Hackney developed a short *Neutral Phrase* that combined isolated instances of each of the core effort parameters in each extreme. The phrase was performed and motion captured using a variety of dancers versed in Laban theories and other individuals in many sessions over several months.

The motion captured performances of the Neutral Phrase were cleaned, edited and fit to kinematic chains. For several of the sessions, a panel of 3 LMA experts documented the expressive qualities (effort factors) in real time. In addition to motion capture records, each session was documented on video tape. The LMA experts cross referenced their annotations and checked the video footage for confirmation.

The Neutral Phrase was characterized by strong isolated arm movements. It was theorized early in the process, that effort parameters could be different

Figure 7.9: Animation of 'Neutral Phrase' demonstrating extremes of LABAN Effort Notation

in various limbs of the body. A consistent study of one aspect of the body could lead to cohesive results.

In Fig. 7.9 is a diagram of the Neutral Phrase after being motion captured, fit to a kinematic chain and then reanimated on a stick skeleton.

To gain the full spectrum of Laban effort factors, the Neutral Phrase was performed with neutral effort, low intensity effort and high intensity effort by each performer. The intention was to create a basis for a numerical range to scale each effort parameter.

Unfortunately, after much time and effort was spent conducting the Neutral Phrase experiments for the database, there was little variation in the effort parameters by the performers when they did the actual phrase. For example, the majority of the phrase was often performed with the combination

of 'strong' and 'bound' combination of the Weight and Flow motion factors. Likewise there were few examples of light and sudden.

As a result, the calculations were focused on differentiating between high and low intensity. An emphasis was placed on mapping these results back on to the motion skeleton.

7.3 Frequency Decomposition

Application of signal processing techniques to analyze motion data was introduced early in the field of motion capture research. In 1997, Bruderlin and Williams attempted to modify motion by altering the contribution of each of the frequency bands for the laplacian pyramid and reconstructing the signal [15].

An initial theory was hypothesized that the high frequencies of the motion data revealed information about the high intensity application of Laban parameters.

In order to compare and modify the signals accordingly, a database of motions for both high and low intensity movements was decomposed into kinematic chains of angles represented in the exponential or log map format. While efforts were made to preserve timing across performances of the Neutral Phrase during recording, obvious discrepancies occurred. Therefore the data had to be aligned. Of course, it was theorized that timing may be a crucial difference between high and low intensity performances, but this was temporarily ignored in the initial experiments.

The data phrases were aligned with the technique of dynamic time warping. Dynamic time warping finds a path such that frames are substituted or deleted so that two motion segments of similar motions are aligned in time

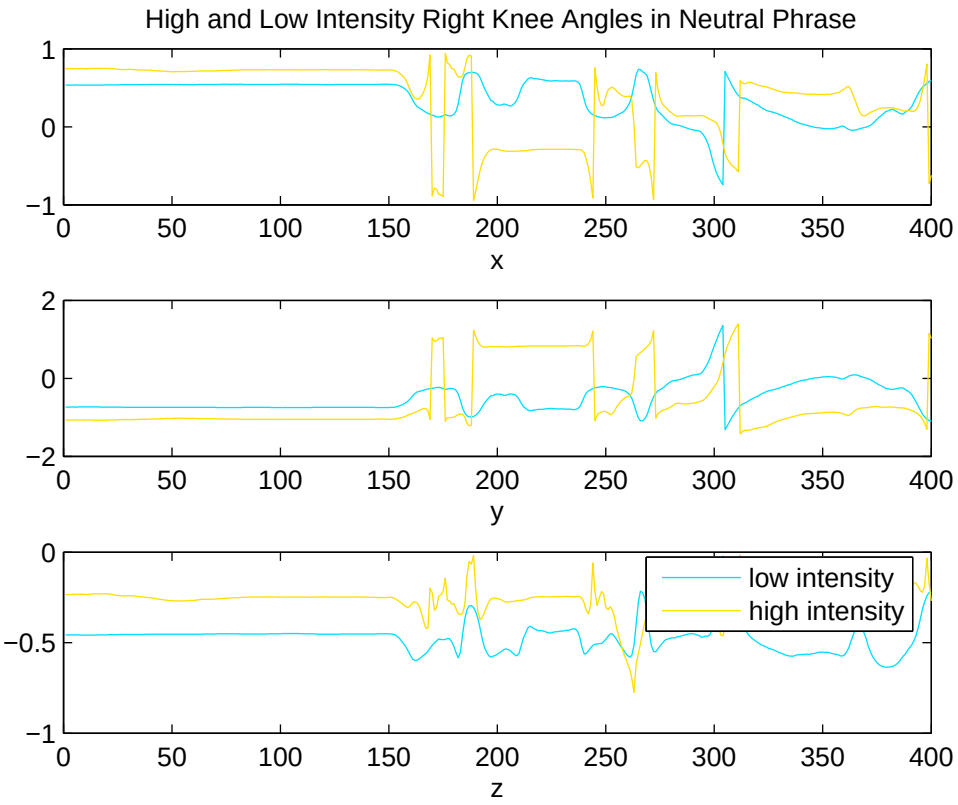

Figure 7.10: Comparison of Euler Angles for Knee Movement in Neutral Phrase at High and Low Intensity

[3]. The time warp algorithm finds the correspondence between time values in sequences of times series such that the overall distance is minimized. For two series a and b, with time points $t_0, ...t_n$ and $u_0, ..u_m$, consider two paths $i(h)$ and $j(h)$ for time points $h = 0, 1...p$, then $i(0) = 0$, $i(p) = n$ and the values in between are $i(h+1) =$ either $i(h) + 1$ or $i(h)$.

The distance D(a, b) is

$$D(a, b) = \sum w(h) d(a_{i(h)}, b_{j(h)})$$

where the weight function is defined as,

$$w(h) = \frac{1}{2}(t_{i(h)} - t_{i(h-1)} + u_{j(h)} - u_{j(h-1)})$$

Two motion performances of the Neutral Phrase that were annotated for demonstrating noticeable variation between Laban effort parameters between 'high' and 'low' intensity were time warped to find a path or frames that aligned the two motions. A frequency decomposition using the laplacian pyramid was performed. Recall from the survey section that the number of frequency bands for each signal is determined by the number of frames, where the number of bands is $fb = n$ and $2^n \leq m \leq 2^{n+1}$ and m is the number of frames. For the Neutral Phrase, 7 bands were derived. The decomposition was performed by successive convolutions of the signal with a filter kernel.

Figures 7.11 and 7.12 demonstrate the decomposition of the High and Low performances of the Neutral Phrase for the right knee. Observe the obvious differences between the two segmentations.

The formulation of the Neutral Phrase places particular emphasis on the segmentation of the movement of the arms. While leg movement is invaluable in the study of cyclic movement, it became clear that the key to segmenting the Laban parameters in the Neutral Phrase was to restrict the investigation to the arms. In Fig. 7.13 the frequency decomposition of the initial 'strong'

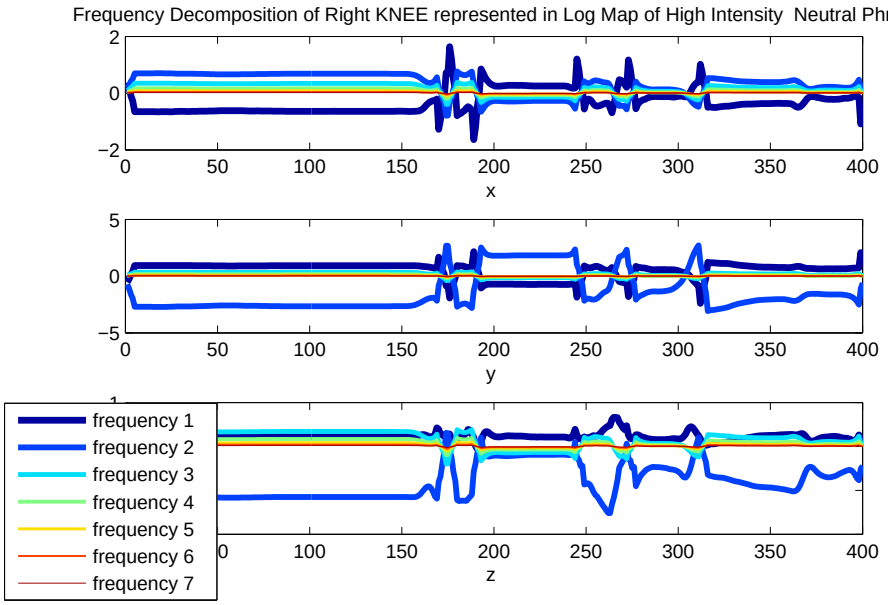

Figure 7.11: Frequency Band Decomposition of Knee Joint for Average High Intensity Performance of Neutral Phrase

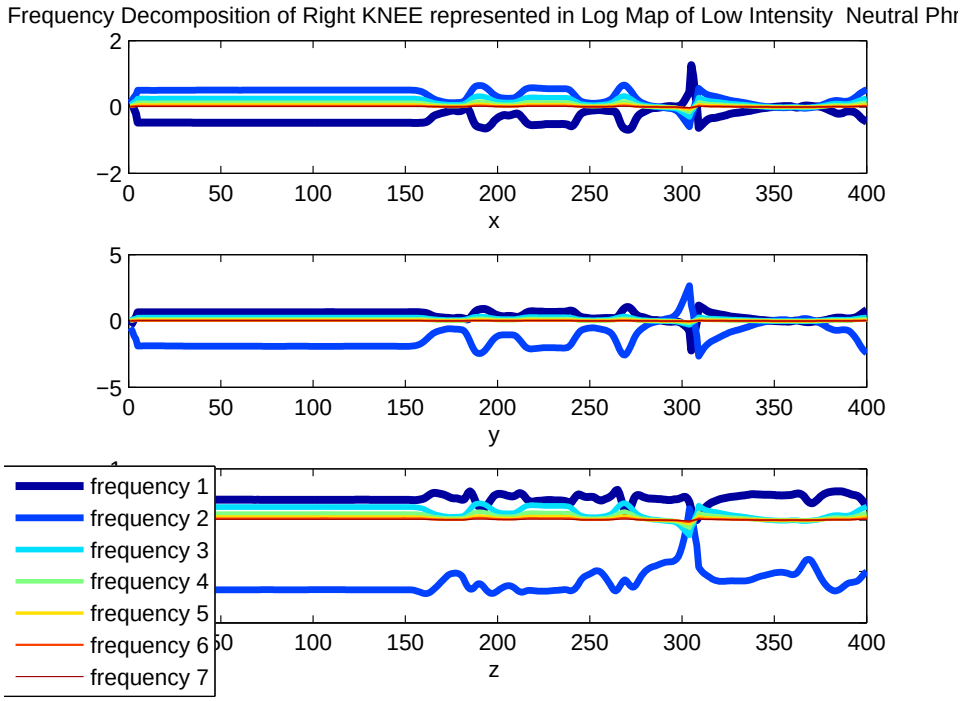

Figure 7.12: Frequency Band Decomposition of Knee Joint for Average Low Intensity Performance of Neutral Phrase

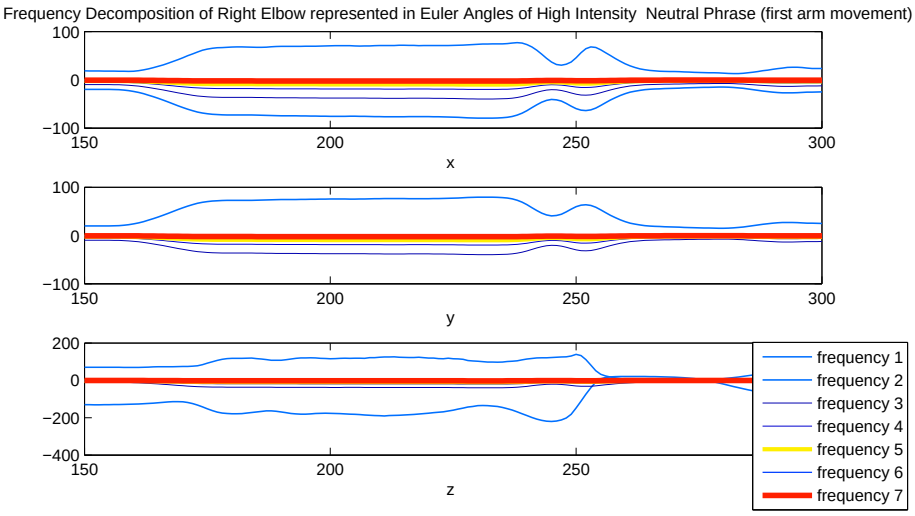

Figure 7.13: Right Elbow Frequency Bands

and 'bound' movement of the right arm in the Neutral Phrase performed at 'high intensity' is shown.

Since it was hypothesized that the concept of 'style' was embedded in the high frequency bands of motion, the author removed the high frequency bands of the high intensity performance angles and compared the reconstructed signal with the low intensity Neutral Phrase performance. A graph of the knee joints is shown in Fig. 7.14. Similarly, we tried mapping the high frequency signals from the high intensity performance onto the low intensity performance . The reconstructed motion signal was of the form

$$G_0 = G_{fb} + \sum_{k=0}^{fb-1} \delta L_k^{low} + (1-\delta) L_k^{high}$$

where $\delta = 1$ if we are including the low intensity signal at frequency band k and zero otherwise.

The results of these experiments was mixed. Simply removing the high frequencies obviously dampened the high intensity signal, but did not produce

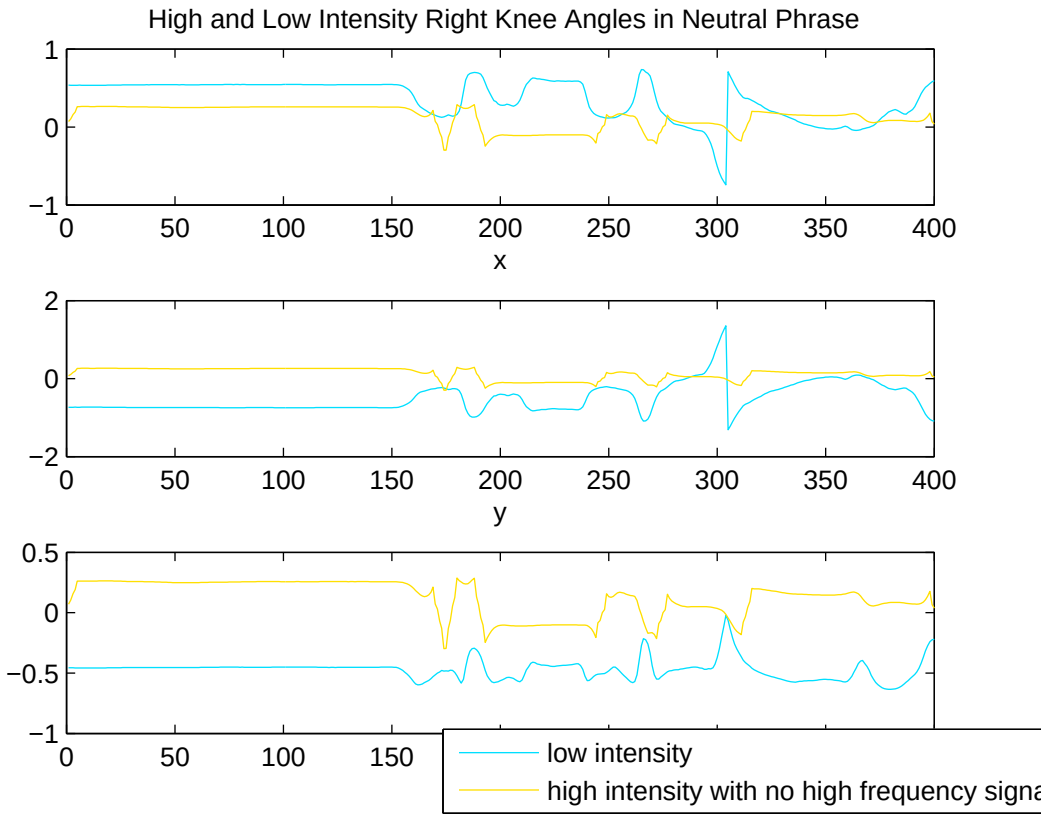

Figure 7.14: Removing high frequency bands from High Intensity Knee Angle

a result similar to the low intensity signal. Likewise, mapping the high frequencies of the high intensity angles on the low intensity performance yielded both animations and plots that were similar to the high intensity performance, but still missing key aspects.

In addition, simply amplifying or diminishing the contributions of the high frequency signals did not yield an obvious correlation between the extremes of Laban parameters. On the other hand, the visual impact of altering high frequency signals suggested that this approach may be a valid means of approximating or exaggerating Laban attributes.

After these experiments, the next natural step in the frequency decomposition approach was to learn coefficients that will map the frequency bands from the high intensity performance to low intensity. For this we wanted to concentrate solely on the movements of the right arm.

The problem with dynamic time warping is in the situations where there are ballistic type movements – for example, someone jumps into the air. It is not recommended to stretch out consecutive frames in these instances over time. The results can produce an artificial 'floating' effect. In addition, we had hoped to study the effect on timing between Laban parameters.

In attempts to find an 'average' high and low intensity signal throughout the varied performances by the dancers, standard signal processing techniques were applied. The signals were normalized in time. This was accomplished by clipping the phrases by the points of initiation and conclusion of movement (determined by zero derivative for angular velocity). Each phrase was decomposed into the first 7 Fourier transform harmonics (same as a laplacian pyramid) and reconstructed with a universal time of 1000 frames.

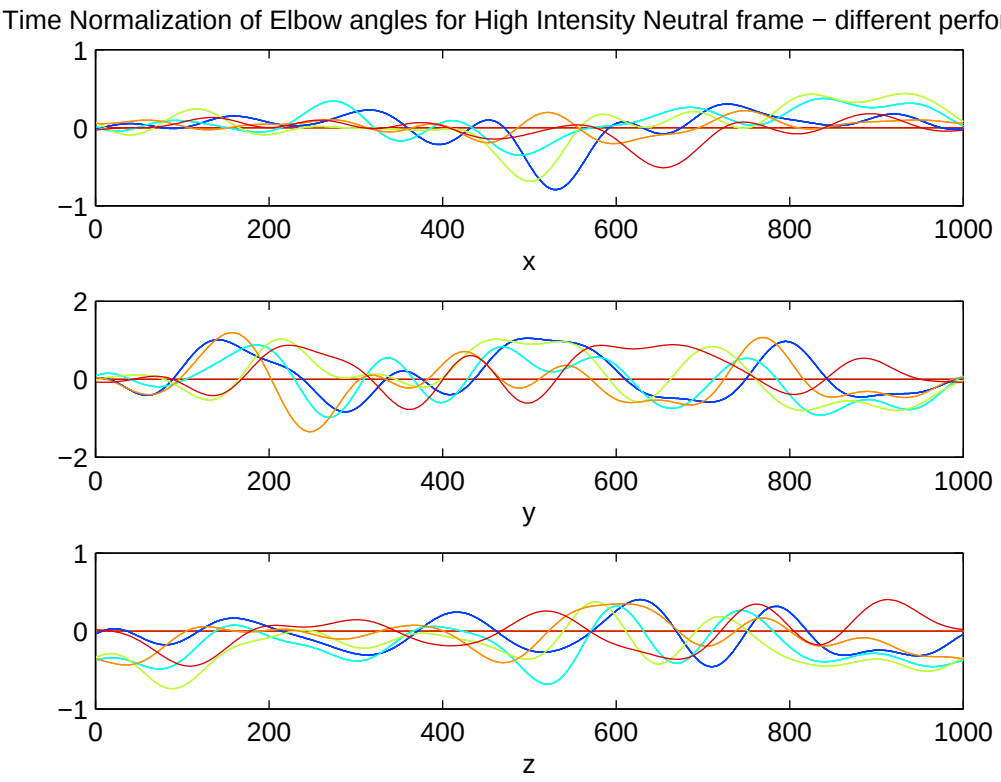

Figure 7.15: Time Normalized results for Elbow angles in High Intensity Neutral Phrase

A cohesive baseline signal was not readily apparent from a plot of normalized signals from a variety of performers as shown in Fig. 7.16.

However to determine the fundamental difference between the low and high intensity movements, the average time normalized signals for all dancers in the low and high intensity categories were analyzed. The plot shown here, Fig. 7.16, demonstrates the changes in Elbow angle. Since we were examining the average across performances, it is not surprising that the signals were dampened and the high frequency data compromised. However, we thought any clear differences in timing of the underlying signal or intensity amplitude

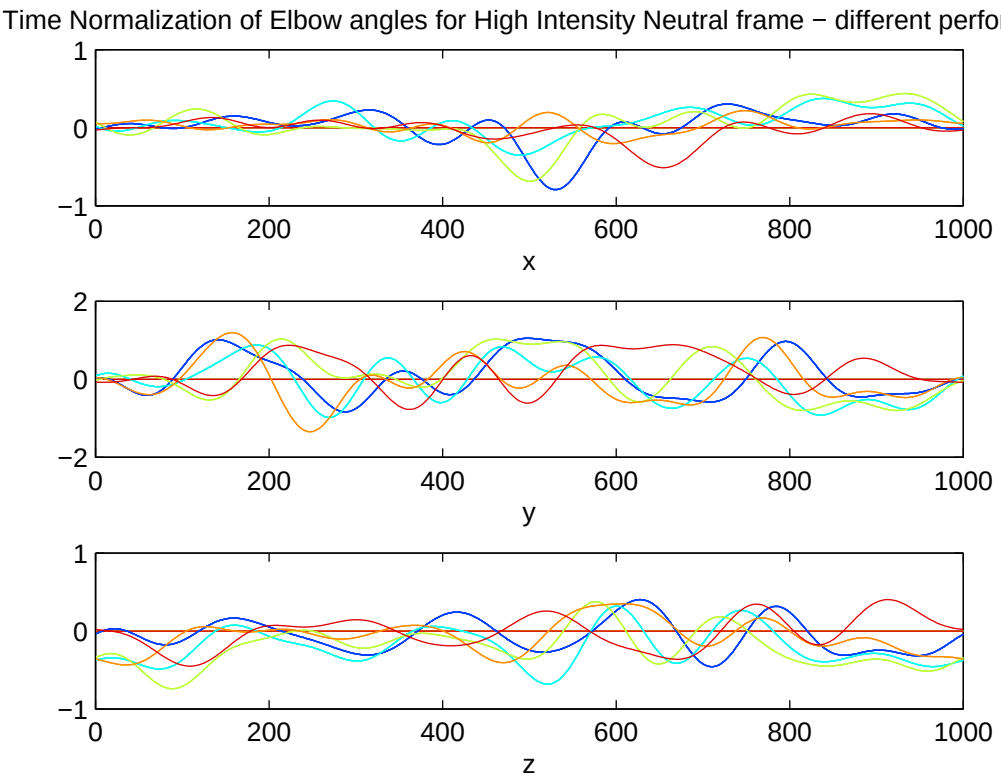

Figure 7.16: Time Normalized results for Elbow angles in High Intensity Neutral Phrase

might become apparent. The results were surprisingly very similar for both the low and high group, most likely because the content of the two phrases was extremely similar.

Another difficulty in performing this analysis was that the actual 'intensity' between *low* and *high* varied tremendously between performers. When examining a single joint between performers (on varied performance of both high and low), it was possible to perform least squared optimization to find a set of coefficients that would map the frequency bands of the high to the low intensity. However, this mapping was not universal among performers. In

addition, the mapping was different for different joints in the body.

7.4 Analysis of Intensity

Faced with difficulties in the actual data, the next step was to seek a clear means of categorizing the low and high intensity performances. By limiting the analysis to just the arm movements, we were able to linearly separate the high and low intensity data across performers.

The metric used was comparing the squared rate of change of angles in the entire arm (in quaternions). Assuming all rotations are in a kinematic chain and represented as local angles off of the parent and using the Euclidean distance metric for quaternions as defined in the chapter "Learning in Quaternions", the formulation reduces to a $3 \times n$ vector for each frame t

$$Dist_m(t) = [|R_{clavicle}(t) - R_{humerus}(t)|^2, |R_{humerus}(t) - R_{Radius}(t)|^2, |R_{radius}(t) - R_{hand}(t)|^2]$$

For the purposes of segmenting, we also include the metric for the previous time frame $t-1$; this parameter demonstrates rate of change. Therefore each frame has a learning vector of 6 components.

7.4.1 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA), which is related to the Fischer Discriminant, is a means of determining the linear combination of features which best separate two or more classes or objects. LDA is closely related to regression and Principle Component Analysis. In the two class case, which is used for determining Low versus High Intensity, the classification problem is to obtain a predictor for the class y of any sample of the same distribution given only an observation x. LDA assumes that probability density functions $p(\hat{x}|y = 1)$ and $p(\hat{x}|y = 0)$ are both gaussian distributions with identical full

rank covariance matrices $\sum_{y=0} = \sum_{y=1} = \sum$.

The distribution $p(y|\hat{x}) = \frac{p(\hat{x}|y)*p(y)}{p(\hat{x})}$ is dependent on the dot product $\hat{w} \bullet \hat{x}$ where

$$\hat{w} = \sum^{-1}(\hat{\mu}_1 - \hat{\mu}_0)$$

Fisher's linear discriminant, which is used interchangeably with LDA, does not use the assumption that the classes are normally distributed or have equivalent covariances. Instead, Fisher separates the two distributions by the ratio of the variance between the classes to the variance within the classes:

$$S = \frac{\sigma^2_{between}}{\sigma^2_{within}} = \frac{(\hat{w} \bullet (\hat{\mu}_{y=1} - \hat{\mu}_{y=0}))^2}{\hat{w}^T(\sum_{y=0} + \sum_{y=1})\hat{w}}$$

The intuitive definition of this measure is the *signal to noise* ratio.

Both forms of analysis were applied to the database of high and low frequencies. However, only the metric above successfully separated the data in both cases. In the graphs in Figures 7.17 and 7.18, the projected values of the discriminant are plotted across frames for the initial arm movement in the Neutral Phrase. During periods of action, that were annotated as 'Strong' and 'Bound' by Laban annotators in the 'high intensity'; the classes are clearly separated. As the movement lulls to more normal and static phases, the classification, as expected, converges.

7.4.2 LLE

The ability to classify limited portions of the body was encouraging. Local Linear Embedding (LLE) and LDA were utilized to study the nature of the Neutral Phrase space and seek a separation between classes.

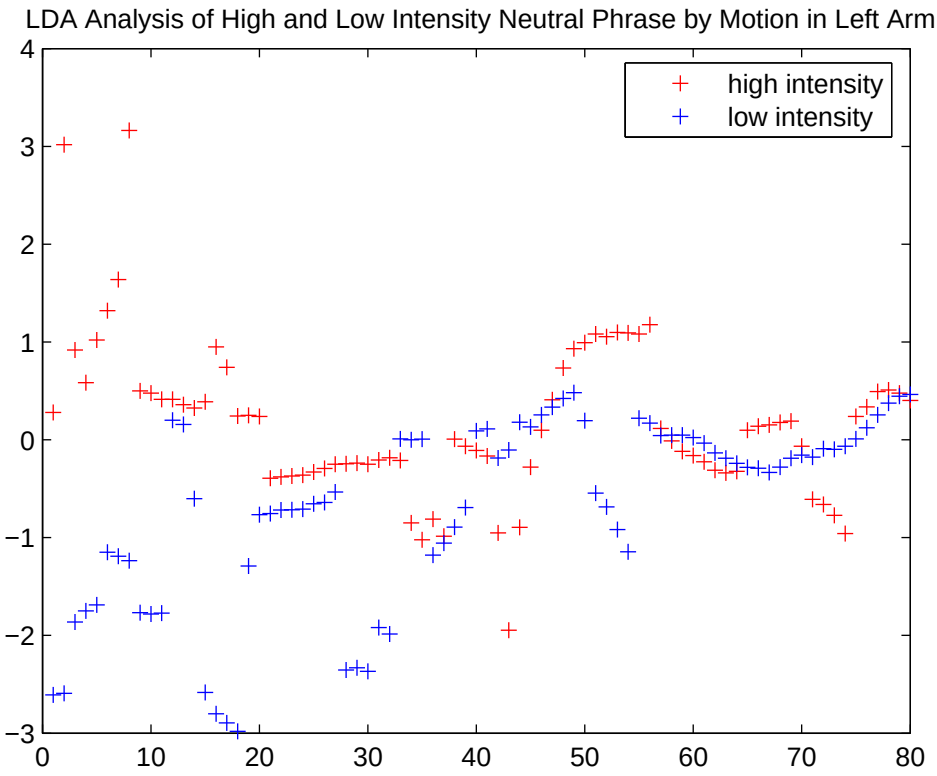

Figure 7.17: Linear Discriminant Analysis Separating High and Low Phrase Data by examining sequence of Movements in the Arm

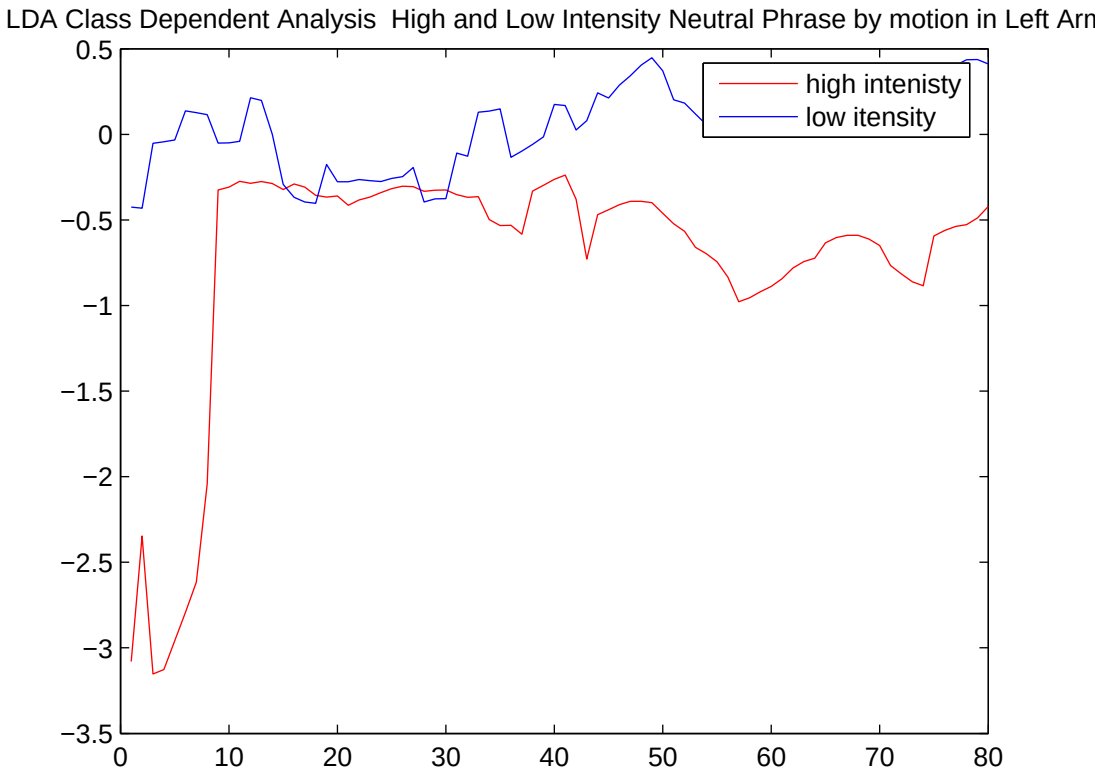

Figure 7.18: Class Dependent Analysis

7.5 Extrapolating for Emotional Effect

Among motion capture research investigations performed recently, the most visually pleasing manipulations of data tended to rely on the actual output of the motion capture system as a basis for animation. Separating the classes was an important step, but the aim of the project was exaggerating the Laban parameters in motion data.

The simplistic first step in reanimation was interpolating and extrapolating between the 'low' and 'high' intensity performances of the Neutral Phrase. In the motion capture database, examples of the Neutral Phrase were chosen that had the most extreme annotation as representatives of the 'high' and 'low' case.

The two example motion segments were aligned using dynamic time warping. Then the low and high were interpolated and extrapolated for values $\alpha = -2, ...2$. Since the interpolation and extrapolation was performed on adjoining angles, spherical interpolation was employed.

The resulting animations, especially in the high intensity extrapolations, were realistic and much 'stronger' in appearance than any of the original motion capture recordings. However, when shown to LMA experts it was not clear that the Laban effort parameters had been combined in a linear manner. One annotator suggested new effort parameters that had not existed in the original motions were being exhibited.

In an effort to understand how a linear combination of features affected the Laban effort space, an extensive database of interpolated and extrapolated motion was created. The Neutral Phrase was subdivided into core components based on the originally choreographed motion. The short segments were placed

in an on-line program that played simultaneous animations of interpolated motion versus a control of either the high or low phrase. Based on the original effort parameters used to describe the phrase, a question was posed as to which animation was a better demonstration of the Laban parameter (and pole) in question.

The program was sent to Laban experts who answered hundreds of questions on the animation. The intent was to create a baseline of comparison in order to fully learn the nature of the Laban effort space.

7.6 Future Work : Personality Division

The collection of a large selection of data will enable a more careful study of the Laban parameters. However, as was learned with much time spent with the Neutral Phrase – segmenting data based purely upon intended Laban effort parameters is quite difficult. The variety among performers made it arduous to establish a baseline for a given effort parameter.

As originally was suggested, the key to establishing a differentiation between content and style is most likely rooted in a frequency decomposition of motion signals. However, finding an exact mapping to Laban effort parameters may be a futile pursuit. The Laban effort parameters were established on important but somewhat objectively nebulous premises of 'intent'. As such, determining an 'indirect' versus a 'direct' movement in a recorded performance may not lie solely in the motion capture data. This was result demonstrated by often conflicting classifications by Laban annotators for the same movement. Seeking to convey such a quality may rely on the inclusion of key perceptual cues such as eye movement.

A second observation in the Laban experiments was the continual combination of a set of 2 or 3 parameters at given poles that almost always seemed to appear in combination. Two examples are 'strong' and 'bound' or 'light and 'free'. This observation was in line with the statements of the dancer Peggy Hackney, who claimed that most individuals had a Laban personality. A person tended to confine all movements to 2 or 3 Laban parameters. In annotating the Neutral Phrase, she showed examples of the different dancer 'personalities', despite the fact that all the performers were supposed to be attempting to achieve the same Laban characteristics.

In addition, isolating movement and comparing limited phrases can be difficult, especially when the parameter being exhibited is subtle. As such, the author proposes future work that employs a more global approach: let average people perform similar movements, such as dancing or walking, but not in a constrained set phrase, in an unconstrained manner. Then attempt to generalize the movements of the person as a given Laban personality.

The Laban personalities, which are correctly defined as 'states' or 'drives are as follows :

1. **Awake State - Space and Time**
2. **Dreamlike State -Weight and Flow**
3. **Remote State - Space and Flow**
4. **Near State - Weight and Time**
5. **Stable State - Space and Weight**
6. **Mobile State - Time and Flow**

7. **Action Drive - Space, Weight and Time**

8. **Passion Drive - Flow, Weight and Time**

9. **Vision Drive - Space, Time and Flow**

10. **Spell Drive - Space, Weight and Flow**

Segment the motions according to natural endpoints (zero crossings in velocity/ angular velocity or other means) and gather segments into vectors of similar size (or apply Fourier transform to normalize in time). Using LLE and multiple discriminant analysis techniques attempt to segment the data into 1 of the 10 states. Alternatively, since it is a supervised learning problem, a linear classifier could be constructed and the results run through a neural network (the only difficulty with this scheme is learning with quaternions and/or exponential map representations of angles).

If it is possible to segment normalized data segments into the correct Laban personalities, which are determined by Laban experts as the output, then any motion can be subdivided and assigned a classification of personality in a large database.

A new personality can be applied to an individual by applying frequency decomposition to the motion in question with a method similar to the technique used by Kathy Pullen [60]. The low band frequencies of the new motion, which represent the content of the motion, can be compared to motions in the personality database. Motions in the desired database with similar frequencies as the sample motion can be time warped or blended together to reconstruct sample motion in a new personality.

CHAPTER 8
SPINE BASED DEFORMABLE SKELETON

During motion capture trials attempting to isolate core Laban characteristics, LMA experts from the dance community repeatedly commented on the striking visually difference in expressive quality between the raw data points from the motion capture system of a dancer versus a character animated from the same data using a kinematic chain. These observations led to the development of a technique to train a deformable skeleton, which hopefully preserves the original expressive nature of the motion capture data. The starting point for this endeavor was the learning of a mapping between spine deformations/shapes and the rigid body kinematic skeleton often exported from motion capture systems for human character animation. This approach was also motivated by the robotic characteristics often attributed to even professional motion-capture based animations. LMA experts further hypothesized that the absence of a human spine contributed to the mechanical qualities of many data-based animations.

The principal intuitions behind the research are rooted in practices from traditional 2-D animation and observations voiced by dancers. The desirable qualities of motion capture including the ability to retarget motion to different characters of various dimensions and create entirely new actions by blending together motion segments, relies on the creation of a rigid body kinematic chain [31]. However, a standard, easily calibrated skeleton that assumes rigid bone structure and covers the basic joints of the human body sacrifices vital

information. While working with various dancers, the necessity of modeling the spine was reinforced. The majority of human motions begin in the core of the body and are articulated outward. It was hypothesized that the *mannequin* problem, that is when an animated figure appears to have disjointed arms and legs moving independently from a stiff torso, could be ameliorated by capturing the initiation of a movement from the base of the spine.

The spine also expands and contracts which visibly affects the entire shape of the body. As an example, consider a person slouched or brooding who jumps to life suddenly. Initially, the spine is contracted and the shoulders are hunched and restricted (appearing shorter in length), but the entire body expands/lengthens with the sudden movement. In traditional 2-D animations, the technique of *squash-and-stretch* amplifies these changes.

The system proposed in this section involves collecting a large database of highly detailed motion capture data, which precisely models the deformations of the spine using a spline representation (changes in other parts of the skeleton are also monitored). This detailed motion capture data is labeled (segmented) manually and subsequently paired with a corresponding standard rigid body skeletal model for each frame. A one to one correspondence between the standard model and spine data is assumed. A probability distribution for the pairs of data is learned by mapping the input to a lower dimensional nonlinear manifold. A deformable skeleton reconstruction can be formulated for any new motion sequence performed by a rigid body standard skeleton of arbitrary size by using the learned probability distribution to hypothesize the correct spine configuration. The deformable skeleton is smoothed and updated across time with a dynamic Bayesian process. With the aid of a simple filter the changes in the deformable skeleton can be exaggerated or diminished

throughout the body for enhanced emotional impact. The net result is a simple post-process technique that can apply a fluid spine, deformable bone lengths and exaggerated deformations with corresponding actions to any standard kinematic chain data.

The net goal is a deformable skeleton which can be applied to ANY motion capture data set. If a mapping is learned between a deformable skeleton based on the spine and a standard rigid body skeleton using a database of motions in a training set then any motion capture output can be easily modified. If these deformations can be further exaggerated or minimized through the entire body or in particular places, the animator has an easy tool for quickly adding emotion to a character. This post process technique is an intuitive method that could be part of a larger expressive toolbox accessible to an animator. It is especially useful for the efficient production of animations – the need for painstaking key-framing techniques is eliminated.

8.1 Spine Algorithm

A large and detailed motion capture session was performed where several people were recorded with motion capture markers glued to the vertebra on raw skin. In addition to the markers along the spine, a traditional set of motion capture markers was placed on the rest of the body. The raw spine data was converted into a spline representation and the traditional marker set was modeled as a rigid body skeleton. Originally, this project had aimed to use a neural network solution to find the mapping. However, initial attempts at using a neural network were unsuccessful. Some of these failures were most likely because of problems in representations of the data. (It took several iterations of experimenting with what should be included in the data.) The

choice of using quaternions was discussed extensively in previous chapters. However, neural network learning of quaternions is extremely difficult due to the spherical nature of the learning space. Even with the metrics provided in the previous chapter, quaternions still are not an ideal choice for the implicitly linear nature of the operations performed in neural networks. Future work of interest may be to explore explicit means of accomplishing this task. A recent book was published on the use of quaternions in neural networks and may contribute to fulfilling the original desired task. A large variety of learning techniques were tested as well as different means of representing the data. Techniques included Principle Component Analysis (PCA), mixtures of Gaussians, K Means Clustering, Local Linear PCA, and Neural Nets. The decision to use these techniques was motivated by reading papers on learning with missing data points from motion capture. In each case the paper used a mixture of Gaussians approximation or a variant of PCA.

8.1.1 Data Acquisition

Several initial experiments consisted of observing whether the spine actually had visible deformations and changes in length when motion captured. Motion capture markers were glued to the vertebrae of the spine. The results were impressive - the changes in length between vertebrae were quite large and the various shapes (that translated to the entire body) were encouraging. A large database of various motions designed to cover a broad range of body configurations was collected from several different subjects.

Data was collected using a 10 camera VICON system. 3mm round markers were glued to bare skin along the vertebrae of the spine as well as key areas of deformation in the core of the body. These areas include clavicles, ribs,

Figure 8.19: Spine Setup

chest, scapula and stomach.

In addition, markers were placed in the positions on the body of a standard motion capture marker set (see Figure 5.1). Theses markers consisted mostly of the joints on the arms and legs.

A database of motion capture sequences was collected using these marker positions on two subjects of different size and gender. The database consisted of many different motions that emphasized deformations in the spine as well as typical everyday movements.

Using the VICON system all the markers were labeled individually. The markers were classified as either deformable markers (part of the spine dataset, see Fig. 8.19) or skeleton markers, which became part of the rigid body skeleton (based on the standard marker set).

8.1.2 Data Representation

The data set consists of motion capture data outputted from a VICON system. In the case of the skeleton data, the data was smoothed, tracked and filtered by the VICON software as well as fit to a rigid hierarchical model.

It was decided that the root position and orientation would interfere

with learning. Therefore, each skeleton (and its spine data) was translated to a center coordinate system. In addition, each skeleton and spine was rotated around the z-axis so that each body was facing in the same direction. The corresponding root positions and rotations were stored for plotting the skeleton after learning was completed. For this project a subset of the database of 1,000 frames was used for learning and testing.

Input: Standard Rigid Skeleton

The skeleton was represented in a similar manner as the rigid body skeleton based on the standard marker model from the VICON system. The input included the quaternions for all the joints in the kinematic chain. For learning purposes, the root orientation and position was removed from data sets. All data was translated to a center coordinate system and rotated around the z axis so that each skeleton was facing in the same direction. For each frame, $t = 1, ...n$, each joint in quaternion format, q_j, where $j = 1, ..m$ for the $m = 17$ core joints in the kinematic chain described in the posture representation chapter, the central joint, q_0, was removed and then the entire skeleton was rotated around the z-axis by the appropriate amount to face forward.

$$\bar{q_j}(t) = q^z_{body}(t) \circ q^*_0(t) \circ q_j(t)$$

Note that each of the q_j are originally local rotations as specified in the kinematic chain format. The translation to root position was simply removal of the position of the root, in this case, the pelvis.

A crucial decision was made early in the process, that is, to learn the correlation between spine deformations and rigid body configurations based on a single frame and dynamics of the movement at a time step t. A possible alternative was to divide motion segments into intervals based on natural breakpoints. This approach is somewhat perilous for a variety of reasons.

First, while there are effective means of determining breakpoints in data such as determining the zero crossing of the velocity of the smoothed data, the methods are inappropriate when applied to the entire body if only some segments of the skeleton are moving. If we are investigating primarily arm movement in one phrase and then leg movement in the next, automatically determining how to weigh the data is confusing. Likewise, any noise in the data can create several small sections which are garbage. Manually segmenting the data is one possibility, but is extremely time consuming. For learning purposes, the data vectors have to be of the same length. Arbitrarily picking a vector size can lead to difficulties with motions of varying lengths. Finally, similar motions with only a change in timing will be viewed as separate entities.

As such, each frame is modeled individually with the inclusion of translational velocity $v = [v_x, v_y, v_z]$ and acceleration $a = [a_x, a_y, a_z]$ for the root position as well as angular acceleration and angular velocity for all joints. This model allows the dynamics to be explored and used for smoothing purposes over time with an AR(1) or AR(2) model. While not explored in this work, a possibility for aligning similar movements of various timings for learning purposes is to utilize the Fourier time normalization in the Laban Personality section. Movements are hand segmented and then normalized in time to the same number of frames.

Therefore, the input vector was a matrix of uncorrelated frames. Each vector represented a pose. Learning was aligning a given pose with certain

dynamics to a corresponding spine. The input vector for each frame t was:

$$in = \begin{bmatrix} v(t)' \\ a(t)' \\ \bar{q}_1(t)' \\ w_1(t)' \\ \dot{w}_1(t)' \\ ... \\ \bar{q}_m(t)' \\ w_m(t)' \\ \dot{w}_m(t)' \end{bmatrix}$$

Output: Spine

The spine representation was slightly more complex. The spine data was raw points from the VICON system. It was manually filtered with a Kalman Filter as well as a Butterworth filter to remove high frequency noise which is a product of the motion capture system. The deformable skeleton was modeled by creating a chain of interactions between markers. In particular, the spine was modeled as a spline of control points represented by the physical markers.

First a base representation of the spine in the standard T-Pose position was determined by calculating relative changes in angles over the chain and the actual length between markers. At each time frame, t, a ratio was calculated for the change in lengths between markers m in the chain, so

$$l_i(t) = \frac{m_i(t) - m_{i-1}(t)}{m_i(0) - m_{i-1}(0)}$$

where $t = 0$ is the base representation, $i = 1...n$ and n is the number of control points. Each control point rotation is calculated via quaternions such that the global rotation, R_i^{global} is equivalent to

$$R_i(t)^{global} = R_{i-1}^{global}(t) * R_i(0) * R_i(t)$$

and $R_i(t)$ is the incremental change in rotation from the standard base rotation $R_i(0)$.

A base spine representation of a root angle plus $k = 10$ control points and 10 corresponding distances was established. It should be noted that the base or root representation is predetermined and not used for learning. Therefore if a control point doesn't vary from the root position, the rotation will be zero (identity). Learning was conducted using the quaternion representation for the spine. Similar to the input representation of data, the rotation orientation of the body was subtracted from the data. Therefore all the spine data was oriented in the same direction. The root orientation was determined by the orientation of the $q_{body}^{-z}(t)$ for each frame used for the corresponding rigid body skeleton in the input vector.

Similarly the rate of change and acceleration in the ratios of bone lengths were calculated and included in the output data vector, with the angular accelerations and velocities. The final vector for each time frame was:
$out = [l_1(t), v_l1(t)a_l1(t), \bar{q}_1(t), w_1(t), \dot{w}_1(t), ...l_k(t), v_lk(t)a_lk(t), \bar{q}_k(t), w_k(t), \dot{w}_k(t)]$
for all frames $t = 1...n$.

8.2 Learning Experiments

8.2.1 PCA

The first learning procedure tested was principle components analysis. Its use was based on an initial experiment where body skeletons were plotted against all possible spine positions that were possible in the data set. It was determined for each body position that there was only 1 possible spine (this was

number of components	MSE
8	1.3411
32	0.2535

Table 8.1: MSE for Reconstructing Training Data

actually the opposite of the assumption when the project began). Therefore, it was established that there is a linear mapping between the input (skeletons) and output (spines).

The input and output data was stacked together as a single matrix and the principle components were calculated based on the variances. From the eigenvalues, it appeared that the first 9 components were significant. However, the eigenvalues did not start to trail off until around the 80th value. (Note that the data set used has 190 dimensions).

The principle components were used as a basis for the data. For testing, weights were constructed by using only the input portion of the vectors. With test vector x.

$$w_i = PC_i(1 : |input|)' \times x \tag{8.2}$$

The corresponding y output or spine was constructed by :

$$\begin{bmatrix} x \\ \\ y \end{bmatrix} = mn + \sum_i PC_i \times w_i \tag{8.3}$$

The corresponding mean squared area (MSE) is listed in Table 8.1.

There were several flaws with the initial experiments. When this work was first attempted the distance metric for skeletons was somewhat arbitrary.

number of principle components	MSE SKELETON	MSE SPINE
4	2.8782	1.6
8	2.68	1.6
16	2.32	1.6
32	2.0686	1.6

Table 8.2: MSE for Test Data

A more accurate distance metric for pose and alternatives will be discussed in later sections. In addition, the math used with the quaternions was not accurate. A constraint was used to maintain unit quaternions. However, as discussed extensively in the learning procedure, linear operations are not appropriate with quaternions. The use of strictly log maps can be appropriate, but checking for singular representations (antipodal geometry with quaternions) is difficult. Instead, the Principle Analysis Algorithm was modified as follows for the quaternion components of the data set. (Note that the ratios, angular velocities, etc, were treated normally).

1. Calculate the sample mean using the quaternion closed form euclidean mean formulation.

2. Ensure that the antipodal geometry for each quaternion in the dataset is the same as the mean.

3. Subtract out the mean and perform a log map for each data point i and joint j

$$\hat{q}_i(j) = ln(M_j^* \circ q_i(j))$$

4. Recombine the log map format of the quaternion vectors with the linear data into a column vector.

5. Perform normal PCA with the new vector.

PCA was performed on the entire data set. The one-to-one correspondence between pose data and spine data was used to project new skeletal data using the same weight coefficient into the lower dimensional manifold and predict the corresponding spine representation.

The number of principle components affected the MSE of the reconstruction data, so a graph was made of the reconstruction errors at various dimensions shown in Fig. 8.20.

However, despite the new form of PCA, the reconstruction errors for test data were disappointing. The plotted results were jittery and not fluent, see Fig. 8.21. While, some of the problems with the plotted results probably needed to be addressed with post processing techniques that took into account the timing and correlation of the data, it was wisely hypothesized that the nature of the data space was *nonlinear*. Principle Component Analysis is an intrinsically linear method of dimensionality reduction. An alternative means that would accurately address a nonlinear data set was sought.

An important aspect was also considered at this point, the use of dimensionality techniques to learn were based on the assumption that the data was embedded on a much lower manifold than the dimension of the vectors (the input vector was of dimension 121 and the output vector's dimension was 130 or 50 when angular velocity and acceleration are excluded). While this seems like a reasonable assumption and the graph of dimensionality seems to concur with this theory, the random nature of the database was of concern.

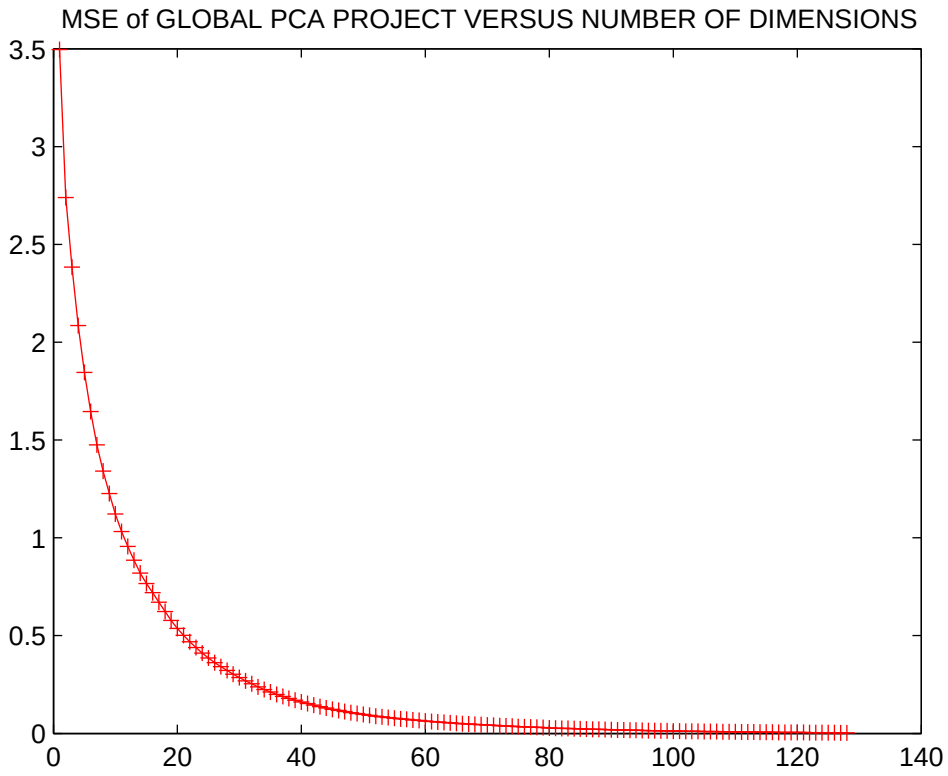

Figure 8.20: PCA dimension

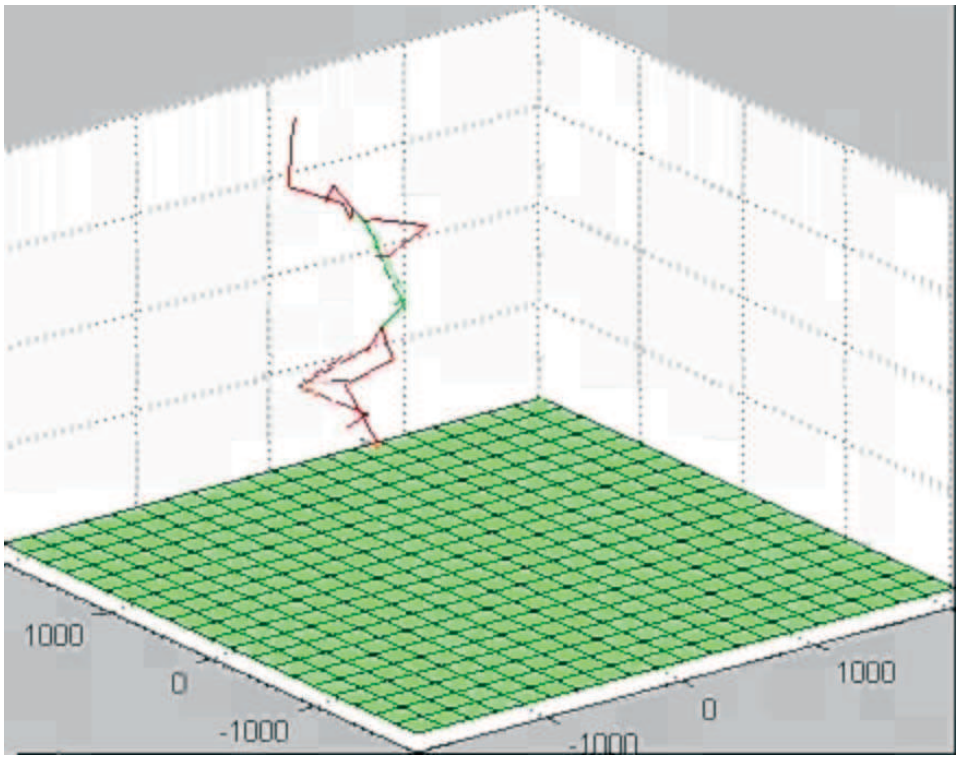

Figure 8.21: Animation of Body and Reconstructed Spine Using PCA

Other work in the motion capture realm that had combined learning procedures with motion databases focused on isolated sections of movement which demonstrated highly cyclic behavior. In these cases, the lower dimensionality of the space is explicit. The random nature of the very separate motions (but all modeled as isolated frames) collected in the database was somewhat concerning.

8.2.2 K Means

In seeking a way of representing the nonlinear manifold of posture data, K-means clustering was performed. The motivations for clustering were numerous. First, it was hoped that clustering would divide frames into categories of similar 'motions'. In this scheme, dynamics could be established for each cluster. The second motivation was a reduction of the dimensionality of the data so that it could be used as input in a nonlinear learning algorithm. Initially, the algorithm in question was mixture of Gaussians.

K-means clustering determines k clusters of data based on minimization of the following objective function :

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_j(i) - c_j||^2$$

where $||x_j(i) - c_j||^2$ is the distance between a data vector $x(i)$ and the cluster center c_j. The algorithm proceeds initially by assigning each of the data vectors to one of the clusters. Then it calculates the mean of each cluster. In this procedure, the data is separated into quaternions and linear data. The quaternion mean is calculated as specified earlier. The algorithm iterates until convergence with the following steps:

1. Assign each data vector to the cluster that has the closest centroid mean.

2. When all items have been assigned, recalculate the mean for each cluster.

3. Repeat until the centroids remain the same.

It should also be noted that the distance metric between quaternions was initially $dist(P, Q) = 2||ln(P^* \circ Q)||$. While much more time consuming, tests were also employed using the mahalanobis distance based on quaternion variance introduced earlier. However, a severe problem with this formulation is that each joint in the pose and spine is weighted equally. Therefore, a misplaced hand in a posture contributes as much to the distance as a completely different thorax or shoulder position. Obviously, this will lead to a clustering that does not accurately reflect human poses.

In addition, the actual mean square error is inaccurate without an appropriate metric for computing distance between poses. The distance metric was later modified in the final algorithm in accordance with these observations. By examining mean square errors, the initial results established that about 200+ K-Means would be needed to represent the subset database of 3000 frames (see below for details).

Another modification to the K-Means algorithm for clustering exists where the clusters are subdivided in half until all items in the cluster are within a set distance from the mean.

8.2.3 Mixture of Gaussians

Mixtures of Gaussians were tested as a means of approximating the non-linear manifold of the data. An initial implementation of mixture of Gaussians was used and an expectation maximization scheme was iterated until convergence. Since there was an established 1:1 mapping between input and output,

K-Means	MSE
8	2.3376
16	1.7829
32	1.3934
64	1.0284
128	.7007
256	.4037
512	.2112

Table 8.3: MSE K-Means Test

a solution similar to PCA could be used with mixtures of Gaussians. A conditional expectation of an output y can be found based on input x, when the data is modeled together as a mixture of n Gaussians.

$$p(x, y|i) = \frac{1}{2\pi\sqrt{|\sum_i|}} exp^{\frac{1}{2}(x-\mu_i)^T \sum_i^{-1}(x-\mu_i)}$$

where $x = \begin{bmatrix} x \\ y \end{bmatrix}$, $\mu_i = \begin{bmatrix} \mu_{x,i} \\ \mu_{y,i} \end{bmatrix}$, $\sum_i = \begin{bmatrix} \sigma_{x,i}^2 & \sigma_{xy,i} \\ \sigma_{xy,i} & \sigma_{y,i}^2 \end{bmatrix}$ The value of y is calculated as :

$$\hat{y} = \mu_{y,i} + \frac{\sigma_{xy,i}}{\sigma_{x,i}^2}(x - \mu_{x,i})$$

and the expected value of y is :

$$\hat{y} = \sum_i h_i \hat{y}_i$$

where the value of h, using $P(x|i) = N(x, \mu_{x,i}, \sigma_{x,i}^2)$ is

$$h_i(x) = \frac{P(x|i)}{\sum_j P(x|j)}$$

The initial plan was to use this solution to map the input to output. While it was not tested, it might have been a better plan to also calculate the expected variance of y, then have a Gaussian distribution at each time frame

and smoothly interpolate (or use spherical interpolation for quaternions) across time frames.

Unfortunately, many problems were encountered when using mixtures of Gaussians. The above solution worked fine for a toy problem using expectations to obtain a mixture of Gaussians. However, it was finding the appropriate mixture of Gaussians that posed a problem with the real data set. The main problem was that the covariance matrices in the mixture of Gaussians became singular early on. After thinking about the problem, this seemed like a desirable result. If certain values of the covariance were going to zero that meant we had hit a subspace in the manifold. On the other hand, if the matrix was not invertible, then probabilities could not be calculated.

A variety of tricks were attempted to keep the matrix invertible. First small values were added to the diagonal of the covariance. This worked in toy examples, but failed with the motion data. Next all the values in the covariance matrix were maintained above a certain threshold (not just zero). However, with numerical issues, the matrix still remained singular in some instances and the probabilities from the Gaussian distribution were unreliable. Finally, I took the svd of the covariance matrix. I set all singular value below a threshold to zero and inverted the rest of the values. Then I reconstructed the covariance matrix $u^* \frac{1}{s}^* v$. In addition, I set the determinant to the product of the nonzero singular values. However, this solution was extremely slow and yielded unsatisfactory probabilities.

Frustrated at this point, I followed the suggestion in a paper that just approximated the mixture of Gaussians from K-Means clusters. The covariances were estimated using svd decomposition. Each cluster was established

with data and the means of each cluster were subtracted. The svd was taken of the matrix formed for each data set. The covariance was reconstructed by $\frac{1}{n} * U * S^2 * U^T$. The approximate mixture of Gaussians was run on the dataset and the above solution was used to find y on a test set. The mean squared error for the spine was around 1.3 which is less than ideal. In addition the movie output is not great. Spines jump in and out, but there are some subtleties that are preserved.

8.2.4 Posture Metric

Since many of the problems in learning were attributed to the actual distance function, a new metric was established. A hierarchical weighting function was created. The angles and lengths associated with the torso and core body parts were given higher weights than the limbs and outer extremities. The weight functions had the restriction that for $w = [w_1...w_m]$, where m is the sum of the number of joints, velocities, accelerations in the data vector, $\sum_i w_i = 1$. The distance between two poses, x and y is then constructed as :

$$Dist_p(x, y) = \sqrt{|x - y|^T W |x - y|}$$

where W is a diagonal matrix of the weights and $|x-y|$ uses the corresponding distance functions for quaternions and linear data.

In applicable situations, for example clustering applications, the quaternion mahalanobis distance was used. For linear data the general mahalanobis distance was included :

$$D(x) = \sqrt{(x - \mu)^T \sum^{-1} (x - \mu)}$$

Ideally, the weights themselves are learned according to importance from the data itself.

8.2.5 Nonlinear PCA

Instead, of using traditional Mixtures of Gaussians in which the expectation algorithm required inversion of singular matrices, Local Linear Projection was employed. In the Local Linear Projection of PCA, the data is clustered and PCA is performed on each of the subgroups. The grouping can be approximated with K-Means clustering or by finding the k-nearest neighbors.

The approach utilized in the final version was finding the k-nearest neighbors using the posture metric identified above. In forming the projection, this leaves 2 parameters undetermined: the number k and the dimension d that will be used for reconstruction. These parameters were determined for the dataset using 'leave one out' cross validation.

Table 8.4 summarizes some of the statistics from leave one out cross validation on the training data and test data for a substantially different motion than anything found in the database (the test data was violent ninja fighting).

From the table, it is clear that the test and training error minimum occurred around $d = 9$ and $k = 75$.

The reconstruction of data can be very simply defined by finding the cluster whose mean has the minimum distance to each of the new test input frames. The weights for the projection are the signals computed for the given cluster. Alternatively, a new data vector can be reconstructed as the sum of weighted components of the n-closest clusters. The weights are determined by the distances of the input vector to each of the PCA clusters. The weights are constrained so that the sum of components is 1, $\sum_j w_j = 1$.

k neighbors	dimension	MSE Training	MSE Test
5	3	.0020	2.8017
5	6	.0026	2.8558
5	9	.0020	2.8610
5	12	.0020	2.7471
25	3	.0026	2.1436
25	6	.0026	2.1399
25	9	.0026	2.1514
25	12	.0026	2.1524
50	3	.0014	2.0357
50	6	.0021	2.0162
50	9	.0020	1.9921
50	12	.017	1.9991
65	9	.013	1.9801
75	9	.012	1.9774
85	9	.013	2.0029

Table 8.4: Nonlinear PCA MSE for dimension and k-neighbors

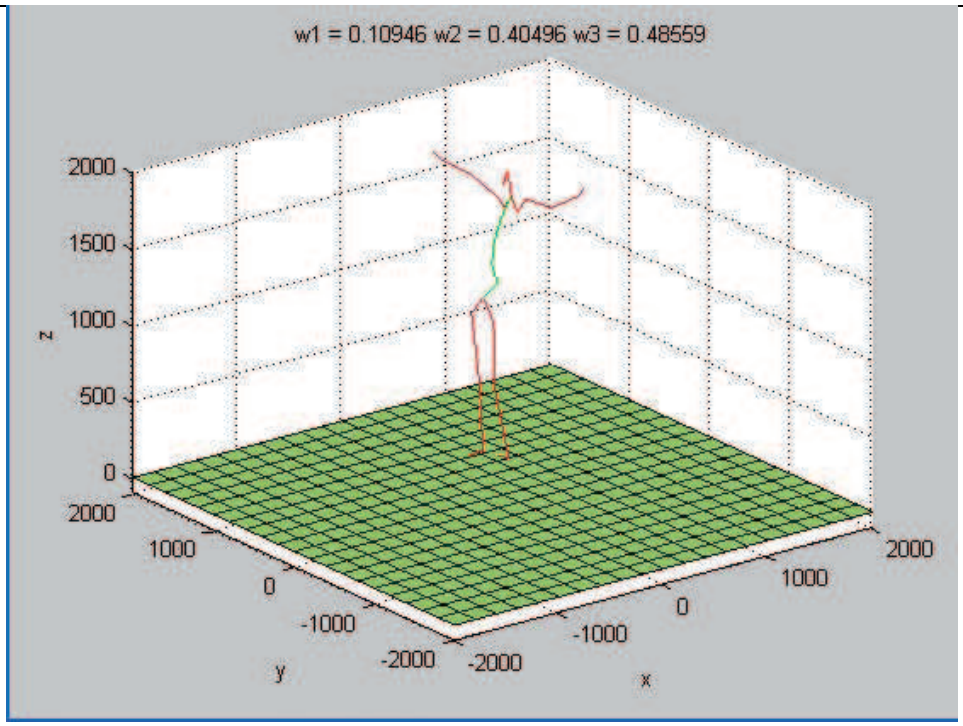

Figure 8.22: Reanimated Test Data using Local Linear PCA

If the PCA weight signals are denoted s, then the reconstruction is performed by :

$$x_r = \sum_j w_j * (\mu_j + \sum_i PC_i \times s_i)$$

8.3 Deformable Skeletons Algorithm

The final successful algorithm utilized to construct a mapping between a rigid body skeleton and deformable spine is defined here. A nonlinear manifold was approximated using piecewise PCA. However, additional techniques had to be employed to construct a smooth correlation between time frames. In clustering the data, all quaternion distances were calculated using a weighted standard quaternion distance function. The weight function was based on a

hierarchy that favored core angles in the spine and standard body representation.

Learning Algorithm:

1. K-means Clustering on Spine Data. Clusters further subdivided so that all members are within the threshold of the mean.

2. Perform PCA on each cluster.

Reconstruction Algorithm for new input sequence :

1. At each time step, t, center and revolve around the z axis all the standard data to face the same direction. Transform quaternions to log map format.

2. Assuming a Gaussian distribution, use cluster covariances and means to find probabilities of the standard skeleton at the frame, t, belonging to each cluster.

3. Find the weight signal for the PCA projection for each cluster. Project standard skeleton into full data PCA space for each cluster and hence obtain spine approximation.

4. Convert spine reconstructions for each cluster back into quaternion space for a given frame. Weight the clusters based on the initial probabilities and perform interpolation between ratios and quaternion using the slime algorithm.

5. Take the reconstructed spine lengths, rotations, velocities, accelerations, angular velocities/accelerations as the measurement value, z, in the Kalman filter. Using the previously reconstructed spine model at time

$t-1$, update the filter to find the new measurement. Repeat for all time frames.

6. Use a template for the T-pose root position based on the skeleton to be animated. Use the deformed spine angles and ratios across frames to calculate physical positions of the body.

The smoothing step using a modification of the Kalman filter is similar to techniques from Bayesian dynamical models. Since the learning is modeling the dynamics, steps could be included to incorporate the dynamical model further into the learning model. This representation is similar to Gaussian Process Dynamical Models which are being published this year as new methods of tracking [75]. The Kalman filter tracking is slightly more complex than the AR(1) model since it has to accommodate orientation as well as position tracking. The functions used in computing errors in the original Kalman filter are linear and therefore are poor approximations when used in conjunction with quaternions. Instead an unscented Kalman filter is used as an alternative [41].

In the unscented Kalman filter, the functions for estimating the Kalman gain function are altered. Recall from the posture chapter that the measurement model, H, in the Kalman filter relates the measurement, z, to the state vector, x, and describes the influence of random noise on the measurement. A general form of this interaction is :

$$z_k = H(x_k, u_k)$$

However, in this situation there is generally no matrix, H, which can fulfill the nonlinear aspect of acceleration. In the classical Kalman filter, the Kalman

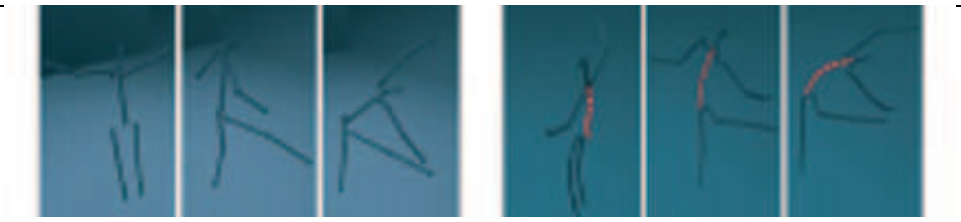

Figure 8.23: Shots from animation of input data of rigid body skeleton and the deformable spine results

gain is calculated using the function:

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1}$$

where R is the measurement error covariance and P_k^- is the estimation error covariance. In the unscented Kalman filter the value P_{zz}, which is the measurement vector covariances, is substituted for the values $H P_k^- H^T$.

$$P_{zz} = \frac{1}{2n} \sum_{i=1}^{2n} (Z_i - z_k^-)(Z_i - z_k^-)^T$$

Similarly, the value of $P_k^- H^T$ in the classical Kalman filter is substituted with the crosscorrelation matrix P_{xz}

$$P_{xz} = \frac{1}{2n} \sum_{i=1}^{2n} (X_i - \bar{x})(Z_i - z_k^-)^T$$

Therefore, the unscented Kalman Gain function is reduced to :

$$K_k = P_{xz} P_{vv}^{-1}$$

Expressive Exaggeration

Amplifying and diminishing select segments of frames is accomplished with a simple tunable parameter that modifies the acceleration of the ratio of lengths: $l_i(t) = l_i(t-1) + v_i(t-1) + c * a_i(t-1)$ where $c < \frac{1}{2}$ to amplify, preferably with $c < 0$, and $c > \frac{1}{2}$ signals to diminish change in lengths. These deformations should be applied to the appropriate limbs and bone segments. In addition, modifications in angles in the spine are also exaggerated in a similar

manner. Key limbs can be identified as demonstrating similar properties as the line of motion of the body – someone throwing a punch will extend the spine from the core and will have an increase in the acceleration of the spine as well as the position of the arm. Limbs whose movements coincide with spine movement, s, have the deforming parameters applied to their bone length.

Future work includes pursuing other non-linear techniques to find a better mapping of $p(s_t|skel_t, s_{t-1})$. Modeling appropriate skin deformations with these learned skeletal deformations could potentially heighten the expressive results. Dynamics could also be incorporated into the system.

Adding complex precision to standard motion capture data with a simple learning technique is an important contribution. The inclusion of a spine offers valuable visual cues and the exaggerated results further the expressive impact.

8.4 Other Learning Methods and Improvements

After a working algorithm was established, other techniques were experimented with to improve results. Recent algorithms that boost efficient and optimal means of learning nonlinear manifolds were attempted.

8.4.1 Eigenmaps

Eigenmaps is a relatively new technique based on the properties of the Laplace Beltrami operator. The method is similar to the technique for determining an optimal nonlinear low dimensional embedding (manifold) such as Roweis and Saul's LLE algorithm. It should be noted that Laplacian Eigenmaps are also related and motivated by spectral clustering [8].

Eigenmaps like other dimensionality reduction algorithms finds a low dimensional nonlinear manifold upon which a high dimensional data set lies.

The manifold is approximated by computing an adjacency graph. The Laplace Beltrami operator is approximated by the weighted Laplacian of the adjacency graph. A key component of the algorithm is that the Laplace Beltrami operator in the heat equation allows the heat kernel (which reduces to a Gaussian function) to choose the weight decay function.

The algorithm preserves locality making it insensitive to outliers and noise. As a result, the method also preserves natural clusters in the data. Global methods [70] do not show a tendency to cluster. In cases where data sets do not contain meaningful clusters, PCA or isomaps may be more appropriate. The algorithm is as follows :

- **STEP 1:** Construct the adjacency graph:
  - ϵ-neighborhoods
  - k nearest neighbors
- **STEP 2:** Choose the weights with the heat kernel:

$$W_{ij} = e^{-\frac{||x_i - y_j||^2}{t}}$$

$$\text{when } t = \infty \; W_{ij} = \begin{cases} 1 & \text{i and j are connected} \\ 0 & \text{i and j are not connected} \end{cases}$$

- **STEP 3:** Compute eigenvalues and eigenvectors for the generalized eigenvector problem :

$$Lf = \lambda Df$$
$$D_{ii} = \sum_j W_{ji}$$
$$L = D - W$$

The dataset was decomposed into the eigenmap representation. However, since the data was listed in simple pose format it was difficult to visual how the poses were clustered in the new representation. Learning was performed by finding the eigenmap embedding. New poses were learned by projecting onto the eigenmap space and computing nearest neighbors with the embedding. The neighbors were used to reconstruct an estimated pose for each frame.

The results in the basic implementation did not yield visually better results than simply using nonlinear PCA. The problem with this method is that there is no clear means of mapping between the new data and the embedding. It has been suggested that a simple neural network could be used to learn the mapping, but was not attempted in the literature or in this project.

8.4.2 LLE and GPLVM

Local Linear Embedding (LLE), which is very similar in nature to eigenmaps, is a means of preserving the local structure of a dataset on a nonlinear manifold. While extremely easy to implement, the results were questionable on the actual data set. A toy problem yielded fine output, but the author was not able to determine clear separation of data using the method on the motion database. In addition, LLE does not yet have a clear means of mapping, $X \rightarrow Y$, between the data set and the embedding space for new data.

Gaussian Process Latent Variable Models (GPLVM), have become extremely popular learning tools in the last two years. GPLVM associate a Gaussian process with the dataset and in simplistic cases reduce to basic PCA. They were used in one instance for learning information on a motion data base. However, this database involved isolated cyclic motions. This author attempted running the original code by Lawrence Saul, but only arrived at a scattered

plot with no clear path or separation with the database of motion used for the spine.

8.4.3 Binary Database

Finally, it was hypothesized that many difficulties in learning were strictly due to the way that was used to distinguish differences in body posture. While the improved distance metric made learning possible, more improvements could be made. Two alternatives were posed for querying the database when searching for the k-nearest neighbors.

The first alternative is based on preserving timing and continuity between spines and poses. A query is made to the database or, rather, the cluster means in locally linear PCA at each step by using the distance of the current pose as well as spine at the previous time step. The query is as follows :

$$Dist(\tilde{d}_i, c_j) = \alpha * dist(\tilde{b}_i, b) + (1 - \alpha) * dist(\hat{s}_{i-1}, s) \tag{8.4}$$

$$\text{where } d = \begin{bmatrix} b \\ \\ s \end{bmatrix} \in Database$$

$$\tilde{d} = \text{ observed data}$$

$$\hat{s} = \text{ predicted spine}$$

$$b = \text{body skeleton}$$

and c is a cluster mean. This improved query ensures continuity in time steps of the reconstructions and works well in combination with the Kalman filter.

The second improvement is to construct a 'binary' query metric. The metric will improve efficiency in querying the database in larger scale applications and may add a more intuitive notion of pose. Instead of comparing all the joint angles in the entire body, a set of binary questions are assigned to each pose. The set of questions are of the nature, is the right arm up ?

or is the right leg in front of the left ? These binary values are computed automatically by comparing the joint angles in each pose. Therefore a query searches fro the pose cluster that matches the set of all the similar constraints as the original pose.

In the spine project, the query metric was only applied in loose form to the rigid skeletal data (the spine data remained in the same spline representation). While an improvement was not seen in the learning MSE, it did ensure that postures in the same cluster retained similar characteristics.

8.5 Future Work

This paper has presented a system for learning a mapping between a rigid kinematic chain and a deformable skeleton. The author envisions that the dynamic deformable model can be easily applied to rigid animations for enhanced expressibility. Future work could include improving the learning algorithm. The inclusion of dynamics into the motion model may be key in creating a true 'squash and stretch' effect.

However, the idea of learning from motion databases may be of greater importance. To date, learning performed on motion data has focused on narrow cyclic style based motions. Learning the dynamics within a cluster of motions may be key to modeling interactions within data. In addition, finding appropriate means to deal with all aspects of quaternions (as in neural nets) will allow far more sophisticated procedures to be performed.

CHAPTER 9
HANDS, BREATHING AND SUBTLE ASPECTS OF MOTION

9.1 Speaking with Hands

A joint collaboration between Matthew Stone and Doug DeCarlo at Rutgers and Chris Bregler led to a paper outlining the importance of hand gesture in conveying meaning during speech [**?**]. The basic premise behind the project was that human utterances are composed of short, clearly-delimited phrases that coincide with gestures that impart meaning and emphasis. The project outlined a method for using a database of recorded motion capture data and speech to create animated conversational characters. The system recombines motion samples with new speech samples to recreate coherent phrases and blend segments of speech and motion together into extended utterances. While the project focused mainly on the construction of databases to create extended animated performances, the results demonstrated that effective hand movement eliminated the need for lips in an expressive performance.

The project presents an integrated framework for creating interactive, embodied, talking characters from a human performance. The approach brings together two key contributions. First, the performer's intuitions about a character's behavior is an alternative to more comprehensive reasoning about communication. Natural language generation techniques link the performer's script to aspects of the application state. Annotators judge whether motions

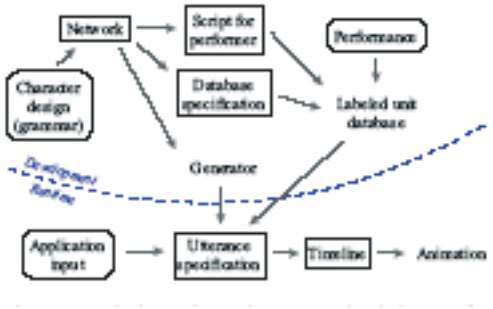

Figure 9.24: Chart demonstrates the design of an interactive conversational character from performance data

performed along with a script are descriptive or expressive. Second, the approach selects and concatenates units of speech and units of motion as part of a single optimization process. The units of speech are short phrases from the original performance and the intervals correspond to intonation phrases.

The character design, a predefined grammar of phrases, is included in the system. The performer's script is computed from a network. The network characterizes the motion capture elements and sets them into a database. The network is compiled into a generation engine. The database also includes descriptions for timing and content of gestures which is annotated with machine assistance. At run-time, the application input, generation system and database together determine a space of possible utterance realizations. An optimization is performed to derive the time line for an utterance which is then animated by stitching together segments of motion and speech performance.

To animate phrases, a suitable sound recording and gesture must be matched. Each edge traversed in the generator corresponds to a choice of sound and a choice of motion for realization. Dynamic programming was used to solve simultaneously for combinations of speech and gesture. A cost, $cost_d(s_i, m_j)$,

Figure 9.25: Frames from Speaking with Hands Animations

was assigned to delivering each phrase based on the match between its sounds, s_i, and motion, m_j. The cost has one term to penalize the amount of time warping required to align the motion with sounds. In addition, a cost for splicing between successive sounds and motions, $cost_s(s_i, m_j, s_k, m_l)$, is assigned. The objective function is of the form $\sum_{p=1}^{n} cost_d(S_p, M_p) + \sum_{p=1}^{n-1} cost_s(S_p, M_p, S_{p+1}, M_{p+q})$. Standard time warping, established by Witkin and Popovic, is used to blend together motion phrases.

Animations frames are shown in Fig. 9.25. The paper was successful in providing a compact system that allows a broad range of conversational phrases to be created from a relatively short database of motion and speech. However, a crucial observation was that the simultaneous delivery of phrases of emotional impact with correct gestures eliminated the need for detailed hand motion and mouth movement.

9.2 Breathing and Other Subtleties

An important result from the speaking with hands paper is that expressive movement relies on certain perceptual cues. Certain details may be sacrificed, if the visual emphasis is correct. Along similar lines, the addition

of breathing and skin deformations with motion capture data can aid the expressive qualities of animation.

The post processing addition of subtle human movement onto motion capture data may appear counter-intuitive to the previous conclusions made about the hand gesture project. However, the basic premise is similar. Imagine a performer engaging in an energetic movement such as a strenuous run. The eye expects to see panting and likewise deformations of the upper torso. This small involuntary movement is key in creating expressive animations. Lack of such features can lead to the strange sensation of the 'uncanny valley', when an animation appears extremely realistic but the key human aspects are missing creating an unpleasant sensation.

Trial motion capture sessions were performed with Eitan Grinspun and Jeff Han recording performer's movements while simultaneously using a breath sensor to monitor breath flow. The sessions were encouraging. Future work hopes to use a method similar to the deformable skeleton algorithm to learn breathing patterns from motion data. The patterns can then be mapped to a model of the human chest. Given new motion capture data, the model will be able to deform in the appropriate manner as an actual subject breathing.

CHAPTER 10
CONCLUSIONS

This thesis entailed three principal projects aimed at examining the expressive attributes of motion. The investigation of the application of Laban notation from dance, found means of separating segments of motion data based on the effort exerted by a performer. While the use of the Fourier transform to perform frequency decomposition and linear discriminant analysis produced successful results, the intrinsic difficulties in objectively characterizing Laban effort parameters suggests a new approach is needed. As such a future system that creates a system for segmenting based on personality was suggested.

This second project constituted the core of thesis. The concept of a deformable skeleton has broad applications in the field of animation and may be a crucial change needed to creating believable animating characters. The work involved a system for learning a mapping between a rigid kinematic chain and a deformable skeleton. The author envisions that the dynamic deformable model can be easily applied to rigid animations for enhanced expressivity. Future work could include improving the learning algorithm. The inclusion of dynamics into the motion model may be key in creating a true 'squash and stretch' effect.

However, the idea of learning from motion databases may be of greater importance. To date, learning performed on motion data has focused on narrow cyclic style based motions. Learning the dynamics within a cluster of motions may be key to modeling interactions within data. In addition, finding

appropriate means to deal with all aspects of quaternions (as in neural nets) will allow far more sophisticated procedures to be performed.

Finally, a paper using small gestures hand gestures that compliment phrases from speech illustrated the visually important subtly cues need to be emphasized for emotional impact. Future work suggests further exploration of breathing and eyes.

REFERENCES

[1] A. Agarwal and B. Triggs. Tracking articulated motion with piecewise learned dynamical models. *European Conference on Computer Vision*, 2004.

[2] O. Arikan, R. Forsyth, and J. O'Brien. Motion synthesis from annotations. *Proceedings of Computer Graphics (SIGGRAPH 2003)*, 2003.

[3] G. Ashraf and K. Wong. Dynamic time warp based framespace interpolation for motion editing. *Graphics Interface*, 2000.

[4] N. Badler. A computational alternative to effort notation. *Dance Technology: Current Applications and Future Trends*, 1989.

[5] S. Balakrishnama and A. Ganapathiraju. *Linear Discriminant Analysis: A Brief Tutorial*. Mississippi State University.

[6] J. Barbic, A. Safonova, J. Pan, C. Faloutsos, J. Hodgins, and N. Pollard. Segmenting motion capture data into distinct behaviors. *Proceedings of Graphics Interface (GI 2004)*, 2004.

[7] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *Proc. IEEE Transactions Pattern Analysis and Machine Intelligence*, 19(7), 1997.

[8] M. Belking and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2003.

[9] B. Bodenheimer, C. Rose, S. Rosenthal, and J. Pella. The process of motion capture: Dealing with the data. *Eurographics*, 1997.

[10] R. Bowden. Learning statistical models of human motion. *IEEE Workshop on Human Modeling, Analysis and Synthesis (CVPR)*, 2000.

[11] M. Brand and A. Hertzmann. Style machines. *In Computer Graphics (SIGGRAPH 2000)*, 2000.

[12] C. Bregler. Learning and recognizing human dynamics in video sequences. *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1997.

[13] C. Bregler, L. Loeb, E. Chuang, and H. Deshpande. Turning to the masters: Motion capturing cartoons. *Proceedings of Computer Grahpics*, 2002.

[14] C. Bregler and S. Omohundro. Nonlinear image interpolation using manifold learning. *Advances in Neural Information Processing Systems*, 7:973–980, 1995.

[15] A. Bruderlin and L. Williams. Motion signal processing. *In Proceedings of ACM SIGGRAPH 95 Annual Conference Series*, pages 97–104, Aug 2003.

[16] P. J. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *Proc. IEEE Transactions on Communications*, 31(4), 1983.

[17] S. Buss and J. Fillmore. Spherical averages and applications to spherical splines and interpolation. *ACM Transactions on Graphics*, 20(2):95–126, 2001.

[18] J. Cassell, C. Pelachaud, N. Badler, M. Steedman, B. Achorn, W. Becket, B. Douville, S. Prevost, and M. Stone. Animated conversation: Rule-based generation of facial expression, gesture and spoken intonation for multiple coversational agents. *Proceedings of SIGGRAPH 1994*, 1994.

[19] J. Cassell, H. Vilhjalmsson, and T. Bickmore. Beat: The behavior expression animation toolkit. *Proceedings of Computer Graphics(SIGGRAPH 2001)*, 2001.

[20] J. Chai and J. Hodgins. Performance animation from low-dimensional control signals. *Proceedings from ACM SIGGRAPH 2005*, 2005.

[21] D. Chi, M. Costa, L. Zhao, and N. Badler. The emote model for effort and shape. *Proceedings of SIGGRAPH 2000*, pages 173–182, jul 2000.

[22] S. Choe and J. Faraway. Modeling head and hand orientation during motion using quaternions. *SAE Transactions*, 2004.

[23] J. W. Davis and V. S. Kannappan. Expressive feature for movement exaggeration. *Proceedings of ACM SIGGRAPH 2002*, page 182, 2003.

[24] J. S. DeBonet. Multiresolution sampling procedure for analysis and synthesis of texture images. *Computer Graphics Proceedings (SIGGRAPH 1997)*, 31:91 – 114, 1997.

[25] R. DeMori and D. Probst. *Handbook of Pattern Recognition and Image*

Processing. Academic Press, 1986.

[26] R. O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

[27] A. Elgammal and C. Lee. Separating style and content on a nonlinear manifold. *CVPR*, 2004.

[28] P. Faloutsos, M. van de Panne, and D. Terzopoulos. Composable controllers for physics-based character animation. *Proceedings of SIGGRAPH 2001*, 2001.

[29] A. Fang and N. Pollard. Efficient synthesis of physically valid human motion. *Proceedings of Computer Graphics (SIGGRAPH 2003)*, 2003.

[30] M. Gleicher. Motion editing with spacetime constraints. *Symposium on Interactive 3D Graphics*, 1997.

[31] M. Gleicher. Retargetting motion to new characters. *Proceedings of SIGGRAPH 1998*, 1998.

[32] M. Gleicher. *Motion Capture and Motion Editing: Bridging Principles and Practice*. A. K. Publishers, 2000.

[33] M. Gleicher. Comparative analysis of constraint-based motion editing methods. *Graphical Models*, 2001.

[34] F. Grassia. Practical parameterization of rotations using exponential map. *The Journal of GRaphics Tools*, 1998.

[35] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popovic. Style-based

inverse kinematics. *Proceedings of SIGGRAPH 2004*, 2004.

[36] J. Hodgins and W. Wooten. Animating human athletes. *Proceedings of the 22nd Conference on Computer Graphics and Interactive Technique*, 1995.

[37] M. Johnson. *Exploiting Quaternions to Support Expressive Interactive Character Motion*. PhD thesis, Massachussettes Institute of Technology, 2003.

[38] L. Kovar and M. Gleicher. Flexible automatic motion blending with registration curves. *Proceedings of 2003 Symposium on Computer Animation*, Jul 2003.

[39] L. Kovar and M. Gleicher. Automated extraction and parameterization of motions in large data sets. *Proceedings of Computer Graphics (SIGGRAPH 2004)*, 2004.

[40] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. *Proceedings of SIGGRAPH 2002*, 2002.

[41] E. Kraft. A quaternion-based unscented kalman filter for orientation tracking. *ISIF*, 2003.

[42] J. LaViola. A comparison of uncented and extended kalman filter for estimating quaternion motion. *Proceedings of the 2003 American Control Conference*, 2003.

[43] N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. *Advances in Neural Information Processing*

Systems, 2004.

[44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[45] Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. *Neural Networks: Tricks of the Trade*, 1998.

[46] J. Lee, J. Chai, P. Reitsman, J. Hodgins, and N. Pollard. Interactive control of avatars animated with human motion data. *International Conference on Computer Graphics and Interactive Techniques*, 2002.

[47] J. Lee and A. Shin. Multiresolution motion analysis and synthesis. *The Internation Workshop on Human Modeling and Animation*, 2000.

[48] J. Lee and S. Shin. A hierarchical approach to interaction motion editing for human-like figures. *Computer Graphics Proceedings (SIGGRAPH 1999)*, 1999.

[49] Y. Li, T. Wang, and H. Shum. Motion texture: A two-level statistical model for character motion synthesis. *Computer Graphics Proceedgins (SIGGRAPH 2002)*, pages 465–472, 2002.

[50] C. Lui and Z. Popovic. Synthesis of complex dynamic character motion from simple animations. *ACM Transactions on Graphics*, 21, 1995.

[51] Y. Lui, R. T. Collins, and Y. Tsin. Gait sequence analysis using friezr patterns. *European Conference on COmputer Vision*, 2002.

[52] A. Maciel. Anatomy-based joint models for virtual human skeletons. *Proceedings of the Computer Animation*, 2002.

[53] V. Maletic. *Dance Dynamics Effort and Phrasing*. Academic Services, 2005.

[54] M. Moakher. Means and averaging in the group of rotations. *SIAM J. Matrix Analysis*, 2002.

[55] T. Molet, R. Boulic, and D. Thalman. A real time anatomical converter for human motion capture. *Eurographics International Workshop on Animations and Simulations*, 1996.

[56] M. Muller, T. Roder, and M Clausen. Efficient content-based retrieval of motion capture data. *Proceedings of ACM SIGGRAPH 2005*, 2005.

[57] K. Perlin. Real time repsonsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics*, 1(1), 1995.

[58] K. Perlin and A. Goldberg. Improv: A system for scripting interactive actors in virtual worlds. *Computer Graphics Proceedings (SIGGRAPH 1996)*, 1996.

[59] Z. Popovic and A. Witkin. Physically based motion transformation. *Computer Graphics Proceedings (SIGGRAPH 1999)*, 1999.

[60] K. Pullen and C. Bregler. Animating by multi-level sampling. *Proceedings of the Computer Animation*, 2000.

[61] K. Pullen and C. Bregler. Motion capture assisted animation: Texturing

and synthesis. *Computer Graphics Proceedings (SIGGRAPH 2002)*, 2002.

[62] B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, 1996.

[63] C. Rose, B. Bodenheimer, and M. Cohen. Verbs and adverbs: Multidimensional motion interpolation using radial basis functions. *IEEE Computer Graphics and Applications*, 1998.

[64] C. Rose, B. Guenter, B. Bodenheimer, and M. Cohen. Efficient generation of motion transitions using spacetime constraints. *Computer Graphics Proceedings (SIGGRAPH 1996)*, 1996.

[65] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[66] A. Safonova, J. K. Hodgins, and N. S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *Computer Graphics Proceedings (SIGGRAPH 2004)*, 2004.

[67] P. Sand, L. McMillan, and J. Popovic. Continuous capture of skin deformation. *Computer Graphics Proceedings (SIGGRAPH 2003)*, 2003.

[68] W. Shao and V. Ng-Thow-Hing. A general joint component framework for reatlistic articulation in human characters. *SIGGRAPH*, 2003.

[69] J. Tenenbaum and W. Freeman. Separating style and content with bilinear models. *Neural Computation*, 2000.

[70] J. B. Tenenbaum, V. deSilva, and D.C. Langford. A global geometric

framework for nonlinear dimensionality reduction. *Science*, 290(22), 2000.

[71] D. Thelen, F. Anderson, and S. Delp. Generating dynamic simulations of movements using computed muscle control. *Journal of Biomechanics*, 26, 2003.

[72] D. Tolani, A. Gosqanmi, and N. Badler. Real-time inverse kinematics techniques for antropomorphic limbs. *Graphical Models*, 2000.

[73] X. Tu and D. Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. *Computer Graphics Proceedings (SIGGRAPH 1994)*, 1994.

[74] M. Unuma, K. Anjyom, and R. Takeuchi. Fourier principles for emotion-based figure animation. *Computer Graphics Proceedings (SIGGRAPH 1995)*, 1995.

[75] R. Urtasun, D. Fleet, and P. Fua. 3d people tracking with gaussian dynamical models.

[76] G. Welch and G. Bishop. An introduction to the kalman filter. 2004.

[77] A. Witkin and Z. Popovic. Motion warping. *Computer Graphics Proceedings (SIGGRAPH 1995)*, 1995.

[78] V. Zordan, B. Celly, B. Chiu, and P. DiLorenzo. Breathe easy: Model and control of simulated respiration for animation. *Eurographics and ACM SIGGRAPH Symposium on Computer Animation*, 2004.