A COMPARISON AND EVALUATION OF MOTION INDEXING TECHNIQUES

by

HARNISH BHATIA

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2010

To my mom, dad, brother, friends and professors.

ACKNOWLEDGEMENTS

The successful completion of this thesis is due to the invaluable guidance, support and inspiration from a number of people.

I would like to express my profound gratitude to my advisor Dr.Gutemberg Guerra-Filho for his continuous support, encouragement, supervision and valuable suggestions throughout this research work. His direction and guidance played an important role in the successful completion of this thesis.

I would like to thank CMU Graphics Lab Motion Capture database for the motion capture data files which I used for testing my experiments. The database was created with funding from NSF EIA-0196217.

I would like to sincerely thank Dr. Manfred Huber and Dr. Jean Gao for taking interest in my work and agreeing to be on my thesis defense committee.

I would like to thank Dr. Farhad Kamangar for being an excellent teacher for the Computer Graphics course which was very important in the completion of this thesis.

I would like to thank my friend Rohit Rawat for all the help he provided towards successfully completing my thesis work.

And most importantly, I would like to thank my family for always giving me all the support and motivation I needed towards pursuing my interests and achieving excellence in all my work including this thesis.

May 17, 2010

ABSTRACT

A COMPARISON AND EVALUATION OF MOTION INDEXING TECHNIQUES

HARNISH BHATIA, M.S.

The University of Texas at Arlington, 2010

Supervising Professor: Gutemberg Guerra Filho

Motion capture (mocap) is a way to digitally represent the spatio-temporal structure of human movement. Human motion is generally captured in long sequences to record the natural and complex aspects of human actions, its sequential transitions, and simultaneous combinations. As the amount of mocap data increases, it becomes important to index the data in order to improve the performance of information access. The motion indexing problem involves several challenging issues due to the data being high dimensional, continuous, and time-variant. Indexing provides the means to search for similar motion in a mocap database, to recognize a query action as one among several motion classes, and fosters the reusability of motion data to generate animation automatically.

The topic of my research is the design, implementation, and evaluation of several approaches to the motion indexing problem. We consider three different existing types of whole-body motion indexing algorithms: dimensionality reduction based techniques, feature function based techniques, and dynamic time warping based techniques. The advantages and disadvantages of these techniques are explored. We evaluate the performance of each technique using a subset of the CMU Motion Cap-

ture Database and its corresponding annotation. These experimental results will allow for an objective comparison between the different indexing techniques and for assessing the deficiencies of whole-body indexing techniques.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Animation is a way to produce the notion of motion using a sequence of still forms. In general, digital animation is performed by rapidly displaying a sequence of images of 2-D or 3-D artwork or model positions, with a small change in them, which creates an optical illusion of motion. The computer animation can be categorized into 2-D and 3-D animation. 3-D computer animation can be of two types: Keyframing and Motion-based animation. Motion indexing is related to the reuse of motion capture data in the motion-based animation domain. A brief survey on animation can be found in Appendix A.

A 'White Paper' on motion Capture by Scott Dyer, Jeff Martin, and John Zulauf [1] explains that motion capture "involves measuring an object's position and orientation in physical space, then recording that information in a computer-usable form. Objects of interest include human and non-human bodies, facial expressions, camera or light positions, and other elements in a scene." Motion capture (mocap) is a way of digitizing the motion which can be useful in the fields of music, fine art dance/performance, sign language, gesture recognition, rehabilitation/medicine, biomechanics, special effects for live-action films, and computer animation of all types, as well as in defense and athletic analysis training. Motion based animation is produced by mapping this captured motion into the animated character's motion. The brief history, procedure of motion-based animation, and different motion capture technologies are explained in Appendix B.

Motion data indexing can be defined as an efficient way of identifying and retrieving similar motions from a large set of motions. Indexing can be performed based on the semantic and symbolic information linked to the data or solely based on the motion content. We describe five different techniques to index the available dataset which use either numerical information or contextual information related to the motion content. The performance time and indexing accuracy are measured for all of these techniques and a comparison is provided for them.

Indexing is a way to make information easy to access. In motion capture data, indexing also aims to perform the data retrieval. The motion data is retrieved on the basis of context and content. Indexing of motion capture data is a challenging problem beacuse of its high-dimensionality and its relation to several other problems in motion-based animation. These problems are explained below:

- Retargeting: Retargeting can be stated as a method to adapt the motion captured from a subject into an animated character. For example, the motion of a tall person carrying a heavy box can be retargeted into a short character carrying a similar heavy box. The reuse of motion should be independent of the method used to capture the motion. Retargeting attempts to preserve as many as desirable properties of the original motion as possible. For the above example, retargeting should result the position of character's hands at a proper place to hold the heavy box. A retargeting technique performs well if it is able to preserve the important aspects of a motion by altering only less important aspects. Limitations to define high level qualities of motion mathematically make the adaptation of one motion into another difficult and less efficient. Also complex metrics of motion requires great amount of work even to identify the properties of motion. These issues make retargeting little trickier in pragmatic situations[2].

- Segmentation: The motion capture is usually performed such that several actions are collected in a single motion sequence. However, the motion data should be stored in small clips containing a single activity. These small clips are derived by segmenting a long sequence of motion on the basis of different behaviors [3]. The problem of segmenting the motion data into small segments with each segment representing a different action is known as segmentation. For example, a motion clip can have climb, walk, run and jump actions. The goal of segmentation is to segment the long sequence into small clips with climb, walk, jump, and run, respectively. This task can be done manually, but for very long sequences and large numbers of such examples, the job is tedious. Consequently, we need to automate the segmentation process.
- Generalization: Controlled human figure animation is done in three ways: procedural, simulated, and interpolated. Procedural animation derives the degrees of freedom using code fragments. Dynamically simulated figure animation uses controllers together with a simulated human to generate motion. Interpolated animation uses sets of example motion with an interpolation scheme to construct new motions [4]. Given a set of example motions, the motion generalization interpolates style, emotions, speed, or variability by smoothly blending multiple motions with kernel functions to produce a new motion. Most techniques parameterizes motion samples to a have control on kinematical, physical, or emotional attributes. The primary issues related to motion generalization are to provide a set of meaningful, high-level parameters, to maintain the aesthetic of the source motions in the interpolated motions, and to extrapolate motion. The generation of a composite motion with these methods often lose the feature of original data. These methods tend to generate a new motion

by blending plural motions on arbitrary parameters which can produce unreal motion.

- Splicing: Splicing [5] is another method to generate new motion by splicing together different body parts of two motions. Splicing the motion of the upper body and that of the lower body can generate a new motion without losing the realism of the original data. However, to splice the motion of the upper body and that of the lower body is possible under a condition that the motion of the lower body of the two actions are similar. One of the examples of splicing is to synthesize motion of a person carrying a heavy box while walking on a curved path. This is accomplished by splicing the upper body of a person carrying a heavy box with the lower body of a person walking along a curved path.
- Transitioning: Transitioning [6] is a technique in which a segment of motion, also known as transition, is used to seamlessly attach two motions to form a single and longer motion. Existing 3-D animation tools support for linear stream of animation of data. To create more interactive and user-driven animation, transitioning can be used. A set of high quality basic motions like walk, jump, kick can be considered and transitions can be created between them to generate animations of varying lengths. It is relatively easy to make high quality basic motions, but generating high quality transitions is relatively tough. Automatic motion transition generation uses a combination of spacetime constraints and inverse kinematic constraints to generate transitions between two motions.

The segmentation technique produces thousands and thousands of motion clips in the motion database. Long sequences of motion are segmented into small motion clips. Segmentation allows the reuse of certain actions of a long motion sequence into a new motion sequence. Reusability avoids the motion capture of actions each time we need those actions in a slightly different manner.

In several applications, like a modern sports game engine, these segmented sequences (thousands in number) are stored in a library called motion library. The access of motion clips through a brute force method is a time consuming and inefficient way. To solve this problem more efficiently, indexing or annotation of motion clips in the motion database is required. Indexing aids in the manipulation of storage of the motion clips such that finding instances of a given motion segment in the complete repository becomes quick. Indexing should reject most of the irrelevant motions in a large motion database for a motion query in real time while providing quick and efficient retrieval of all or almost all similar ones.

Industry standard for computer games, animated movies, and special effects is undoubtedly the data driven animation. Segmentation and indexing helps in use and reuse of huge collections of motion data. On the other hand, recognizing human actions in survelliance applications is used for various purposes like monitoring crowded urban environments and protecting critical assets. In the motion capture domain, motion indexing corresponds to action recognition.

The problem of classification of human activities can be referred as action recognition. Given a number of actions, action recognition consists of classifying a new action as one of these actions. The human tracking problem is addressed as the problem of whole body tracking. In 3D motion data tracking, motion recognition uses the recovered parameters such as joint coordinates and joint angles. Once motion capture is obtained, mathematical models can be constructed based on these parameters and classification of actions is performed [7].

1.1 Motion Representation

Motion is generally described mathematically without considering the physical forces underlying it. This is also known as Kinematics and provides descriptions of

position, velocity, acceleration, and their rotational counterparts: orientation, angular velocity, and angular acceleration.

Two useful kinematic analysis tools are forward kinematics and inverse kinematics. Forward kinematics is a process of computing world coordinates of joints based on specified DOF values. The position and orientation of the subject are calculated from information of joints at a specified time . Inverse kinematics is a process of computing a set of DOF values from the position of a joint at a desired world space. The representation for joints (position and orientation) plays an important role in these kinematics [8].

A skeleton can be described as a framework of bones connected by articulated joints. The relative movement within the skeleton is provided by these joints which are represented mathematically by 4x4 linear transformation matrices. Different varieties of joints are constructed by combining rotations, translations, scales, and shears with these matrices. Most joints in the skeleton are associated with rotational data. There are a number of ways of representing rotations: matrices, quaternions, and Euler angles.

Matrices are the most accurate and least limited way of representing rotations. Mathematically, a matrix is a grid of numbers. When these numbers are applied to another matrix, or number, or a point, in the correct order, they can modify the values of what they are being applied to. For example, a rotation matrix can place a point in a new position by multiplying the point with itself which yields rotation of that point.

Euler angles represent the spatial orientation of any coordinate system as a composition of rotations from a reference coordinate system. An Euler angle is one of the three parameters required to describe such an orientation. In other words, any orientation can be achieved by combining three elemental rotations. Euler angles

provide fast interpolation but do not follow the great arc between the rotations and may result in wild swings about the canonical axes. The order of rotation is important when we combine rotations. Hence, there are many types of Euler angles depending on the order of rotations applied. Euler angles suffer from the problem of Gimbal lock. A Gimbal lock is the phenomenon of two rotational axis of an object pointing in the same direction which causes loss of one degree of freedom.

Rotational joints can also be implemented with quaternions. In the field of mathematics, quaternions are a number system that extends the complex numbers. They were described by an Irish Mathematician, Sir William Rowan Hamilton. Quaternions are used in theoretical and applied mathematics. A quaternion is a vector in 4D space that can be used to define a 3D rigid body orientation [9]. One of the notable characteristics about quaternions is that the product of two quaternions in not commutative. In other words, the product of two quaternions depends on what factors are present to the right and left of the multiplication sign. Hamilton defines a quaternion to be the quotient of two directed lines in a 3-D space or as the quotient of two vectors. In fact, they are the first noncommutative division algebra to be discovered. Some advantages of quaternions are: concatenating rotations is faster and more stable, extracting the angle and axis of rotation is simpler, and motion interpolation is easier.

Cartesian Coordinates are used to represent each point in a plane uniquely using a pair of numerical coordinates. These coordinates are the signed distances from the point to coordinate axes. Same principle can be used to specify any point in 3-D space by three Cartesian coordinates and signed distances to three mutually perpendicular planes. In general, same principle can be applied to represent a point in n-dimensional space, with n mutually perpendicular hyperplanes. They provide a link between Euclidean geometry and algebra. All geometric shapes can be represented

using Cartesian equations.

Cartesian coordinates are useful in most of the disciplines involving geometry. It is the most common coordinate system used in the field of Computer Graphics. The Cartesian coordinates of a point are represented in parentheses and separated by commas. 3D Cartesian coordinates specify a precise location by using three coordinate values: x, y, z. In general, z-axis is vertical and pointing up, so that the x and y axes lie on an horizontal plane. They are useful in specifying the location of a joint in space.

1.2 Mocap file formats

These algorithms described in this thesis work on mocap data and hence an overview about Mocap data is provided. More specifically, the ASF/AMC file formats are used in our experiments. The details about ASF/AMC file format is also provided. Before listing the various file formats for Mocap data, we introduce, the common terminologies used (Meredith and Maddock, 2001) [**?**] in the Mocap field:

- Skeleton: The representation of the internal bone structure of the motion capture subject.
- Bones: The skeleton is composed of bones. To create the characters appearance, vertices of a mesh are attached to these bones. Individual translation and orientation changes on bones are applied during the animation.
- Channel or Degree of freedom: During the course of motion, each bone of a skeleton can have position, orientation and scale changes. The attributes that describe theses changes are known as channels. Animation can be obtained by changing the values in each channel over time.
- Animation Stream: The channels can be combined to form an animation stream. For the whole body, the number of channels varies in a stream.

- Frame: Each slice of time of a motion is known as a frame and depicts the body pose as the skeleton moves. A mocap data can have frame rates as high as 240 frames/sec and as low as 30 frames per second [10].

Table 1.1: List of existing file formats for Mocap data.

File Format	Brief Description
ASC	Introduced by Ascension.
AOA	ASCII file format created by Adaptative Optics.
ASK/SDL	Variant of BVH file format developed by Biovision.
ASF/AMC	Developed by Acclaim, a video game company. It has two files to describe motion: ASF for skeleton and AMC for motion.
BVA and BVH	Biovision Hierarchical Data or BVH was developed by a motion capture company called Biovision.
BRD	The format is uniquely used by the motion capture system Ascension Technology Flock of Birds developed by LambSoft.
C3D	Developed by The National Institutes of Health and mainly used in Biomechanics research projects.
CSM	An optical tracking format used by Character Studio for importing marker data.
DAT	Developed by Polhemous.
GMS	It is a low-level, binary, minimal, but generic, format for storing Gesture and Motion Signals in a flexible, organized, optimized way.
HTR and GTR	Developed by Motion Analysis, HTR stands for Hierarchical Translation Rotation and is used for skeleton. GTR (Global Translation Rotation) is a variant of HTR.
MNM	This format is used to rename the segments of BVH and markers of CSM to Autodesk 3D Studio Maxs convention.
TRC	Developed by Motion Analysis and used to store raw as well as output data.
V/VSK	V is a binary motion data format developed by Vicon Motion Systems and is used with VSK file. VSK has skeleton hierarchy.

A list of existing file formats for Mocap data are described in the Table 1.1. We concentrate on the description of the ASF/AMC file format. The Acclaim Format file is made of two files. The ASF file contains the information about the bones and the skeleton hierarchy. The AMC file contains the motion related information. Two

separate files to describe motion is beneficial in a way that we can have multiple AMC files for a single ASF file in a single motion capture session.

The Acclaim Skeleton Format (ASF) has an extension of .asf and describes the joints and the skeleton hierarchy. In this file format, lines beginning with a hash sign (#) are comments. The .asf file is divided into sections and each section starts with a keyword. All keywords in the file start with a colon character ":". The ":version" keyword describes the version of the skeleton definition. The ":name" keyword denotes the name of the skeleton to be used. The ":units" keyword describes a definition for all values and units of measure used. The ":documentation" section has documentation information which remains from one file creation to the next. The ":root" section describes the parent of the hierarchy. The ":bonedata" keyword starts a block that describes all of the remaining bones in the hierarchy.

Each bone is enclosed within a "begin" and "end" pair of keywords and has the following information:

- "id": A unique ID number for each segment.
- "name": The unique name to each segment. Figure 1.1 shows the bone names and ids for a skeleton.
- "direction": The direction of the bone which explains how to draw the segment with respect to the parent segment.
- "length": The length of the bone which, combined with direction, gives the offset of the bone.
- "axis": The global orientation of an axis vector. The token letters "xyz" describe the order of rotations.
- "dof": The possible degrees of freedom in the bone. The possible values are tx, ty, tz, rx, ry, rz, and l. Each of these tokens are channels appearing in the

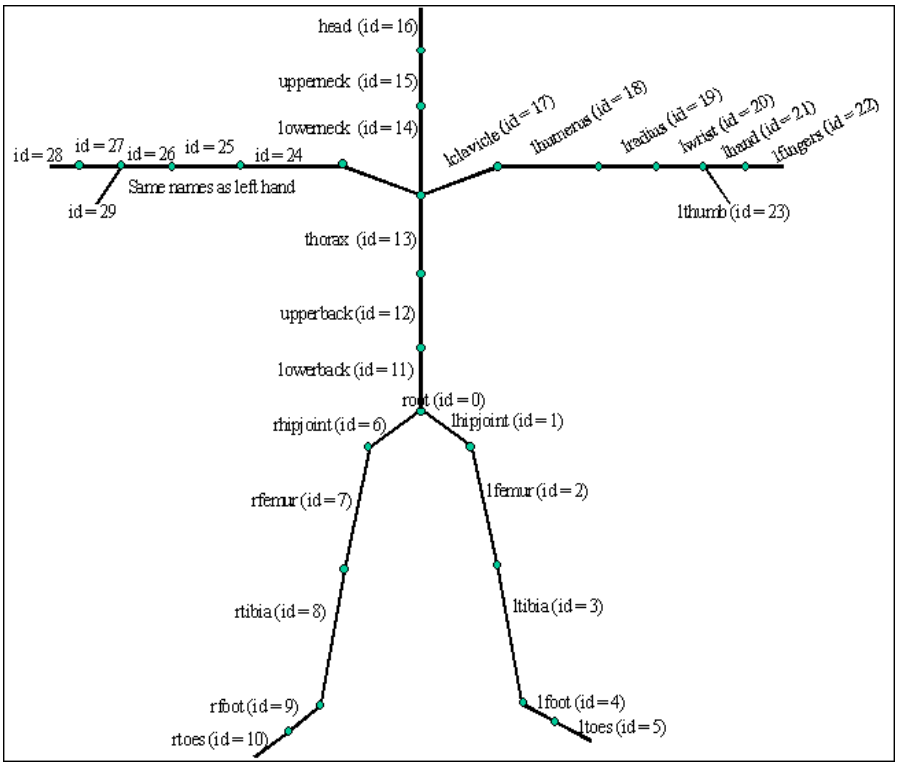


Figure 1.1: Skeleton with bone names and id.

AMC file. The order in which tokens are specified determines the order of the channels.

- "limits": The limits of the degrees of freedom by providing the minimum and maximum allowed value for that channel [11].

The ":hierarchy" section is the next section and describes the hierarchy of the bones declared in the ":bonedata" section. Figure 1.2 shows the hierarchy in an ASF file. The ":hierarchy" section is also enclosed within a "begin" and "end" pair of keywords in which each line contains the parent bone followed by its children. This information helps in connecting the bones together in the proper hierarchy. A snapshot of a small portion of ASF file is in Figure 1.3.

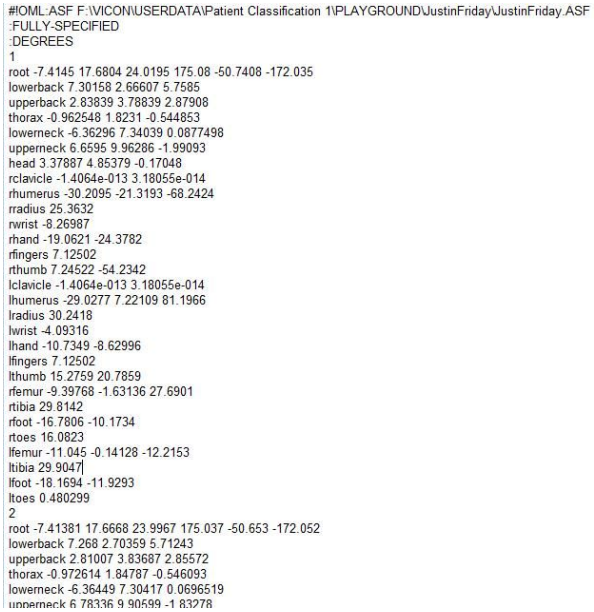
The Acclaim Motion Capture Format (AMC) file contains the motion data for a skeleton defined by .asf file. The motion data is given one sample at a time. This file

```
  begin
    id 30
    name rthumb
    direction -0.707107 -6.34907e-011 0.707107
    length 0.845506
    axis -90 -45 -2.85299e-015   XYZ
   dof rx rz
   limits (-45.0 45.0)
          (-45.0 45.0)
  end
 :hierarchy
  begin
    root lhipjoint rhipjoint lowerback
    lhipjoint lfemur
    lfemur ltibia
    ltibia lfoot
    lfoot ltoes
    rhipjoint rfemur
    rfemur rtibia
    rtibia rfoot
    rfoot rtoes
    lowerback upperback
    upperback thorax
    thorax lowerneck lclavicle rclavicle
    lowerneck upperneck
    upperneck head
    lclavicle lhumerus
    lhumerus lradius
    lradius lwrist
    lwrist lhand lthumb
    lhand lfingers
    rclavicle rhumerus
    rhumerus rradius
    rradius rwrist
    rwrist rhand rthumb
    rhand rfingers
  end
```

Figure 1.2: Hierarchy in a sample ASF File.

```
# AST/ASF file generated using VICON BodyLanguage
# ----------------------------------------------
:version 1.10
:name VICON
:units
  mass 1.0
  length 0.45
  angle deg
:documentation
   .ast/.asf automatically generated from VICON data using
   VICON BodyBuilder and BodyLanguage model FoxedUp or BRILLIANT.MOD
:root
   order TX TY TZ RX RY RZ
   axis XYZ
   position 0 0 0
   orientation 0 0 0
:bonedata
  begin
    id 1
    name lhipjoint
    direction 0.566809 -0.746272 0.349008
    length 2.40479
    axis 0 0 0  XYZ
  end
  begin
    id 2
    name lfemur
    direction 0.34202 -0.939693 0
    length 7.1578
    axis 0 0 20  XYZ
   dof rx ry rz
   limits (-160.0 20.0)
          (-70.0 70.0)
          (-60.0 70.0)
  end
  begin
    id 3
    name ltibia
    direction 0.34202 -0.939693 0
```

Figure 1.3: Part of a sample ASF File.



```
#!OML:ASF F:\VICON\USERDATA\Patient Classification 1\PLAYGROUND\JustinFriday\JustinFriday.ASF
:FULLY-SPECIFIED
:DEGREES
1
root -7.4145 17.6804 24.0195 175.08 -50.7408 -172.035
lowerback 7.30158 2.66607 5.7585
upperback 2.83839 3.78839 2.87908
thorax -0.962548 1.8231 -0.544853
lowerneck -6.36296 7.34039 0.0877498
upperneck 6.6595 9.96286 -1.99093
head 3.37887 4.85379 -0.17048
rclavicle -1.4064e-013 3.18055e-014
rhumerus -30.2095 -21.3193 -68.2424
rradius 25.3632
rwrist -8.26987
rhand -19.0621 -24.3782
rfingers 7.12502
rthumb 7.24522 -54.2342
lclavicle -1.4064e-013 3.18055e-014
lhumerus -29.0277 7.22109 81.1966
lradius 30.2418
lwrist -4.09316
lhand -10.7349 -8.62996
lfingers 7.12502
lthumb 15.2759 20.7859
rfemur -9.39768 -1.63136 27.6901
rtibia 29.8142
rfoot -16.7806 -10.1734
rtoes 16.0823
lfemur -11.045 -0.14128 -12.2153
ltibia 29.9047
lfoot -18.1694 -11.9293
ltoes 0.480299
2
root -7.41381 17.6668 23.9967 175.037 -50.653 -172.052
lowerback 7.268 2.70359 5.71243
upperback 2.81007 3.83687 2.85572
thorax -0.972614 1.84787 -0.546093
lowerneck -6.36449 7.30417 0.0696519
upperneck 6.78336 9.90599 -1.83278
```

Figure 1.4: Snapshot of a sample AMC File.

has an extension of an .amc and defines the actual channels of the animation. Each frame starts with a line declaring the frame number. The next line contains the data for the bone animation comprised of the bone name and the data for each channel (defined in the .asf file) of that bone. The frame sections in the file continue until the end of the file[11]. A snapshot of a small portion of AMC file is in Figure 1.4.

This thesis concentrates on indexing of motion data which is one of the issues related to motion-based animation. The chapter 1 of this thesis describes motion indexing, motion representation, and motion file formats. The chapter 2 describes the two different PCA-based techniques for motion indexing and also explains them from an implementation point of view. The chapter 3 describes the idea and implementation details of a geometric feature-based motion indexing technique. Dynamic time warping based indexing techniques are described in the chapter 4. The chapter 5 of the thesis describes the two types of performance evaluation: Time performance and

indexing accuracy of these five techniques. Conculsions and future work are discussed in chapter 6.

CHAPTER 2

PCA-BASED TECHNIQUES

In this section, the overview of motion segmentation and PCA is provided and then the two techniques are described which use SVD as a tool to analyze the data and segment it. These two techniques are refered using the research paper, Segmenting motion capture data into distinct behaviors[3]. Both of these techniques traverse the data creating segments of data. The motion is considered as a sequence of poses. This sequence is segmented if there is a change in the distribution of data locally. Ideally, high level behaviors, which are different verbs, are segmented out like walk, climb and, jump. The first method uses Principal Component Analysis (PCA) to segment the motion on the basis of the dimensionality change in local motion. The second technique uses Probabilistic PCA to create a probabilistic distribution model of a motion and segments the data based on this model. We also propose a novel technique to perform indexing on the segmented data.

Motion Segmentation is a way of decomposing a long sequence of motion into smaller subsequences based on different behaviors. The collection of long sequences during motion capture and the posterior segmentation is an efficient way to capture natural and realistic actions. Segmentation of motion data can be performed during the capture or manually after the data is captured. As captured sequences become longer in one session, the manual segmentation becomes time consuming. On the other hand, automatic segmentation takes into account statistical properties of the data to segment the motion automatically. Usually, a segmentation method builds a mathematical model to analyze these statistical properties.

Segmentation of motion data is followed by indexing of data, query retrieval, and other processing. The indexing and retrieval of motion data can be described as a process of identification and extraction of given motion clips. In general, indexing is based on user semantics and content of the data. Indexing and segmentation based on data content is an important problem. Different techniques and mathematical models like motion graphs, Hidden Markov Models, Support Vector Machines were used for recognition and processing of time series data. Among these techniques, we have chosen three different techniques for motion data indexing which are described below

The Principal Component Analysis (PCA) and other dimensionality reduction techniques are used in modeling and other processing of high dimensional data by researchers in the field of computer graphics, robotics, computer vision, machine learning, and biomechanics. As stated in the book "Principal Component Analysis" by I.T.Jollife, "The central idea of PCA analysis is to reduce the dimensionality of a data set consisting of large number of interrelated variable, while retaining as much as possible of the variation present in the data"[12]. The PCA technique transforms the existing datasets into a new set of variables which are also known as principal components. These components are uncorrelated and ordered such that the first few components contain most of the variation of the original data set. If x is a vector in a p dimensional space and if p is small or correlations between these p variables are highly related, then x does not need to be described by all the p variables. Fewer derived variables ($<< p$) are enough to preserve most of the variance of the information.

The first step in a PCA analysis is to have a linear function $\alpha_1' x$ of the elements of x having maximum variance where α_1' is a vector with p components:

$$\alpha\prime_1 x = \alpha_{11}x_1 + \alpha_{12}x_2 + \ldots\ldots + \alpha_{1p}x_p = \sum_{j=1}^{p} \alpha_{1j}x_j \tag{2.1}$$

Next stage involves finding $\alpha_2' x$ with maximum variance and uncorrelated with $\alpha_1' x$. So after kth stage there will be $\alpha_1' x$, $\alpha_k' x$ variables with maximum variance and no correlation between them. Thus, the dimensionality of the original vector is reduced to k ($k << p$)while preserving most of the information[13]. Segmentation based on PCA reduces the number of dimensions in the working data. PCA uses Singular Value Decomposition of the data.

PCA-model based segmentation has also been widely used for recognition, tracking, and segmentation of high-level behaviors. However, some of the methods like the one proposed by Arikan, Forsyth, and O'Brien[14], depends on the training data as well. The PCA based segmentation uses higher level actions as a basis to segment data whereas most of the techniques available for segmenting time series data like a video or audio perform segmentation on the basis numerical information.

2.1 Implementation

We implement two techniques based on PCA for motion indexing. The first technique is based solely on SVD and the second technique uses a variation of PCA called Probabilistic PCA (PPCA) to estimate the distribution of the motion data. PPCA is an extension of the traditional PCA. PPCA performs dimensionality reduction and also can be utilized as a general Gaussian density model. In PPCA, maximum-likelihood estimates for similar frames associated with the covariance matrix can be efficiently computed from the principal components. The PCA method

discards the singular values which contains less information, whereas in PPCA the singular values are modeled with noise.

The segmentation problem can be mathematically described as: Given a long sequence of motion M, segment the motion M into distinct behaviors $M_1, M_2, \cdots\cdots, M_s$. The temporal boundaries of the behaviors and the number s of behaviors are determined as a part of the segmentataion of M. A motion sequence can have segments with same behavior repeated at different times. For example, a motion M can have walk, climb, run and then walk again. In this case, a segmentation technique should result in four segments $\{M_1 = walk, M_2 = climb, M_3 = run, M_4 = walk\}$.

A motion is represented with sequences of frames and frame rate can be 60 frames/sec, 120 frames/sec or 240 frames/sec. A frame is described by rotations of all the joints in the skeleton at a particular time. Absolute body position and body orientation is not considered in both PCA-based techniques. Quaternions are used to represent rotations of the joints.

The whole motion forms a trajectory in $D = J \times 4$ dimensional space, where J is the number of joints in a body and each joint requires one quaternion for its representation. A quaternion is treated as a vector of length 4. Each frame x_i $(i = 1, 2, \cdots, n)$ is represented as a point in D dimensional space, where n is the total number of frames in the motion.

The center of the motion trajectory is given by the mean of the motion sequence:

$$\bar{x} = \frac{1}{n} \cdot \sum_{i=0}^{n} x_i. \tag{2.2}$$

Simple motions have frames which are clustered around the mean and, after PCA, require very few dimensions for their representation. This happens because correlations exist between similar motions. So, more similar the motion is, less are the number

of dimensions needed to represent the motion. This fact forms the basis to use the PCA method for the analysis of data.

2.1.1 SVD Analysis

Motion data for a simple motion can be modeled around the center of motion using r dimensions as $x_i' = \bar{x} + \alpha_{i1}v_1 + \alpha_{i2}v_2 + \ldots\ldots + \alpha_{ir}v_r$, where the vectors $v_1, v_2 \ldots\ldots, v_r$ form the axes of a hyperplane and $\alpha_{i1}, \alpha_{i2} \ldots\ldots, \alpha_{ir}$ are the coefficients specific to that particular frame. Basically, when we project a frame of dimension D to a hyperplane of dimension r, we keep only r dimensions of that frame and discard the other $D - r$ dimensions.

In order to obtain this dimensionality reduction, PCA is performed on the whole motion to find the reduced number of dimensions, r. The PCA is done using Singular Value Decomposition of the motion data. Before performing the SVD on the data, the center of motion (mean) is subtracted from the data. These frames are arranged in a matrix Q of size $n \times D$ containing quaternion values, where n is the total number of frames and D is the number of dimensions as described above. After applying SVD on this matrix, three separate matrices U, V and Σ are obtained such that $Q = U\Sigma V^T$, where the U and V matrices have orthogonal unit vectors as columns and the matrix Σ is a diagonal square matrix with its diagonal values σ_i being decreasing, non-negative, and singular. The first r columns of V are the basis $v_1, v_2 \ldots\ldots, v_r$ of the optimal hyperplane of dimension r. These r values become the principal components of the motion and the motion dimension changes from D to r. When the dimensionality of a motion is reduced, there is some projection error introduced in the reduced motion which is given as:

$$e = \sum_{i=1}^{n} \|x_i - x_i'\|^2. \tag{2.3}$$

2.1.2 Calculation of the Reduced Dimension

It is important to determine the lowest number of dimensions, r, needed to represent each frame of motion. This number plays an important role in determining the amount of information that needs to be preserved. If r is chosen small, more information is lost and error of projection is increased. If r is chosen big, the motion will have some unnecessary dimensions and computation of data becomes expensive. The preserved information ratio, E_r is given by:

$$E_r = \frac{\sum_{j=1}^{r} \sigma_j^2}{\sum_{j=1}^{D} \sigma_j^2} \tag{2.4}$$

The ratio E_r defines the amount of information which is preserved if r dimensions are preserved and the rest $D - r$ dimensions are discarded. The method to determine r starts by assuming the smallest value of r, say 1. The method increments the value of r until E_r is greater than some threshold τ, where τ is less than 1. Once the value of the E_r exceeds the threshold, we stop iterating through r. Therefore, given an appropriate threshold τ, the chosen value of r should be able to preserve the features necessary for a human to tell one motion apart from other motion and should introduce very little change to the appearance of the motion.

2.1.3 PCA-SVD Technique

This technique finds the segments in the motion sequence on the basis of the change in the intrinsic dimensionality of the motion. As the transition from one action to another occur in a motion sequence, the number of dimensions required to preserve the information with small projection error increases. This can be stated in a different way as: When there is a change from one behavior to another in a motion and if the number of dimensions is fixed, the projection error increases.

The PCA-SVD algorithm finds the value of r. To determine the initial r, only the first k frames of motion are used ($k = 240$). The value of r should be such that it sufficiently represents these fixed k frames and the ratio of preserved information E_r is greater than the threshold τ. The threshold τ is set to 0.9 for this technique.

Once the value of r is determined, the next step is to calculate the total projection error e for the k frames given by the equation (2.3). After the initialization is performed for the first k frames, the algorithm computes the SVD of the first i frames ($i \geq k$), where i is steadly increasing by one at each iteration. The total projection error e_i is obtained at each iteration using a fixed reduced dimension r. For a simple motion the error e_i rises at a constant slope. This is because the hyperplane remains the same for a single simple motion and new frames for the same motion introduce almost the same projection error. However, when frames corresponding to a new behavior appear, the error e_i starts to rise quickly. Hence to detect the change in motion, the error derivative is calculated $d_i = e_i - e_{i-l}$, where the parameter l is introduced such that there is enough gap between frames to avoid noise artifacts. The value of the parameter l is chosen as 60. The initial value for i is equal to k. To avoid the initial oscillation of the average due to a small number of frames, the calculation of error derivative, d_i is done only after i becomes larger than a specified parameter i_0 (i_0 is defined as $i_0 = k + l \ = 300$).

The derivative d_i remains more or less constant for the same motion with little oscillations due to noise. However, when a new motion is introduced, the derivative d_i rises sharply above the previous constant value. Hence, at any point, a possible segment boundary can be checked by computing the average and standard deviation of all error derivatives for the previous data points d_j, where j is less than i. If the current error derivative d_i is more than k_σ times the standard deviation from the average ($k_\sigma = 3$), a segment boundary is created at that frame.

Once the boundary is found, the algorithm is restarted with motion containing only the frames starting at the boundary frame till the end. Again, all the steps starting from the calculation of r are followed. The algorithm finds all the possible boundaries in the motion starting from the first frame to the last frame and outputs the boundaries of the segments in terms of frame numbers.

2.1.4 PPCA Technique

This algorithm starts with defining the mean $\bar{x}$ and the covariance C for the Gaussian distribution model using the first k frames of the motion. The intrinsic dimensionality calculation is done for these k frames and the value of r is calculated as in the SVD-PCA technique. For the PPCA technique, the value of τ is 0.95. This approach provides an accurate model of a particular behavior because it captures the correlation in the motion of different joint angles as well as the variance of all joint angles. The SVD calculation for the first k frames is done with the value of k set to 150 yielding matrices U, V and Σ.

The mean $\bar{x}$ of the distribution is equal to the center of motion, which is defined in equation (2.2) for the SVD calculation. The covariance C is based on the estimated intrinsic dimensionality of the motion. To determine C, first the averaged square of discarded singular values σ is calculated:

$$\sigma^2 = \frac{1}{D-r} \cdot \sum_{i=r+1}^{D} \sigma_i^2, \tag{2.5}$$

where $D = 4J$ and J is the number of joints in the body.

The covariance matrix C is calculated by the following equations:

$$W = V_r \cdot \left({\Sigma_r}^2 - \sigma^2 I\right)^{\frac{1}{2}} \tag{2.6}$$

and

$$C = \frac{1}{n-1} \cdot \left(WW^T + \sigma^2 I\right) = \frac{1}{n-1} \cdot V\tilde{\Sigma}^2 V^T, \tag{2.7}$$

where V_r is a matrix with the first r columns of matrix V and Σ_r is the upper left $r \times r$ block of matrix Σ. The matrix $\tilde{\Sigma}$ is a $D \times D$ diagonal matrix, obtained fromΣ with replacing all discarded singular values with σ. I is the identity matrix and n is total number of frames.

Once the value of mean $\bar{x}$ and covariance C are calculated, we estimate the likelihood of the frames $k+1$ through $k+T$ having the Gaussian distribution defined by the mean and covariance, where The value of T and K are set same and is equal to 150 frames. This estimation is done by computing the Mahalonobis distance H for frames $k+1$ through $k+T$ using the equation below :

$$H = \frac{1}{T} \cdot \sum_{i=k+1}^{k+T} (x_i - \bar{x_i})^T C^{-1} (x_i - \bar{x_i}) . \tag{2.8}$$

The Mahalanobis measurement is independent of data because distances are calculated in units of standard deviation from the mean and are therefore good to discriminate frames. A reasonable estimate of T is half of the anticipated number of frames in the smallest behavior in the database.

The average Mahalanobis distance, H, is then computed iteratively by increasing k by a small number of frames, $\triangle$, and repeating the estimation of distribution for the first k frames. This time the value of k is increased by $\triangle$ and H also corresponds to a new distribution. The value of $\triangle$ is set to 10 frames.

The Mahalanobis distance, H, has some characterstic patterns. When the motion frames corresponding to a similar action appear, the Mahalanobis distance decreases gradually and reaches a valley point (local minima). However, when the frames of a different behavior appear, there is an increase in the value of H. The subsequent decrease in H begins when the frames of the new behavior start appearing and as a result, the distribution begins to accommodate the new behavior. Thus the segmentation takes place when H forms a peak. The algorithm declares a segment

boundary when a valley in H is followed by a peak. We also take into account the difference between the valley and the peak to avoid false segments. This difference should be greater than some threshold R. The value of the threshold is chosen by the following equation (2.9):

$$R = \frac{H_{max} - H_{min}}{10}, \tag{2.9}$$

where H_{max} and H_{min} are the maximum and minimum possible values of H. If the value of R is increased, it results in fewer segments that correspond to more distinct behaviors. Whereas if R is decreased, finer segmentation happens.

To find the next boundary, the whole algorithm is restarted on the rest of the motion. The output of this technique is the same as the first one: The segment boundaries in terms of their frame numbers.

2.1.5 Indexing of Segments

Segments are obtained as a result of the above segmentation techniques for a motion sequence. The indexing of two motion sequences involves the indexing of their respective segments. Therefore, indexing uses the information retrieved during the PCA analysis step of segmentation. When all segments are derived, we store the dimension r, the start frame, and the end frame for each segment. Our algorithm for indexing can be described as:

1. Combine two segments from different motions by concatenating the second segment to the first segment. Suppose the first segment is a $m \times D$ matrix and the second segment is a $n \times D$ matrix, the combined segment becomes a $(m+n) \times D$ matrix. Then use the r_1 from the first segment to calculate the preserved information , E_{r_1}, for the combined segment.

2. Similarly combine the two segments by concatenating the first segment to the end of second segment. Then use the r_2 from second segment to calculate the preserved information, E_{r_2}, for the combined segment.
3. If E_{r_1} is equal to E_{r_2} and both E_{r_1} and E_{r_2} are greater than the threshold τ, then it is a match, otherwise, it is not a match.

CHAPTER 3

FEATURE-BASED TECHNIQUE

This chapter describes the overview and implementation of feature function based indexing technique. This method has been mentioned in the reserach paper, Efficient content-based retrieval of motion capture data[15] and the book, Information Retrieval for Music and Motion[16].

Although there is a massive increase in the use of motion capture data, there still is no efficient motion retrieval systems that allow identifying and extracting user-specified motions. Large databases containing time series data require efficient indexing and retrieval methods. Previously, retrieval systems were based on manually generated textual annotations, which roughly describe the motions in words. For large datasets, the manual generation of reliable and descriptive labels becomes infeasible. When the motion is identified and extracted on the basis of semantic description of the motion, it is known as content-based motion retrieval. However, the different motions are logically related and distributed randomly in the dataset. The motion specification can be done in query mode in which a user provides the description of the motion in the form of a small motion example. The important aspect in motion retrieval is the notion of similarity used to compare the query with the motions in the database. Two motions may be regarded as similar if they represent variations of the same action or sequence of actions. These variations may concern the spatial as well as the temporal domain.

In general, similar motions need not be numerically similar. Motion capture data has a richer semantic content because the position and the orientation of all

joints are known for every pose. Hence, the feature-based technique uses various kinds of qualitative features describing geometric relations between specified body points for each pose. However, the use of combinatorial, relational or qualitative attributes in place of quantitative or metrical attributes has been done before in other domains such as visual object recognition in 2-D and 3-D, action recognition and tracking [17].

Indexing techniques based on relational features have logical similarity and hence are well suited for indexing. A feature-based technique allows flexible and efficient content-based retrieval and browsing in large motion capture databases.

3.1 Implementation

In this technique, we introduce relational features that describe geometric relations between specified joints of the body or short sequences of poses. The several relational features are Boolean in nature and encode spatial, velocity-based, as well as directional information. This binary sequence is invariant under spatial as well as temporal deformations and hence good to perform motion comparisons during retrieval.

Geometric boolean relational feature functions are defined which express geometric relations between certain body joints. A subset or all of these features can be applied to a motion sequence depending on the users requirement. A set of feature functions, when applied to the motion sequence, produce a binary version of the motion. This binary sequence is inherently segmented on the basis of zeros and ones. The segments inherit the semantic qualities of the underlying features and are also robust to the local time variations. This type of segmentation becomes adaptive by having different combinations of feature functions as required. To perform similarity check on this binary motion stream, the standard information retrieval techniques

based on indexing with inverted lists are used. The feature vectors are used as index words and indexing is carried out at the segment level rather than at the frame level. This approach provides fast content-based and fault tolerant retrieval.

3.1.1 Notation for feature-based method

Intially, lets introduce some notations which are used in the feature-based algorithm:

1. Data Stream: A data stream is a sequence of frames and each frame specifies the 3D coordinates of the joints at a certain point in time. Hence, a motion sequence is represented by a 3-dimensional matrix where the dimensions represent time (frames), joints in the body, and the axes (x, y, z) for the joint position.
2. Pose: In a geometric context, a frame correspond to a pose. Mathematically, a pose can be described as $P \in \Re^{3J}$, where J is the number of joints in the body. The j-th column of P, denoted by P_j, corresponds to the 3-D coordinates of joint j.
3. Document: A motion capture data stream can be expressed as a function $\Gamma : [1 : T] \rightarrow \Re^{3J}$, where T denotes the number of poses. The document function Γ maps from the time domain to the set of poses.
4. Motion clip: A subsequence of consecutive frames.
5. Trajectory: The curve described by the 3D coordinates of a single body point.

3.1.2 Feature Functions

A Boolean feature which describes geometric relations can be described as a function. Mathematically, it is equivalent to a function $F : \Re^{3J} \rightarrow \{0, 1\}$, mapping the set of poses into $\{0, 1\}$. The Figure 3.1 shows the output of a boolean feature function which checks whether the right leg is ahead of the whole body for a walk

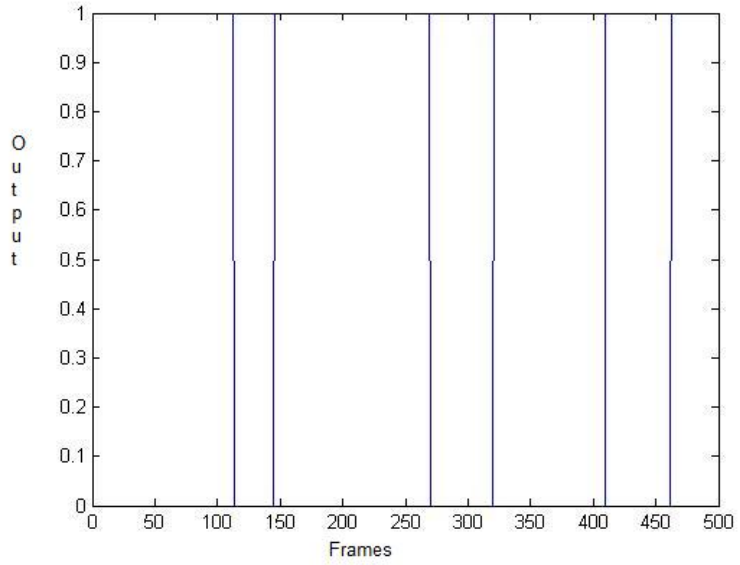


Figure 3.1: Result of boolean feature function.

motion.

Boolean functions for every pose can be combined to form any boolean expression which itself is boolean in nature. For example, the conjunction and the disjunction of boolean functions F_1 and F_2 are boolean expressions. Using f boolean functions for some $f \geq 1$, a combined function $F : \wp \rightarrow \{0,1\}^f$ is obtained. This F is called a feature function and the vector $F(P)$ is known as a feature vector or simply a feature of the pose $P \in \Re^{3J}$. The composition $F \circ \Gamma$ is a combination of feature functions for the motion data stream Γ.

Feature functions, which are defined by joint coordinates, are invariant under global transforms such as Euclidean transformations and scalings. These features are very coarse because they express only a single geometric aspect and mask all other aspects for a pose of a motion. This makes such features robust to variations in the motion capture data stream that are not correlated with the aspect of interest. This property

of feature functions is helpful on focusing on certain aspects of motion which are required by the application.

A generic relational feature is constructed such that it encodes certain joint constellations in 3-D space and time. This generic feature depends on a set of joint variables, denoted by $j_1, j_2, \cdots, j_s$ as well as on a variable θ for a threshold value or a threshold range, where s is the number of joints used in the feature. Feature functions are classified as follows:

1. Feature functions describing a plane.

   This category of generic functions are plane-based. The equation of the plane is defined by the 3-D coordinates of the joints $j_1, j_2, \cdots, j_s$. A plane function uses four joints to calculate its boolean value $F_{plane} = F_{\theta,plane}(j_1, j_2, j_3; j_4)$. The three plane-based generic functions which are used in the algorithm are defined as follows:

   - The plane is defined such that it passes through the joints j_1, j_2, j_3 and the function checks if the joint j_4 is on or infront of the oriented plane. It has value 1 if the joint j_4 is lying on the plane or infront of the plane and its value is 0 when joint j_4 is behind the plane.
   - The plane is defined such that it passes through the joints j_1, j_2, j_3 and the function checks if the joint j_4 has a signed distance greater than $\theta \in \Re$ from the oriented plane. It has value 1 if the signed distance is greter than θ, else its value is 0.
   - The plane is defined in terms of a normal vector given by joints j_1 and j_2 and is fixed at joint j_3. The function takes the value 1 if joint j_4 has a signed distance greater than $\theta \in \Re$ from the oriented plane, else takes the value 0.

2. Feature functions describing an angle.

   A segment is defined by two joints of the body and an angle is defined between two segments. Angle functions use four joints to calculate its boolean value $F_{angle} = F_{\theta,angle}(j_1, j_2; j_3, j_4)$. The algorithm uses only one angle-based function, two directed segments are defined between the two pair of joints (j_1, j_2) and (j_3, j_4). An angle is determined between these two segments. The function assumes the value 1 if this angle is within the threshold range $\theta \subset \Re$, else it has value 0.

3. Feature functions describing velocity

   These feature functions operate on velocity data that is approximated from the 3-D joint trajectories of the input motion. The velocity based functions calculate the velocity of a joint. The velocity of a joint is calcluated for each frame by taking a few previous frames and a few subsequent frames. Thus a window is created for the frames with the current frame as the center. The velocity is calculated by taking the difference of corresponding frames: one from left of the center frame and other from the right of the center frame. The function takes the value 1 if the magnitude of this velocity is greater than the threshold, else it is 0.

   The feature-based algorithm uses two velocity functions, this generic velocity feature function, $F_{fast} = F_{\theta,fast}(j_1)$ is defined for one joint j_1 and calculates the velocity of that joint. The function assumes the value 1 if joint j_1 has an absolute velocity above θ, else it has value 0. Another velocity-based function $F_{move} = F_{\theta,move}(j_1, j_2, j_3; j_4)$, calculates the velocity of joint j_3 relative to joint j_1. The velocity is projected in the direction given by the normal vector of the oriented plane spanned by joints j_1, j_2, and j_3. The function takes the value 1 if this velocity is greater than the threshold θ, and the value 0 otherwise.

Basically, all of the relational features defined above assume the feature value zero corresponds to a neutral, standing pose.

3.1.3 Threshold Selection

When the relational features are designed, it becomes very important to have an appropriate threshold parameter θ for each feature function which is semantically meaningful. The specific choice of a threshold has a strong influence on the semantics of the resulting relational feature. If the threshold is increased, the feature functions will not be able to detect subtle movements. The threshold selection is purely dependent on the application in use. To make the relational features invariant under global scaling, the threshold parameter θ is specified relative to the respective skeleton size. Thresholds of most of the feature functions in the algorithm are expressed in terms of humerus length, and some in terms of hip width or shoulder width. By doing this, the threshold becomes independent of the skeleton size. Difference in the sizes of skeletons due to different actors can be handled by this approach.

3.1.4 Formation of Document

A set of 19 geometric feature functions are selected to create a structure, document. The description of these features is given in Table 3.1:

3.1.5 Adaptive Segmentation

A fixed combination of 19 geometric feature functions are applied to obtain the document Γ for each motion sequence. The document is a matrix of zeros and ones for all frames. Some of the consecutive frames in the motion sequence can yield the same feature vectors which are known as runs. This can be mathematically explained as: Given the fixed feature function $F : \Re^{3J} \rightarrow \{0,1\}^f$, then two poses $P_1, P_2 \in \Re^{3J}$ are

F-equivalent if the corresponding feature vectors $F(P_1)$ and $F(P_2)$ are the same. An F-run of Γ is defined to be a subsequence of Γ consisting of consecutive F-equivalent poses, whereas F-segments of Γ are defined to be the maximal F-runs.

The segments of a given motion stream are runs of maximal length. Each of these segments corresponds to a unique feature vector. A document has many segments and becomes a sequence of feature vectors. This sequence is referred as the F-feature sequence of document Γ and denoted by $F[\Gamma]$.

For example, consider the segmentation of the motion consisting of a right foot kick followed by a left hand punch. The first step is to create a feature function which is a combination of any number of available boolean generic functions. To describe this motion we use, a feature function $F^4 : \Re^{3J} \rightarrow \{0,1\}^4$ consisting of the following four boolean functions:

1. Angle-based function which checks whether the right knee is bent or stretched.
2. Angle-based function which checks whether the left elbow is bent or stretched.
3. Plane-based function to check whether the right foot is raised or not.
4. Plane-based function to check whether the left hand is reaching out to the front of the body or not.

The total number of different feature vectors possible for a feature function is equal to 2^f, where f is the number of boolean functions in the feature function. Hence, for the above feature function F^4 there are 16 different feature combinations possible and hence 16 feature vectors possible.

This type of feature-dependent segmentation provides for temporal invariance because the motion capture data streams are compared at the segment level rather than at the frame level. So the time factor denoted by frames is not considered. Mathematically stated: The sequence of F-segments of a motion stream Γ is segmented and each segment corresponds to a unique feature vector. These feature vectors are

known as the F-feature sequence of Γ and denote by $F[\Gamma]$. If $M+1$ is the number of F-segments of Γ and if $\Gamma(t_m)$ for $t_m \in [1:T]$, where $0 \leq m \geq M$, then $\Gamma(t_m)$ is a pose of the m-th segment and $F[\Gamma] = (F(\Gamma(t_0)), F(\Gamma(t_1)), ..., F(\Gamma(t_M)))$.

In the feature based segmentation, two motions that differ by some deformation of the time axis will yield the same F-feature sequences. This method is adaptive in a sense that the number of segments will be based on the selection of feature functions. Fine feature functions which consists of many components result segmentations with many short segments, whereas coarse features functions lead to a smaller number of long segments.

3.1.6 Indexing and Querying of Database

A database Γ consists of motion capture data streams. Indexing can be defined as the retrieval task which can identify motion clips contained in a specified query. It can be stated more precisely as: The database consists of a collection $\Gamma = (\Gamma_1, \Gamma_2, \cdots\cdots, \Gamma_I)$ of motion capture data streams $\Gamma_i, i \in [1:I]$, where I is the number of documents. To simplify things, it can be assumed that Γ consists of one large document by concatenating the documents $\Gamma_1, \Gamma_2, \cdots\cdots, \Gamma_I$, keeping track of document boundaries in a supplemental data structure. The query Q consists of an example motion clip. A feature function $F : \Re^{3J} \rightarrow \{0,1\}^f$ can be replaced by specifying the feature vectors obtained during segmentation and use the notation $F[\Gamma] = w = (w_0, w_1, \cdots\cdots, w_M)$ and $F[Q] = v = (\upsilon_0, \upsilon_1, \cdots\cdots, \upsilon_N)$ to denote the resulting F-feature sequences of Γ and Q, respectively, where M and N denotes the number of segment numbers for the respective two feature sequences.

3.1.6.1 Inverted File Index

Inverted file index is a type of indexing used in text-based retrieval systems. It is an index data structure storing a mapping from content to its locations in a database file, or in a document, or in a set of documents. It is also known as postings file or an inverted file. The database Γ is indexed with respect to F and inverted indices are used. For each feature vector $v \in \{0,1\}^f$, an inverted file index or inverted file, $L(v)$, is created. This file consists of all indices $m \in [0, M]$ of the sequence $w = (w_0, w_1, \cdots, w_M)$ with $v = w_m$. Hence, $L(v)$ gives information about the F-segments of Γ which exhibit the feature vector v. Figure 3.2 shows the flow of inverted list creation.

Inverted lists are sets represented as sorted, repetition-free sequences. The number of inverted lists for the feature vector depends on the number of boolean functions it is composed of. The vector $v \in \{0,1\}^f$ has f boolean functions and has 2^f inverted lists associated with it. An inverted file index of the database Γ consists of these 2^f inverted lists $L(v)$ for vector v and is denoted by I_F^Γ.
The inverted lists contain the position of the segments and the segment lengths of the motion file. Each segment position appears in exactly one inverted list, the size of the inverted file index is proportional to the number M of segments of Γ.

3.1.6.2 Algorithm for Feature-based Indexing

The indexing or matching of the similar motion clips depends on the definition of similarity. There can be different notions for declaring two motion clips as similar. Searching a similar motion can be as restrictive as finding motion clips which are exactly the same or it can be little less restrictive as fuzzy search which only checks

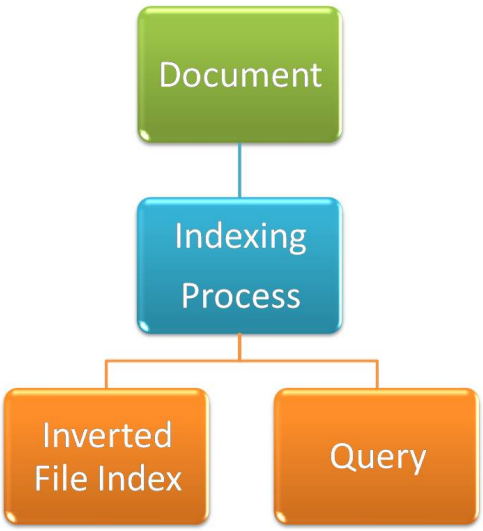


Figure 3.2: Process of Indexing Document.

for similarity between some parts of a motion clip. Figure 3.3 shows the process of finding exact hits.

The technique described here is based on an exact hit and tries to find the exact match of a motion. The two motion clips are considered similar with respect to the selected feature function if they exhibit the same feature sequence. Mathematically, if $w = (w_0, w_1, \cdots, w_M)$ and $v = (v_0, v_1, \cdots, v_N)$ are the feature sequences of the database and the query, respectively, then an exact hit is an element $k \in [0, M]$ such that v is a subsequence of consecutive feature vectors in w starting from index k. This is denoted with the expression vkw. The set of all exact hits in the database Γ is defined as: $H_\Gamma(v) := \{k \in [0, M] \,|vkw\}$. In terms of inverted lists, which are shifted and intersected, the set of all exact hits can be expressed as:

$$H_\Gamma(v) = \bigcap_{n \in [0:N]} (L(v_n) - n)\,. \tag{3.1}$$

The input lists $L(v_n)$ are additively adjusted by $-n$. The more efficient way is to adjust the lists appearing as intermediate results by $+1$ prior to each intersection step. After the final iteration, an adjustment of the resulting set by $-(N+1)$ yields

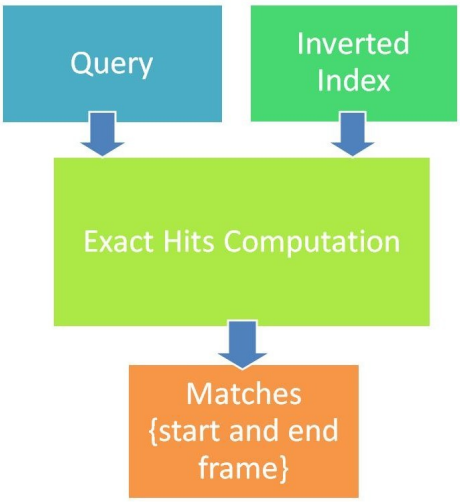


Figure 3.3: Process of computing Exact Hits.

the same set of exact hits.

The iterative algorithm to compute exact hits has the following inputs: an admissible fuzzy query, v, and an inverted file index I_F^Γ for 2^f inverted lists $L(v), v \in \{0,1\}^f$. The output of the algorithm is a list of exact hits which is denoted by $H_\Gamma(v)$.

The algorithm starts by incrementing the first inverted list denoted by $L(v_0)$ and assign it to $L^0 := L(v_0) + 1$. After that, for all $n = 0, \cdots, N-1$, L^{n+1} is computed iteratively by intersecting L^n and $L(v_{n+1})$ and then incrementing it by 1: $L^{n+1} := (L^n \cap L(v_n + 1)) + 1$, where $n = 0, \cdots, N-1$. Once L^N is computed, $N+1$ is decremented to adjust the inverted lists and this adjusted lists are the exact hits: $H_\Gamma(v) = L^N - (N+1)$.

Table 3.1: List of Feature Functions used to create Document.

Function	joint1	joint2	joint3	joint4	threshold
Right leg ahead	root	ltibia	lhipjoint	rtoes	NA
Left leg ahead	root	rtibia	rhipjoint	ltoes	NA
Right leg sideways	lhipjoint	rhipjoint	rhipjoint	rtibia	1.2*hip width
Left leg sideways	rhipjoint	lhipjoint	lhipjoint	ltibia	1.2*hip width
Right knee bent	rfemur	rhipjoint	rfemur	rtibia	110
Left knee bent	lfemur	lhipjoint	lfemur	lrtibia	110
Right foot raised	[0,0,0]	[0,1,0]	[0,Ymin,0]	rtibia	1.2*right humerus length
Left foot raised	[0,0,0]	[0,1,0]	[0,Ymin,0]	ltibia	1.2*left humerus length
Right hand reaching out to the front	root	lclavicle	rclavicle	rwrist	right humerus length
Left hand reaching out to the front	root	rclavicle	lclavicle	lwrist	left humerus length
Right hand above neck	thorax	lowerneck	lowerneck	rwrist	0.2*right humerus length
Left hand above neck	thorax	lowerneck	lowerneck	lwrist	0.2*left humerus length
Right elbow bent	rhumerus	rclavicle	rhumerus	rwrist	110
Left elbow bent	lhumerus	lclavicle	lhumerus	lwrist	110
Hands far apart sideways	lclavicle	rclavicle	lwrist	rwrist	2.5*shoulder width
Right hand lowered	[0,0,0]	[0,-1,0]	[0,Ymin,0]	rwrist	-1.2*right humerus length
Left hand lowered	[0,0,0]	[0,-1,0]	[0,Ymin,0]	lwrist	-1.2*left humerus length
Root behind frontal plane	rtibia	upperneck	ltibia	root	0.5*right humerus length
Spine horizontal	upperneck	root	[0,0,0]	[0,1,0]	110

CHAPTER 4

DYNAMIC TIME WARPING TECHNIQUES

This chapter provides a brief overview of Dynamic Time Warping (DTW) and the process of threshold selection. The threshold plays an important role in determining the matches in the algorithm. The latter part of chapter describes two techniques to index motion data using Dynamic Time Warping. Both these techniques have been refered from the book, Information retrieval for music and motion [16].

The Dynamic Time Warping algorithm can be explained as an algorithm that calculates an optimal warping path to correspond two time series. The algorithm calculates both warping path values between the two series and the distance between them using dynamic programming. Formally, consider two numerical sequences $(a_1, a_2, \cdots, a_n)$ and $(b_1, b_2, \cdots, b_m)$ which are of different length $(n \neq m)$. The algorithm starts with the calculation of local distances between the elements of the two sequences. These distances can be of different types depending on the data and application. Euclidian distance is the most commonly used distance. The local distances are stored in a distance matrix D which has n rows and m columns. Each element of the matrix $D$is given by:

$$d_{ij} =\mid a_i - b_j \mid, i = \{1, n\}, j = \{1, m\}.$$

For $i = 1, \cdots\cdots, n$ and $j = 1, \cdots\cdots, m$, where all denotes a local distance. The next step in the DTW algorithm is to determine the matching cost matrix C. This is performed using the local distances matrix D and by computing the minimal distance

between the two sequences using a dynamic programming algorithm and the following optimization criterion:

$$c_{ij} = d_{ij} + min\left(c_{i-1,j-1}, c_{i,j-1}, c_{i-1,j}\right),$$

where c_{ij} is the minimal cost between the subsequences $(a_1, a_2, \cdots, a_i)$ and $(b_1, b_2, \cdots, b_j)$. A warping path is a path through the matching cost matrix from the element c_{11} to element c_{nm} consisting of those c_{ij} elements that have contributed to the distance in c_{nm}.

The global warp cost, GC(a, b), of the two sequences is defined as shown below:

$$GC(a, b) = 1/p \sum_{i=1}^{p} w_i,$$

where w_i are the elements belonging to the warping path and p is the number of elements in the path. The warping path is typically subject to several constraints:

1. Boundary conditions: The path starts in the left bottom corner and ends in the right top corner. This means that the warping path starts and finishes inthe diagonally opposite corner cells of the cost matrix.
2. Continuity: The path advances gradually, step by step. The indices i and j increase by a maximum of 1 unit at each step. This restricts the possible steps in the warping path to adjacent cells including diagonally adjacent cells. Given $w_k = (i, j)$, then $w_{k-1} = (i', j')$, where $(i - i') \leq 1$ and $(j - j') \leq 1$
3. Monotonicity: Both indices i and j used for traversing through sequences never decrease. This forces the points in the path W to be monotonically spaced in time. Given $w_k = (i, j)$, then $w_{k-1} = (i', j')$, where $i - i'$ 0 and $j - j'$ 0.

Dynamic Time Warping (DTW) finds the best global match or alignment between the two sequences by time warping them optimally. The Figure 4.1 shows the result of DTW on two sequences of a pick box from floor action.

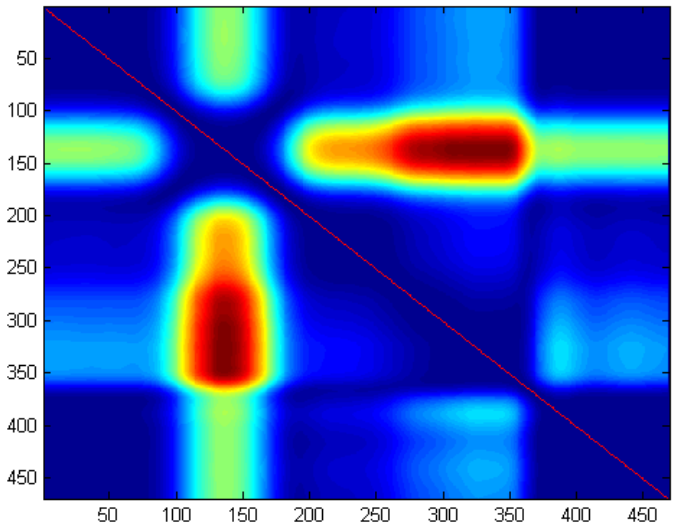


Figure 4.1: Result of DTW technique on 2 similar pick queries.

DTW is very flexible in the presence of nonlinear time deformations. It is used in many areas for pattern matching. Last *et al.* and Keogh[18] use DTW to solve distortions of the time axis and compare time series data. Cardle *et al.* sketch how to use DTW-based indexing techniques to accomplish the retrieval task of motion files. Kovar and Gleicher[19] identify numerically similar motions by means of a DTW-based index structure termed match web.

We used two different local distance measures for matching motion data using DTW. The first distance function uses joint rotations represented as quaternion and the second distance function takes into account a cartesian 3-D point cloud to calculate distance. DTW is a very powerful tool but it uses the whole sequences and can be very demanding in terms of processing and space required.

4.1 Scaling and Thresholding the DTW Score

Dynamic time warping tries to find the best matching occurrence of the query in a given sequence. If both the query sequence and the sequence to be searched are synchronized, there is a frame by frame match. However, if the given sequence is performed faster or slower than the query, the query is shrunk or enlarged to match the other sequence. This is very common in motion sequencing as humans rarely have movements at a uniform pace. Hence, a minor warp is always expected.

In cases where the actions being compared differ greatly, the cost of matching future consecutive frames in the sequences may be very high, compared to keeping one sequence frozen and warping large lengths along the other. This results in long horizontal or vertical paths in the matching cost matrix. To ensure these matches are rejected, we can associate a penalty for following non-diagonal steps.

In our implementation, we keep count of the number of non-diagonal steps made in the matching path. We later increase the cost obtained by an amount proportional to these counts. These also depend on the actual length of the match. We use regression to determine the proportionality constants and dependence on length. We obtain these values for all the motion files in the database, and store them in a matrix $A = [C_1 \mid C_2 \mid C_3 \mid C_4 \mid C_5]$, where C_1 is a column with DTW score, C_2 is a column with warp along sequence 1, C_3 is a column with warp along sequence 2, C_4 is a column with length of DTW match, and C_5 is a column with a constant 1.

For a correct DTW match, we expect the score to be around a value Y^+, and for an incorrect match, we expect it to be around a value Y^-. We create a column vector B, where ith row is Y^+ if the query matches the ith sequence, else ith row is Y^-. In our case, we chose $Y^+ = 0$ and $Y^- = 100$. We use MATLAB's regression

operator $\setminus$ to determine the relationship between A and B as $x = A \setminus B$. We then compute the corrected DTW score as:

$$CorrectedDTWscore = x_1 * c_1 + x_2 * c_2 + x_3 * c_3 + x_4 * c_4 + x_5$$

The corrected DTW score is thresholded to give a binary yes or no decision for a match. Ideally, a threshold of $\frac{(Y^+ + Y^-)}{2}$ should perform well. However, for a better estimate, we iteratively determine the best threshold between Y^+ and Y^-, which gives the maximum number of correct decisions for a particular query.

4.2 Implementation

Dynamic time warping is used to compare two motion series in the presence of nonlinear time deformations. The choice of measure for local cost or distance plays an important role. The two techniques differ in the local distance measures. One uses joint angle parameters (quaternions) and the other uses 3-D coordinates for all joints.

4.2.1 3-D Point Cloud based Distance:

A motion data stream encodes the joint configurations over time. It represents this configuration as a series of frames and each frame has an absolute position of root and the offset of each of the joints with respect to the root. Each frame can be thought of as a cloud of 3-D points where each 3-D point corresponds to one joint. A local distance measure is based on a comparison of pose-driven 3-D point clouds is used. Before the actual comparision between the two poses is performed, the following steps are performed:

- A time window around both frames is incorporated to provide the notion of smoothness.

- Then the two short pose sequences are converted to 3-D point clouds based on the joints of the underlying kinematic chain.
- Finally, an optimal alignment of the two point clouds is computed with respect to a suitable transformation group to achieve invariance under certain global translations and rotations.

After this pre-processing, the total distance between corresponding points of two poses is calculated which constitutes the 3-D point cloud distance between the two original poses. The distance is expressed mathematically as:

$$C^{3D}\left(\Gamma\left(n\right),\Gamma\left(m\right)\right)=min_{\theta,x,z}\left(\sum_{i=1}^{k}w_i\parallel p_i-T_{\theta,x,z}\left(p\prime_i\right)\parallel\right), \tag{4.1}$$

where $\Gamma(n)\in\Re^{3J}$ and $\Gamma(m)\in\Re^{3J}$ are the two poses contained within the same mocap data stream $\Gamma:[1:T]\rightarrow\Re^{3J}, and n,m\in[1,T]$. The pose $\Gamma(n)$ has ρ preceding frames and ρ subsequent frames, yielding the $2\rho+1$ frame $[n-\rho,n+\rho]$. A point cloud P containing $k=|J|(2\rho+1)$ points $p_i=(x_i,y_i,z_i)^T$ is formed with 3-D points of all joints. The comparison between the poses $\Gamma(n)$ and $\Gamma(m)$ is performed on the basis of the corresponding point clouds. $T_{\theta,x,z}$ is the transformation that simultaneously rotates all 3-D points of a point cloud about the y axis by an angle $\theta\in[0,2\pi)$ and then shifts the resulting points in the xz plane by an offset vector $(x,0,z)^T\in\Re^3$. These transformations provide invariance under certain global transformations. The weights w_i are assigned to 3-D points based on their impact on the motion. The minimization of θ,x,z is performed for the 3-D point cloud distance computation. This step is the same as the "Procrustes problem" for solving optimal solutions of the parameters [20]. The minimization of θ,x,z is done by solving the equations below:

$$\theta_{min}=arctan\left(\frac{\sum_{i=1}^{K}w_i\left(x_i\cdot z_i\prime-z_i\cdot x_i\prime\right)-\left(\bar{x}\cdot\bar{z}\prime-\bar{z}\cdot\bar{x}\prime\right)}{\sum_{i=1}^{K}w_i\left(x_i\cdot x_i\prime-z_i\cdot z_i\prime\right)-\left(\bar{x}\cdot\bar{x}\prime-\bar{z}\cdot\bar{z}\prime\right)}\right), \tag{4.2}$$

$$x_{min}=\bar{x}-\bar{x}\prime\cdot\cos(\theta_{min})-\bar{z}\prime\cdot\sin(\theta_{min}), \tag{4.3}$$

$$z_{min} = \bar{z} + \bar{x}' \cdot \sin(\theta_{min}) - \bar{z}' \cdot \cos(\theta_{min}), \tag{4.4}$$

where $\bar{x} = \sum_i w_i x_i$, $\bar{z} = \sum_i w_i z_i$, $\bar{x}' = \sum_i w_i x_i'$ and $\bar{z}' = \sum_i w_i z_i'$.

The input to the equations are the point clouds $P = (p_i)\ i \in [1 : K]$ with $p_i = (x_i, y_i, z_i)^T$ and point cloud $P' = (p_i')\ i \in [1 : K]$ with $p_i' = (x_i', y_i', z_i')^T$. The weights $w_i \in \Re, 1 \leq i \leq K$ are defined such that $\sum_{i=1}^{K} w_i = 1$

The 3-D point cloud distance was proposed by Kovar and Gleicher[19] and are invariant only under a three-dimensional subgroup of the six-dimensional group of rigid 3-D Euclidean transformations. Rotations around y axis and translations in xz plane are considered which are not sufficient for all classes of motions. The 3-D point cloud distance is not invariant to scaling too.

4.2.2 Quaternion-based Distance:

Unit quaternions are a good mathematical tool to represent joint rotations. Joint configurations in a motion stream can be encoded using quaternions. Each pose configuration, (t_r, R_r, R_b), consists of a translational parameter t_r and a rotational parameter R_r that determines the root coordinate system. The remaining rotational parameters R_b describe the joint angles within the skeletal model.

Since joint orientations are represented by quaternions, a local distance measure is defined to compare poses described by quaternions:

$$C^{Quat} : \jmath \times \jmath \rightarrow [0, 1]$$

If $j = (t_r, R_r, R_b)$ and $j' = (t'_r, R'_r, R'_b)$ are two pose configurations and q_b and q'_b denote the unit quaternions describing the relative joint rotations R_b and R'_b, respectively, then the local distance between unit quaternions is calculated by using the inner product between single quaternions: $C^{Quat}(j, j\prime) = \sum_{b \in B} w_b \cdot 2/\pi \cdot arccos \mid q_b \bullet q_b' \mid$, where $\mid q_b \bullet q_b' \mid$ is the absolute value of the inner product of q_b and q'_b and gives the

cosine of the angle $\phi \in [0, \pi]$ enclosed between these unit vectors, w_b is the weight assigned to joint b and B is the set of joints b. These weights are taken into consideration because some of the joint rotations have a much greater overall effect on the character's pose than others. The value $\phi = arccos \mid q_b \bullet q\prime_b \mid$ is the geodesic or spherical distance on S^3 which is also the length of the shortest connecting path between the points q_b and q'_b on the four-dimensional unit sphere S^3. The factor $2/\pi$ is used for normalization.

Advantages of using quaternion-based local distance are:

- The local distance becomes invariant under global transformations such as translations, rotations, and scalings of the skeleton by considering only the joint rotations and leaving the root parameters out.
- The geodesic or spherical distance between unit quaternions gives a natural cost measure for rotations.
- The use of inner product to calculate the distance between unit quaternions is an efficient way of distance computation.

Though there are advantanges of using a quaternion-based distance, there are disadvantages too. The use of weights for joints is not very effective as the importance of certain rotational joint parameters may change with the course of the motion. The quaternion-based distance considers only differences in body postures, but lack higher derivatives like joint velocities and accelerations. These derivatives are required to perform smooth transitions in blending applications. Although this distance measure provides invariance under any global 3-D Euclidean transformation, it also leaves the progression of the 3-D coordinates in space more or less unconsidered. This can cause an over abstraction concerning motion identification. With all the advantages stated above, the quaternion-based distance is used as local distance measure in a second indexing technique based on dynamic time warping.

CHAPTER 5

EXPERIMENTS AND RESULTS

We designed two experiments to evaluate the time performance and the indexing accuracy of five different algorithms for motion indexing. The first two techniques are PCA-based and rely on the preserved information ratio obtained by dimensionality reduction of data. The first technique uses only SVD and the second uses PPCA. The third technique is based on the geometric feature functions which use joint positions of the body as parameters. The fourth and fifth techniques are based on Dynamic Time Warping and use a local distance as a metric for indexing. The first DTW-based method, DTW-Quat, uses quaternion-based distance measure and the second DTW-based method, DTW-3D, uses 3-D point clouds to calculate the distance measure. These experiments are performed using the motion capture files from CMU Graphics Lab Motion Capture database.

5.1 Performance Testing

The time performance evaluation is performed by running the five methods on a set of 10 files where each file is compared with itself and the other files. A total of 100 matches are performed by each method. These 10 motion files are of sizes ranging from 199 frames to 5,423 frames. The set of motion files used for performance testing can be found in Appendix C.

All the experiments are run on Allienware systems with Intel Core 2 Quad CPU Q9650 3.00GHZ processor, Seagate Barracuda 7200rpm hard disk, and 8 GB DDR RAM.

TimeB.png

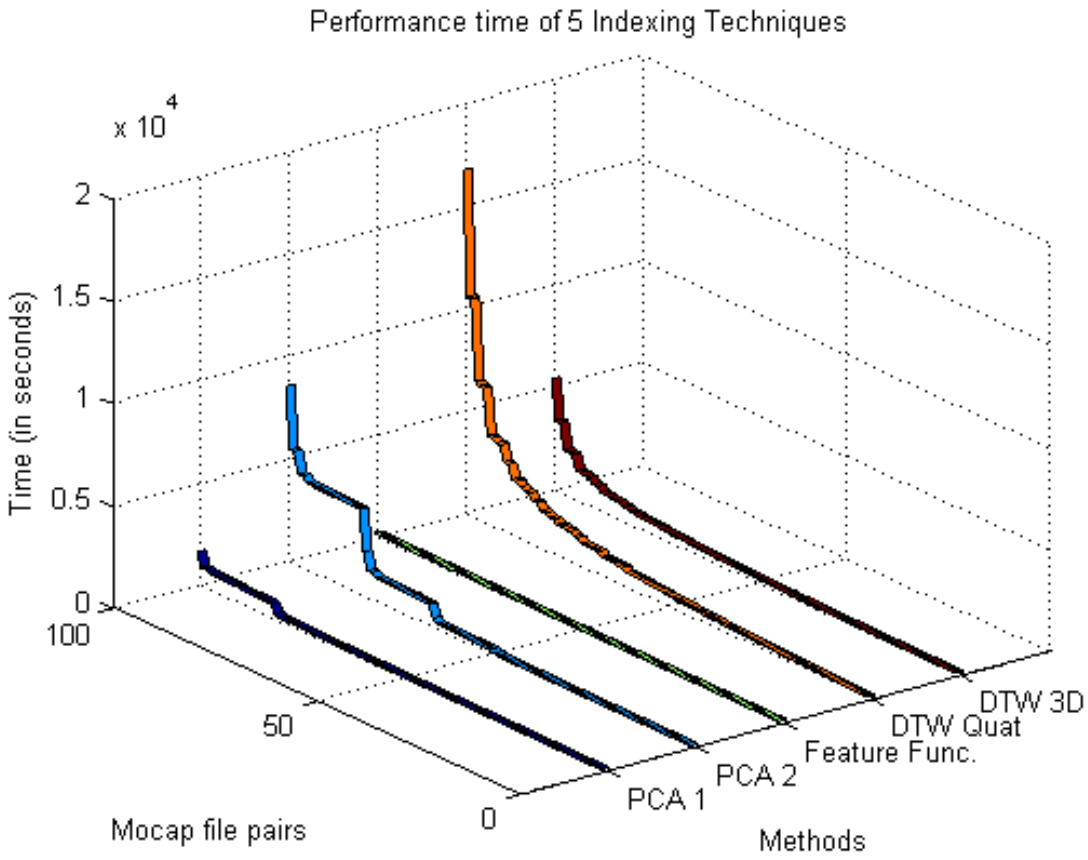


Figure 5.1: Performance of five techniques for 100 mocap file computations.

The graph in Figure 5.1 shows the time taken by each method to run on all pairs of mocap files. There are 100 pairs formed out of 10 files and we have compared the computation time for the five methods. The time performance of the feature-based techniques is the best. The reason being the use of inverted file indices which are very efficient for search operations. Whereas, DTW quaternion-based technique has worst time performance. Dynamic time warping is a time consuming technique because it calculates the distance from each frame in the motion to all frames in the other motion.

5.2 Indexing Accuracy Testing

To evaluate the accuracy of the indexing methods, each method is tested on a data set of 150 mocap files with 5 query files. Each query is run on the dataset of 150 files to find the matching files among the 150 mocap files in the database.

The five motion queries in the dataset are walk, jump, punch, sit and pick up the box actions. The reason for choosing these actions as queries is that these actions are very common and the CMU database has a large number of files containing these actions.

The dataset of 150 motion files has a good mix of various actions such that a large range of human actions is covered. They also represent important types of movement such as locomotion (walk and jump), postural (sit), and non-locomotional (punch and pick up the box). It contains 113 files which contain actions similar to at least one of the five queries. Among these 113 actions, there are 40 walk actions, 35 jump actions, 15 sit actions, 13 punch actions, and 10 pick up actions. The remaining 37 files are completely different from the five queries. A manually annotated data for these 150 test files was prepared. These annotations are helpful in comparing the result of the query on the test data set. The details of the data set can be found in Appendix C.

For each query, the techniques are quantitatively evaluated according to the ratio of false positives and false negatives among the 150 test files. The false positives ratio is the number of positive matches which do not correspond to a positive match in the manually annotated data divided by the total number of positive matches in the manually annotated data. The false negative ratio is the number of negative matches which do not correspond to negative matches in the manually annotated data divided by the total number of negative matches in the manually annotated data. These two statistical measures are good error criteria and are universally accepted.

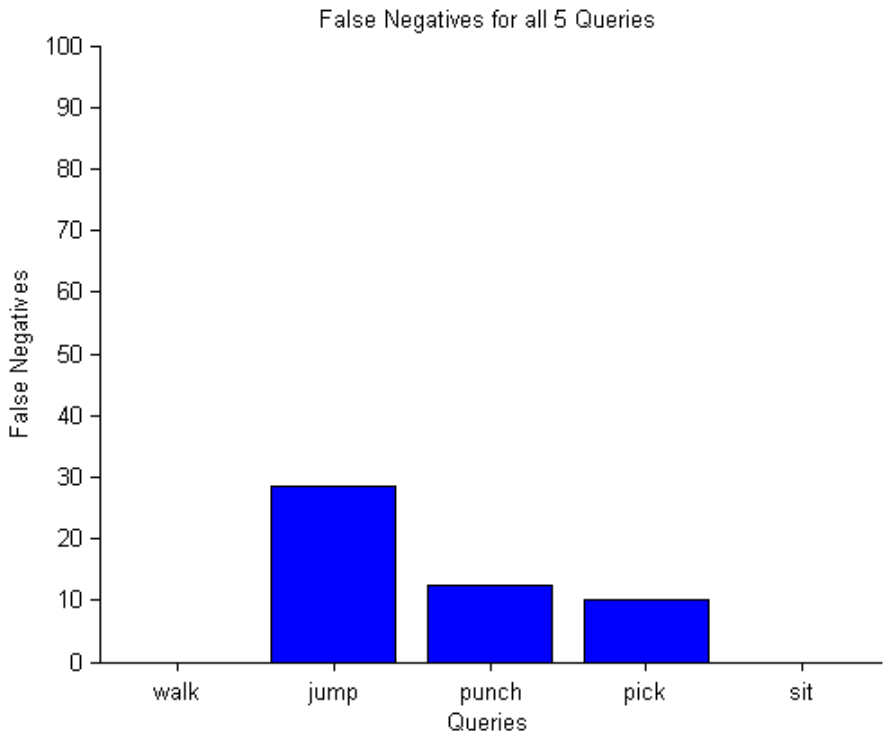


Figure 5.2: False negatives for PCA-SVD technique.

The PCA based technique PCA-SVD first segments the data according to different behaviors and then tries to find matching segments for the given motion files. The results reveal that the technique identifies many false positives (see Fig 5.3). The main reason for this is the improper data segments (inaccurate start and end points) created by the technique which deteriorate indexing. However, the number of false positives is low (see Fig 5.2).

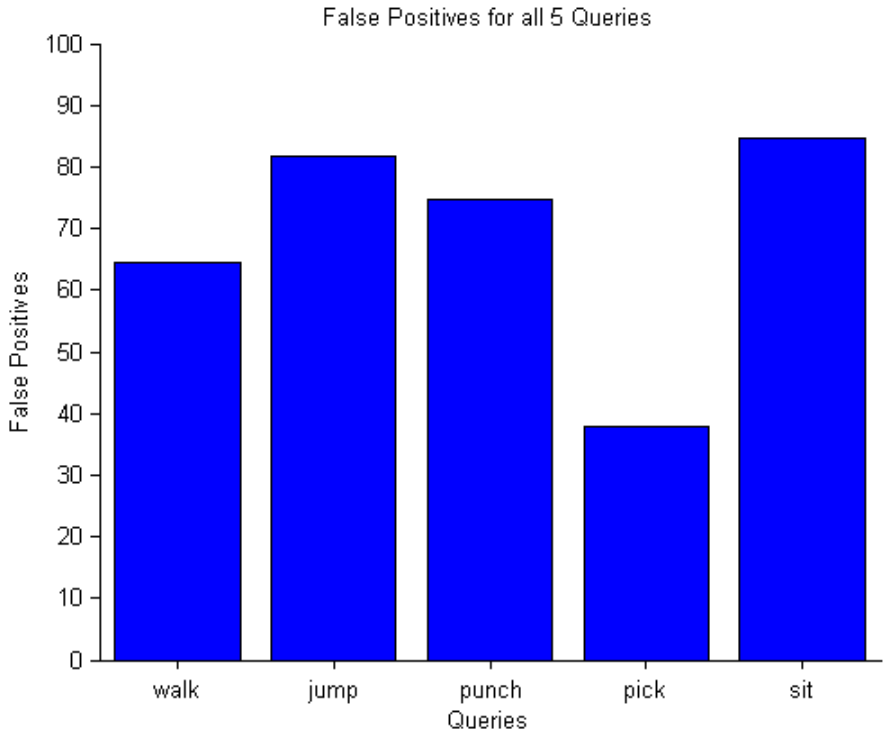


Figure 5.3: False positives for PCA-SVD technique.

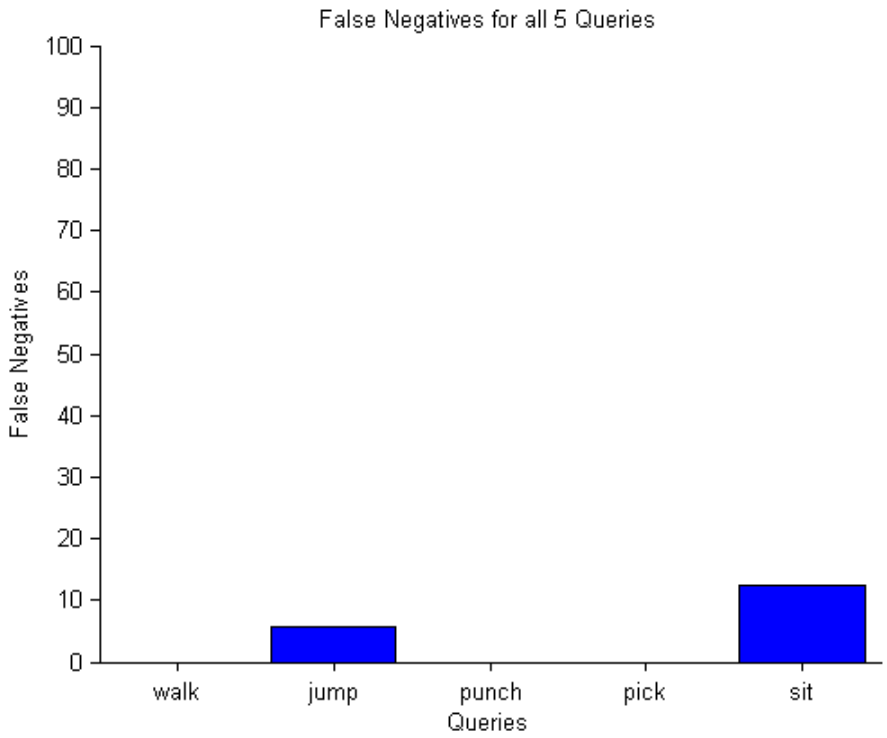


Figure 5.4: False negatives for PPCA technique.

The experimental results for the second PCA-based technique, PPCA, are also not satisfactory. This technique also segments the data first and then performs indexing on these segments. The segment end points detected by the technique are accurate but it also creates a number of false segments. These small and false segments are responsible for a large number of false positives (see Fig 5.5). On the other hand, the number of false negatives is reasonably low (see Fig 5.4).

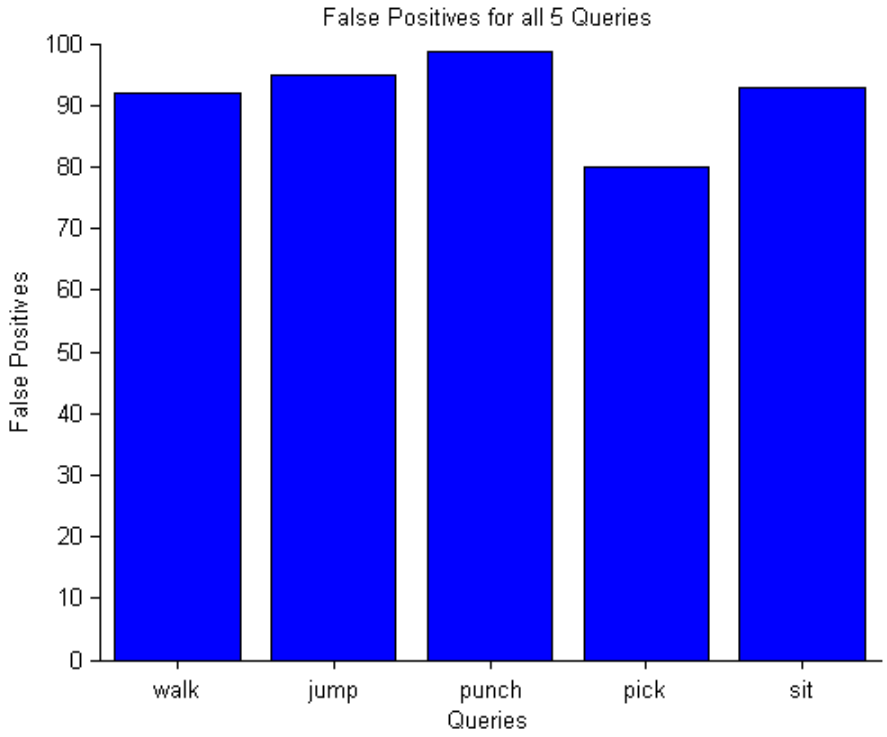


Figure 5.5: False positives for PPCA technique.

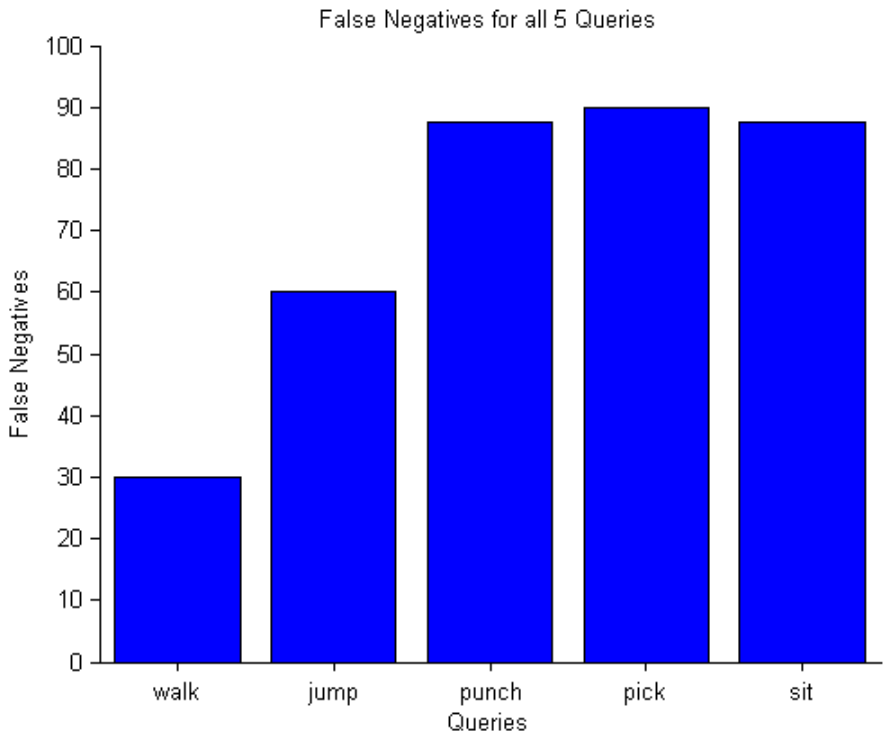


Figure 5.6: False negatives for Feature-based technique.

The feature-based technique finds the exact query sequence in each dataset match. The results are accurate but are constrained by the fact that the binary sequence of the query and the data file should be exactly the same. That reduces the number of matches and results in large number of false negatives (see Fig 5.6). The remarkable thing about this technique is that it finds all the occurrences of the query in the data file and not just the best possible match. The number of false positives is very low (see Fig 5.7).

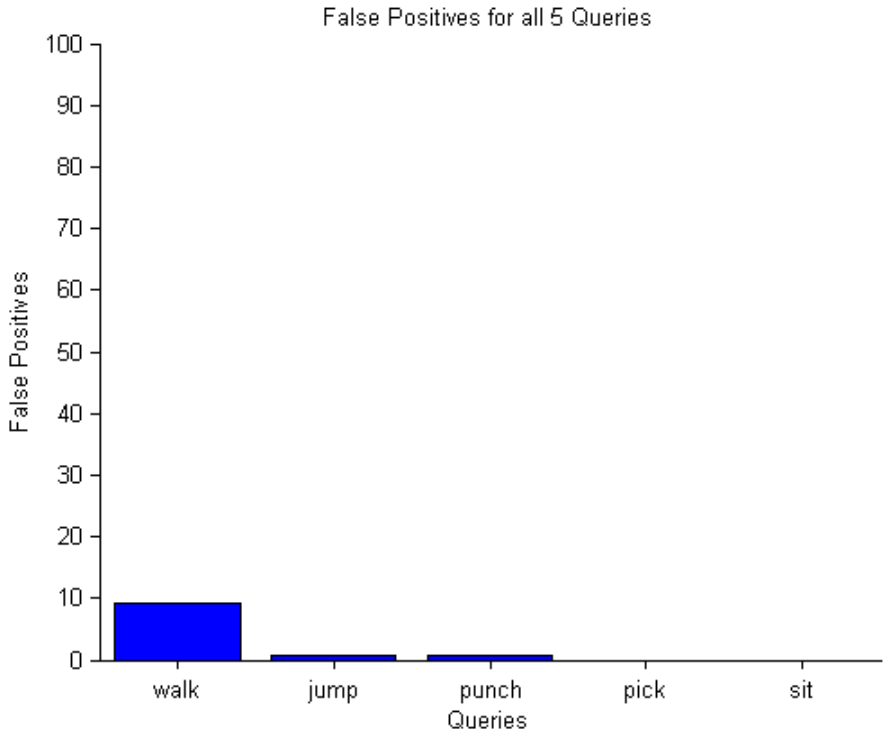


Figure 5.7: False positives for Feature-based technique.

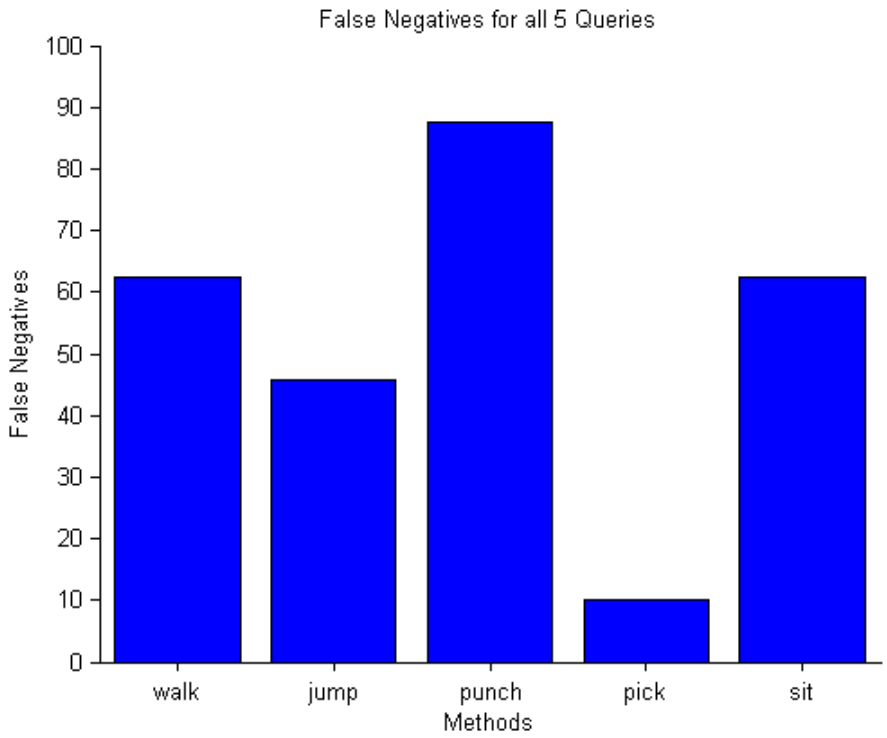


Figure 5.8: False negatives for DTW-Quat technique.

Dynamic time warping tries to find the matching between the two files based on the minimum distance warping path. DTW matching is affected if there are pattern differences in the files. For example, if the query is an action describing a jump and the dataset file is a hop, then DTW-Quat will return a rejection. Similarly, a fast walk or a walk with swinging arms gives a low score when matched with a normal walk. There are a large number of false negatives detected (see Fig 5.8). We make use of all 30 joints to calculate the local distance measure and the distance increases even if a small number of joints mismatch. The noticeable feature of this technique is that the number of false positives is zero because to have a match there should be a very small difference in the joint configurations (see Fig 5.9).

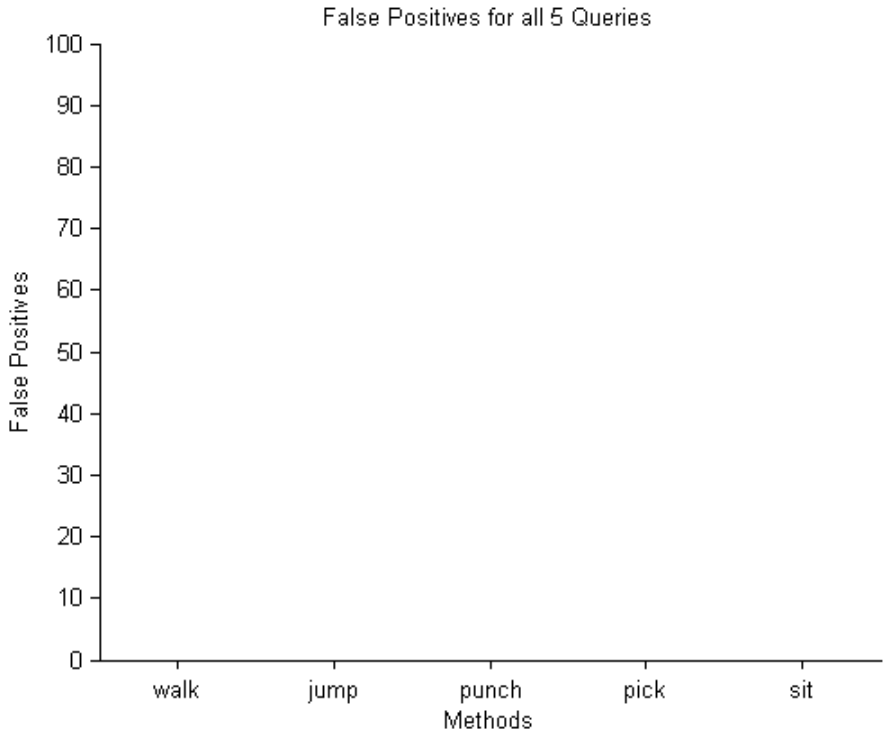


Figure 5.9: False positives for DTW-Quat technique.

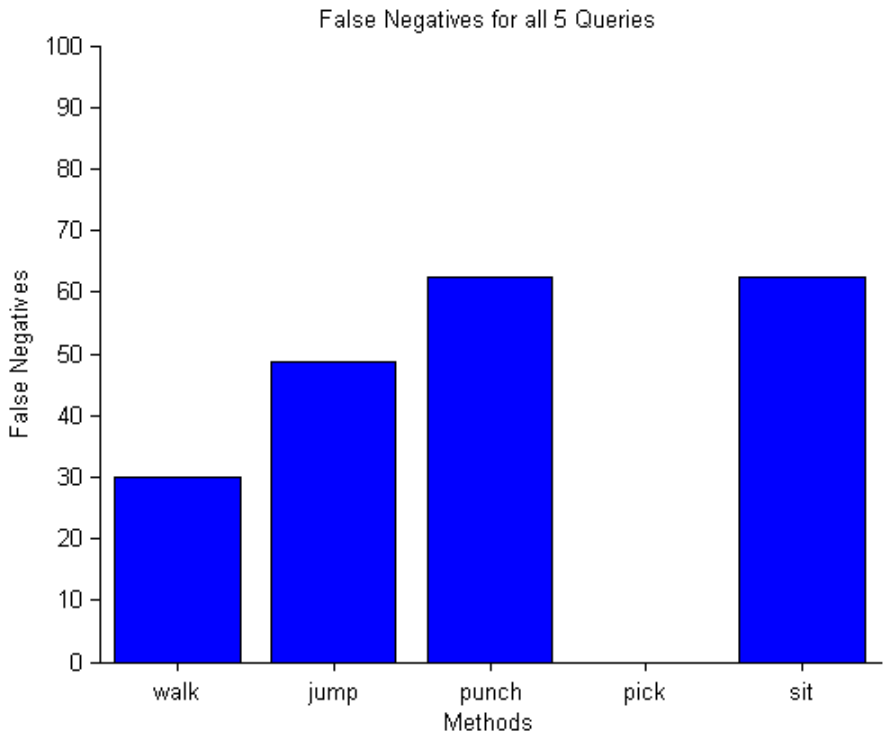


Figure 5.10: False negatives for DTW-3D technique.

Similar to the quaternion-based method, Dynamic Time Warping using 3D coordinate points based technique performs better than PCA-based and feature-based techniques. The false negatives are less in all the five queries compared to the quaternion-based method because the latter is very sensitive is very sensitive to small differences in joint configurations (see Fig 5.10). The pick up the box query gives 0% false negatives beacuse all the pick data files have the same orientation and 3-D location. Also the number of false positives are less which is an indication of a good technique (see Fig 5.11).

5.2.1 Overall Results

The false positives detected by each method are shown in Figure 5.12 and the graph in Figure 5.13 shows the overall false negatives of each of the methods.

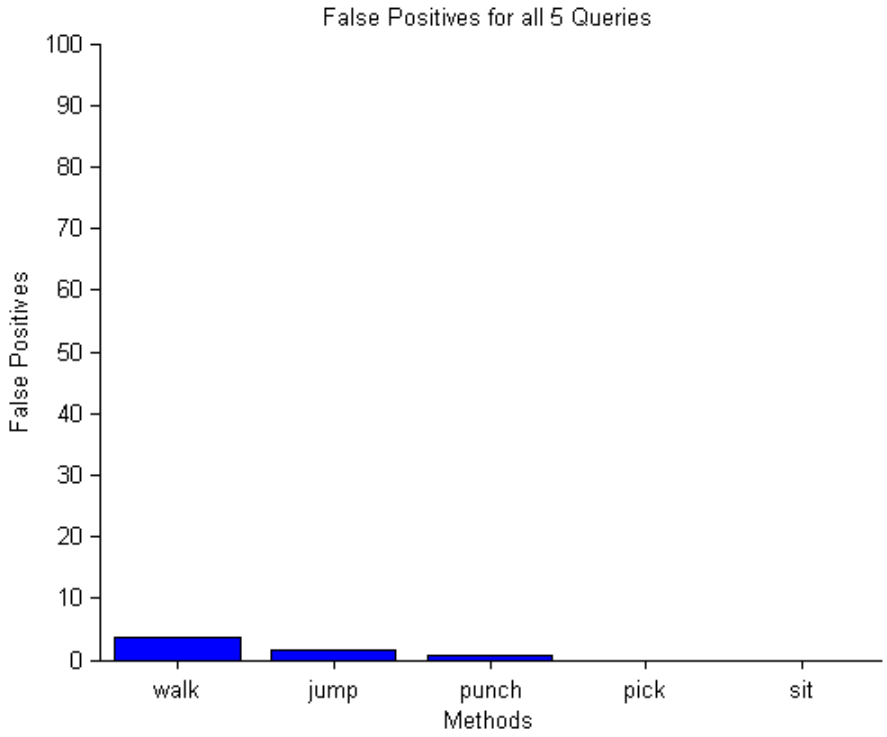


Figure 5.11: False positives for DTW-3D technique.

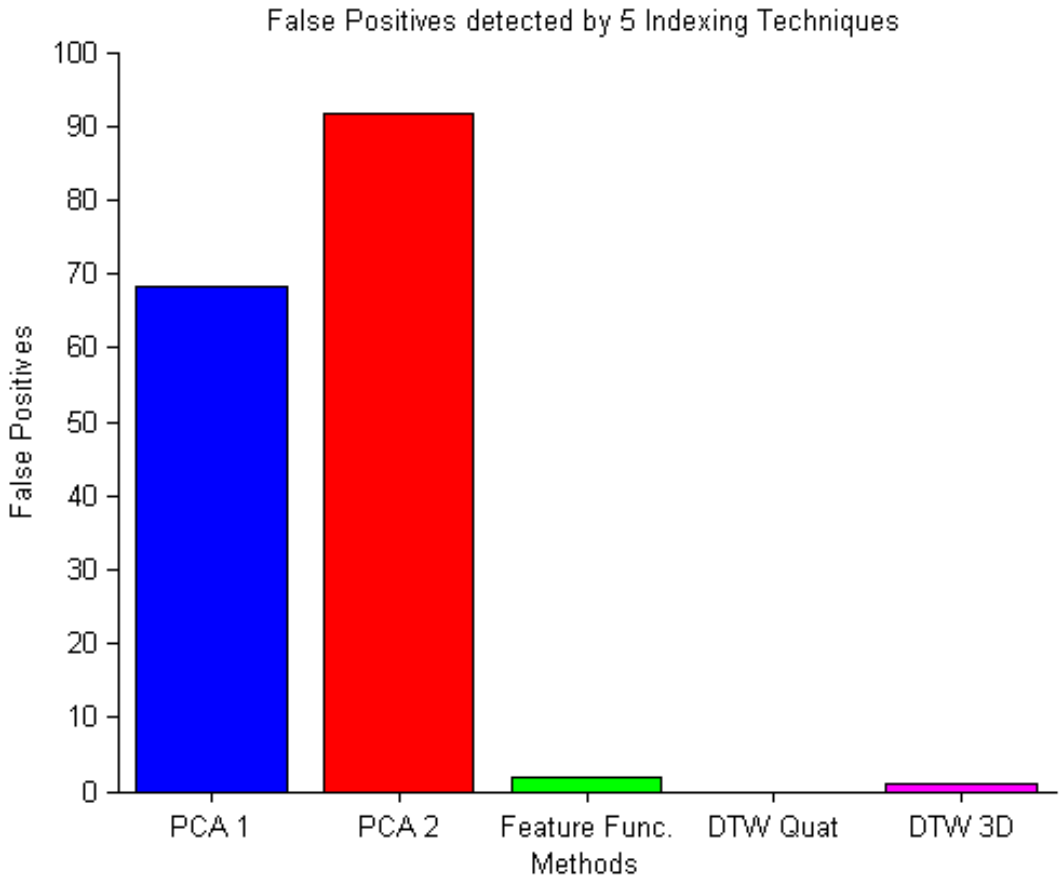


Figure 5.12: False Positives detected by each Technique.

The two PCA-based have the lowest number of false negatives but at the same time have large number of false positives (close to 90%) which shows that these two techniques incorrectly classify non-matches as matches. The feature-based and DTW quaternion-based techniques perform better than the PCA-based techniques. They have almost similar results with DTW-Quat performing a little better because both these techniques are very sensitive to small differences in joint configurations. DTW-3D has the best performance among all the five techniques because although it has a small number of false positives, the false negatives are also very less.

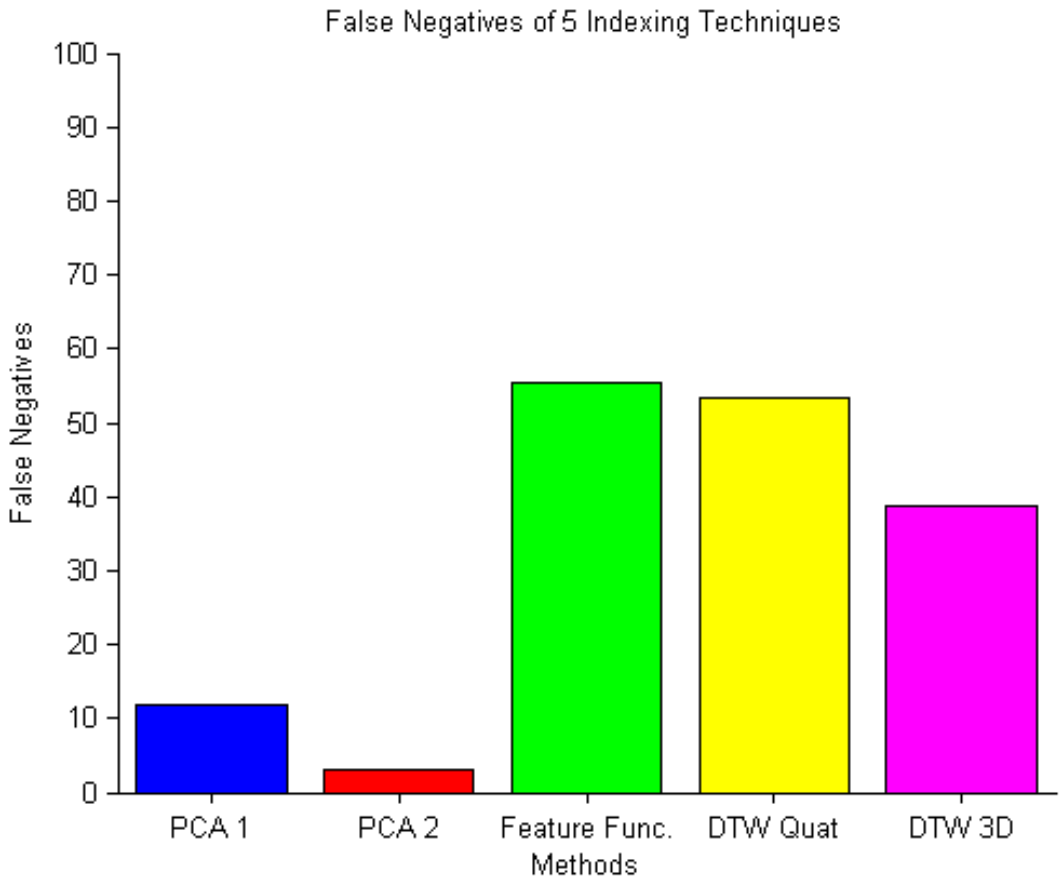


Figure 5.13: False Negatives detected by each Technique.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, several methods of motion indexing were implemented and compared. Dynamic Time Warping 3D-based technique was found to perform the best among all five techniques when compared by false positives and false negatives metrics. The time performance of feature-based technique was found to be the best but at the same time the indexing performance was poor. Two PCA-based segmentation techniques were implemented and a new method was developed to perform indexing on these segments. Due to poor segments created by segmentation techniques, the indexing results of the new method were not satisfactory.

The motion indexing problem can be generalized to consider sets of essential actuators instead of the whole body. This generalized problem will address the indexing and retrieval of a particular action even when spliced with other simultaneous actions.

All the five techniques implemented consider all joints of the body to find a match. This leads to a mismatch in cases of hybrid motions where several motions are performed simultaneously. For example, a walk action combined with a hand wave cannot properly matched with either a walk query or a hand wave query. Methods can be developed which select a subset of joints relevant to the query ignoring the other joints while matching.

Furthermore, uniform scaling in addition to Dynamic Time Warping can be used to improve results when matching sequences, where we stretch and compress the sequences by linearly re-sampling them to make the same lengths and then perform

DTW[18]. DTW matching can also be improved by constraints such as Sakoe-Chiba band[21].

APPENDIX A

ANIMATION

Animation techniques can be broadly classified as Traditional Animation, Stop Animation and Computer Animation.Traditional Animation, also known as classical animation or hand-drawn animation, is a technique in which each frame is hand drawn. It is historically popular and was used for most animated films of the 20th century. The individual frames of these films are photographs of drawings, which are first drawn on paper. Few examples of traditional animation are full animation, limited animation, rotoscoping and live action animation.

Stop Animation is a technique which utilizes physically manipulated objects such as puppets and props to create action. This object changes in small increments between the frames, creating the illusion of a motion when the series of frames is played as a continuous sequence. Sometimes, clay figures are also used to create this type of animation and such technique is called clay animation. Various types of clay animation include Puppet animation, cutout animation, object animation and model animation.

Before computer systems came into existence the whole process of animation was tedious and time consuming. With the use of computers the frame redrawing labor had been minimized since, copying and pasting duplicate elements between successive frames is very efficient. Computer Animation, also known as CGI animation, is a technique which uses a computer to generate motion. The illusion of motion is created by displaying an image on computer screen repeatedly. Each time the previous image is replaced by a new image that is similar to the previous, but advanced slightly in the time domain. The two main types of computer animation are 2D animation and 3D animation.

2-D computer animation refers to creating a notion of movement in a two dimensional space using 2-D bitmap graphics or vector graphics. This method considers only 2 dimensions namely X and Y. 2-D computer animation is a digital version of the earlier

frame-by-frame techniques. Some tools used to produce 2D computer animations are Flash and Illustrator. Many people also generate simple 2-D computer animations with applications like PowerPoint. 2-D animation somehow lacked the essence of a real motion, since all the real-world sceneries and objects are in 3-D space. Many cartoons were created by simulating the 3-D effect by the use of gradients, and varying highlights.

3-D animations are created by adding one more dimension to the 2-D environment. This extra dimension can be character, camera, or particle. The two dimensional object has length and breadth but it lacks thickness. That is the reason a 2D object cannot be rendered from viewpoints. The extra dimension in 3-D animated objects gives depth to them.3D Animation is the most prevalent of all techniques of computer animation nowadays.

3-D animation is mainly done as key frame animation and motion based animation. One of the most well-known and oldest styles of animation is keyframe animation, or keyframing. Keyframes are key points in a sequence of motion that define pivotal points in that motion. The beginning and ending positions are always keyframes, other keyframes are defines as pivotal points inside a sequence. Keyframing concerns changing the shape, position, spacing, or timing of an object in a sequence of frames. The major changes to the object happen in the key frames. In traditional animations, the keyframes were drawn by the artists and a rough animation is created for a scene. Then the necessary cleanup and in-between is performed to refine the animation. In computer animation this workflow is basically the same. The animator creates the important frames of a sequence and the software fills in the gap between key frames. This process is also known as tweening. The animator can correct the result at any point or change an in-between by including an additional keyframe to further refine the movement. Contrary to tweening, in this case every frame of the animation is

directly modified or manipulated by the animator. This method resembles much of the drawing of traditional animation and gives complete control over the animation to the animators. Surprising fact about keyframing is that the techniques have not changed much since the early 1900's. Most of the basic principles are still applied today. However modern 3-D software tools and packages had made keyframing easier to use.

The recording and mapping of motion from a real object into a synthetic object is known as motion-based animation. It involves sensing, digitizing, and recording of the motion of a real object. It is very difficult to extract the objects motion directly from a raw video. Therefore the object whose motion is captured is typically instrumented to detect and record the positions of key feature points. The instrumentation is usually done by placing markers on the body of a subject or by wearing a body suit which has markers or sensors attached to it. The position of these markers are recorded by a special-purpose camera and then digitized. Motion capture was first used sparingly due to the limitations of the technology. Nowadays the technique has matured and it has been increasingly used in everything from video game animation to CG effects in movies. Motion Capture is not as precise as Keyframing but it provides an immediate result which is not found in traditional animation techniques. Moreover keyframing is a slow animation method.[22]

APPENDIX B

MOTION CAPTURE TECHNOLOGY

B.1 Breif History

Human beings are experts at recognizing movement from birth. Human beings perceive visual information in this order: movement, form and color. Since movement is the strongest, it is instinctively the most important for our survival. In the late 1800's several people (most notably Marey and Muybridge, with the help of photography) analyzed human movement for medical and military purposes. Disney Studios traced animation over film footage of live actors playing the scenes to get a realistic motion for the human characters in Snow White. This method is also known as rotoscoping and has been successfully used for human characters ever since. In the late 1970's, when it began to be feasible to animate characters by computer, animators adapted traditional techniques, including rotoscoping. After late 1970's humans were used to produce computer animation.

- 1980-1983: Simon Fraser University Goniometers

  During this time techniques and devices used in Biomechanics were applied to analyze human motion by the computer graphics community. In the early 1980's, Tom Calvert, a professor of kinesiology and computer science at Simon Fraser University used the motion capture apparatus together with Labanotation and kinematic specifications to fully specify character motion.

- 1989: Kleiser-Walczak Dozo

  In 1989, Kleiser-Walczak produced Dozo, a (non-real-time) computer animation of a woman dancing in front of a microphone while singing a song for a music video. They used an optically-based solution from Motion Analysis which uses multiple cameras whose output is the 3-D trajectory of each marker in the space. However, with these types of system, the hindrance was tracking points as they are occluded from the camera.

- 1991: Videosystem Mat, the Ghost

  Around this time, Waldo C. Graphic, Videosystem, a French video and computer graphics producer, worked on the problem of computer puppets, which resulted in real-time character animation system whose first character was Mat the ghost. Using DataGloves, joysticks, Polhemus trackers, and MIDI drum pedals, puppeteers interactively performed Mat, chroma-keyed with the previously-shot video of the live actors. Videosystem, now known as Medialab, continues to develop a reliable production tool, having produced several hours of production animation in total, for more than a dozen characters.

- 1992: SimGraphics Mario

  Around 1992 SimGraphics developed face waldo, a facial tracking system. The important feature of this system was that one actor could manipulate all the facial expressions of a character by just miming the facial expression himself. Real-time performance of Mario in Nintendo's popular videogame was the first big successes with the face Waldo system and its concomitant VActor animation system. Since then SimGraphics has been continually updating the technology of face Waldo.

- 1993: Acclaim

  At SIGGRAPH '93 Acclaim introduced a realistic and complex two-character animation done entirely with motion capture. Acclaim mainly uses the system to generate character motion sequences for video games.

- Today: Commercial systems

  Over the years, Ascension, Polhemus, SuperFluo, and others have released commercial motion tracking systems for computer animation. SoftImage, has integrated these systems into their product creating "off-the-shelf" performance animation systems.

B.2 Motion Capture Technologies

1. Mechanical: In mechanical motion capture the performer wears a very basic skeleton set made of straight metal pieces which is hooked to the back body and directly track body joint angles. Hence, it is also known as exo-skeleton motion capture systems. As the performer moves, these articulated mechanical parts also move with sensors feeling the joints rotation and measuring the performer's relative motion. Variations of mechanical mocap systems include gloves, mechanical arms, or articulated models.

   Mechanical mocap systems provide occlusion-free, low cost, and unlimited volumes of capture. These systems are real-time and wireless in nature, but they often need calibration. They also need extra sensors to determine the performers direction. Motion captured using mechanical systems are not interfered by light or magnetic fields. However, the absolute positions are not known and are estimated from the rotations. Therefore, these type of mocap system is unaware of ground. Hence, a jumping action cannot be cannot be captured due to possible feet sliding. The range of the suits used in mechanical mocap lies in the range of $25,000 to $75,000 which excludes the cost for extra positioning system.

2. Acoustic: This type of motion capture systems uses a triad of audio receivers and an array of transmitters. These transmitters are attached to performers body and are triggered sequentially to produce a click sound. The receivers measures the time the sound took to reach it from each transmitter. The calculated distance of the three receivers is triangulated to provide a point in 3D space. This positional data serves as input for an inverse kinematic system to drive the animated skeleton. This method doesnot suffer from any occlusion is-

sue but the cables used in the system cause somehindrance during performances. Furthermore, current systems are not accurate in capture.

3. Electromagnetic: Also known as magnetic motion capture, this system is very popular and involves the use of a static magnetic transmitter and an array of magnetic receivers. The transmitter is centrally located and receivers are strapped on to various parts of the subject's body. The system measures the spatial relationship with the transmitter. Calculation of position and orientation is done by the relative magnetic flux of three orthogonal coils on both the transmitter and each receiver. This system was first used in the military to track the movement of pilots. The captures are absolute and rotations are absolutely. Orientation in space can be determined which is very useful. The magnetic markers are not occluded by nonmetallic objects but can be affected by magnetic and electrical interference from metal objects in the environment like cement floors which usually contain metal.
4. Optical: Optical motion capture makes use of directionally-reflective or infrared emitting balls referred to as markers the system triangulates the 3D position of a subject using two or more cameras. Markers are attached to the body of the subject and cameras, at least three, are calibrated to provide overlapping projections. The computer is presented with each camera view to calculate the 3D position of each marker. The resulting data stream consists of a 3D position data for each marker. Major commercial vendors for optical motion capture systems are Vicon Systems and Motion Analysis.

   The optical systems can be further classified based on the type of marker used:

   (a) Passive markers: Markers are coated with a retro reflective material to reflect light back. The light is generated near each camera lens.

(b) Active markers: Markers have one or multiple LEDs which are illuminated timely to identify them by their relative positions.

(c) Time modulated active markers: Markers are same as in active marker system but are strobed one at a time.

(d) Semi Passive imperceptible markers: Makers use photosensitive marker tags to decode the optical signals. These tags can compute not only locations of each point, but also their orientation, incident illumination, and reflectance.

Optical systems have become quite popular over the last couple of years. The data obtained is very detailed. These systems provide the most freedom of movement to the subject as they do not require any cabling. The problem with optical systems is that a subject can easily occlude, or hide, one or more markers which can create "holes" in the data stream. This is sometimes reduced by adding more cameras but it also makes the tracking of the markers complicated and increases execution time. Increasing markers is not a good solution as it introduces confusion in the capture.

Markerless motion capture is an advancement in optical motion capture systems. They do not require subjects to wear equipment. Special computer algorithms are designed to allow the system to analyze multiple streams of optical input and identify human shapes, breaking them down into constituent parts for tracking.

B.2.1 Why Motion Capture is required?

Motion Capture can be refered as the 3D representation of a live performance of an actor. No matter how different an actor or object look in the simulated 3D world after motion capture, it will resemble complete physical and emotional performance. This property of motion capture makes it useful for many applications in many dif-

ferent areas. Following are the reasons whch are responsible for popularity of motion capture:

1. Impractical or dangerous situations in real life can be simulated very easily with motion capture.
2. Scenes which involve lots of resources and are expensive to perform can be done easily with mocap.
3. Realistic human movements and interactions, weight and exchange of forces which might be complex to generate can be created in a accurate manner using mocap.
4. It reduces the time of animation compared to keyframing and hence its gaining more popularity in entertainment industry.
5. Mocap can be cost and time effective as large amounts of animation data can be produced within a given time when compared to traditional animation techniques.
6. Mocap is very useful in medical science as it provides a realistic movement or reactions of a patient. Virtual surgery, diagnosis, and other innovations in the medical field can rely on mocap.

B.2.2 How is it done?

Motion capture using optical systems is performed in the following steps:

- Studio set-up: A capture room is set usually using 8 cameras and can go up to 24 cameras for multiple captures.
- Camera Calibration: Cameras are calibrated to provide overlapping projections. An initial estimation about the relative position and orientation of the cameras is made and then iteratively refined until convergence is achieved.

- Capture of movement: Subject wears a body fitting motion suit and markers are placed on the suit near selected joints of the subject. Each joint has a different degree of freedom and the choice of which joint should have markers depends on the requirements of the capture. The markers can be concentrated on a particular part of the body like the face to capture facial expressions, or can be placed on different joints all over the body. On an average, 30 markers are placed on a subject's body.

  The subject performs the actions and the mocap computer software records the positions, angles, velocities, acceleration and impulses. Data is sampled at up to 144Hz. The high sampling rate is advisable when fast motions, such as sports motions, are captured . Slower sampling rates for fast motions can often produce problems in the inverse kinematics phase since there is less frame-to-frame coherence. Mocap system produces data with 3 degrees of freedom for each marker, and direction of bones must be inferred from the relative orientation of three or more markers.
- Clean-up of data: Motion capture data is noisy and often contains gross errors. The amount of data cleaning required, usually depends on the complexity of the motions, the number of motion capture cameras, and the number of simultaneous performers. One of the simplest methods to clean the data is by first finding the size of the skeleton by determining arithmetic mean of the distances between the translated joint locations over a motion or repertoire of motions.
- Post-processing of data: The captured data can be used directly or can be post-processed depending on the application. Post-processing involves labeling each track with a marker code, filling track gaps caused by occlusions, correcting possible gross errors, filtering or smoothing noise, and interpolating data along

time. The data can be changed completely for characters or modified during post-production. It can appear as two-dimensional or three-dimensional objects.

B.2.3 Applications:

Mocap is gaining popularity in many fields. Video games use the mocap technology to simulate the characters like athletes, and dancers to gain more realism. For multi-player games mocap is very useful to simulate the inter-actions between the players since mocap has the ability to preserve the realism in actions and emotions. Motion capture is used in movies for CG effects and to create computer generated characters. Motion-based animation is essential for creating characters that move realistically. Some famous movie characters such as Gollum, the Mummy and King Kong were created using mocap. The 2006 Academy Awards had two out of three nominees for best animated film as "Monster House" and "Happy Feet" which used mocap technology. The recently released movie "Avatar" used Motion capture to animate characters and environment is a big release.

Motion capture can be used for many clinical applications in medicine. Researchers use mocap to analyze the behavior of patients which can be helpful in determining the traits related to the disease. An examples is the behavioral analysis patients suffering from Parkinson's disease. Another example is gait Analysis which is the process of analyzing the movement patterns of different people while walking. An in-depth gait analysis requires the knowledge of elementary spatio-temporal parameters such as walking speed, hip and knee angles, stride length and width, time of support, among others. To obtain this information, a 3D human motion capture system has to be used.

Motion capture is a very good tool to train animators. While access to motion capture is not a substitute for developing good art skills and good traditional character

animation abilities, it can be helpful in understanding and creating more realism. Motion capture is also useful for perceptual research. Test subjects are presented with abstract movements distilled from motion capture data and repeatable experiments can be developed that provide insights into human perception.

Virtual Reality (VR) applications requires the use of motion capture. It provides a much more involved and much better realistic experience than using a joystick or a positional handle. Motion capture also has great possibilities for location-based entertainment.

Biomechanical analysis is done which extensively uses motion capture for its ability to produce repeatable results. Motion capture can be used to measure the extent of a patient's disability as well as a patient's progress with rehabilitation. Prosthetic devices can be effectively designed and evaluatedusing motion capture.

APPENDIX C

LIST OF MOCAP FILES USED FOR TESTING

For Performance testing:

74_09

127_23

138_19

141_14

75_19

79_71

81_08

77_31

60_08

14_02

For Indexing accuracy testing:

Query Files: The query files are pruned from the original data files of CMU database.

Walk Query: 35_01

Jump Query : 118_10

Sit Query : 75_19

Punch Query : 86_01

Pick Query : 115_01

Test Data: The below is the list of test data files. The data files 86_01, 86_02, 86_04, 86_05, 86_06, 86_08, 02_05, 144_13, 144_14, 144_20, 144_21, 144_23, 113_13 and 111_19 are pruned beacuse they are very long.

35_01 walk

35_02 walk

35_03 walk

35_04 walk

35_05 walk

35_06 walk

35_07 walk

35_08 walk

35_09 walk

35_10 walk

141_25 fast walk

16_15 walk

16_16 walk

16_21 fast walk

16_22 fast walk

16_31 walk

16_32 walk

16_47 walk

16_58 walk

32_01 walk

32_02 walk

39_01 walk

39_02 walk

39_03 walk

39_04 walk

02_01 walk with hands moving more

02_02 blinks

08_01 walk with hands moving more

08_02 walk with hands moving more

08_03 walk with hands moving more

10_04 walk with hands moving more

08_06 fast walk

08_07 long stride walk

08_08 fast walk

08_09 fast walk

12_01 slow

12_02 slow

12_03 walk

35_28 walk

35_29 walk

115_01 Pick

115_02 Pick

115_03 Pick

115_04 Pick

115_05 Pick

115_06 Pick

115_07 Pick

115_08 Pick

115_09 Pick

115_10 Pick

19_12 turns while sitting

22_02 sit down

22_03 sit down

23_01 sitting and writing

23_09 sit down

86_15 body jerk, one leg higher

143_18 sit down

75_17 superfast, medium sit

75_18 superfast, high/ok sit

75_19 sit down

75_20 sit on the ground

113_15 sit down

15_10 walk and sit sequences

86_01 punch

86_02 punch

86_04 punch

86_05 punch

86_06 punch

86_08 punch

111_19 go forward and punch

113_13 go forward and punch

141_14 very fast punch

143_23 fast punch

144_20 right punch in different directions

144_21 right punch in different directions

144_13 left punch in different directions

144_14 left punch

02_05 right punch in different directions

118_01 jump

118_02 jump

118_03 jump

118_04 jump

118_05 jump

118_06 jump

118_07 jump

118_08 jump

118_09 jump

118_10 jump

118_11 jump

118_12 jump

118_13 jump

118_14 jump

118_15 jump

118_16 jump

118_17 jump

118_18 jump

118_19 jump

118_20 jump

133_03 walk and jump

133_04 walk and junp

133_05 walk and jump

133_06 walk and jump

13_39 jump at same place

13_40 jump at same place

13_41 jump at same place

13_42 jump at same place

105_39 jump forward

105_40 jump forward

105_41 jump forward

105_42 jump forward

91_39 jump forward

91_40 jump forward

91_41 jump forward

85_09 Motorcycle

138_11 Story

132_42 walk with rotating shoulders

81_06 Push

69_06 walk and turn

40_06 climb sit step over and have walks in between these actions

91_19 march

10_01 Kick Ball

49_06 cartwheel

62_18 opening box

62_19 closing box

64_01 swing

74_07 lifting up

75_13 hopscotch

79_01 chopping wood

79_08 boxing

79_33 chug a beer

79_72 crying

79_95 rowing

82_01 static pose

09_01 run

143_25 waiving

143_08 jumping twists

141_17 walk and sit

141_21 shrug

141_22 high five

141_23 shake hands

141_27 mope

139_05 pulling a gun

134_01 duck under

123_01 walk and carry

113_02 bow

113_03 curtsey

113_12 peekaboo

90_08 sideflip

90_34 wide leg roll

88_01 backflip

REFERENCES

[1] S. Dyer, J. Martin, and J. Zulauf, *Motion capture white paper*, 1995.

[2] M. Gleicher, "Retargetting motion to new characters," in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. ACM, 1998, p. 42.

[3] J. Barbic, A. Safonova, J. Y. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard, "Segmenting motion capture data into distinct behaviors," in *Proceedings of Graphics Interface 2004*. Canadian Human-Computer Communications Society, 2004, pp. 185–194.

[4] B. Bodenheimer, C. Rose, S. Rosenthal, and J. Pella, "The process of motion capture: Dealing with the data," in *Computer Animation and Simulation*, vol. 97, 1997, p. 318.

[5] Y. Takebayashi, K. Nishio, and K. Kobori, "Motion synthesis with motion splicing."

[6] C. Rose, B. Guenter, B. Bodenheimer, and M. F. Cohen, "Efficient generation of motion transitions using spacetime constraints," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, pp. 147–154.

[7] O. Masoud and N. Papanikolopoulos, "A method for human action recognition," *Image and Vision Computing*, vol. 21, no. 8, pp. 729–743, 2003.

[8] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popovic, "Style-based inverse kinematics," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 522–531, 2004.

[9] A. J. Hanson, "Visualizing quaternions," in *ACM SIGGRAPH 2005 Courses*. ACM, 2005, p. 1.

[10] "Scholarsarchive at oregon state university: Classification of motion capture sequences," id: 1.

[11] J. Lander, "Working with motion capture file formats," *Game Developer*, vol. 5, no. 1, pp. 30–37, 1998.

[12] I. Jolliffe, *Principal component analysis*. Springer verlag, 2002.

[13] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society.Series B, Statistical Methodology*, pp. 611–622, 1999.

[14] O. Arikan, D. A. Forsyth, and J. F. O'Brien, "Motion synthesis from annotations," in *ACM SIGGRAPH 2003 Papers*. ACM, 2003, p. 408.

[15] M. Mller, T. Rder, and M. Clausen, "Efficient content-based retrieval of motion capture data," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 677–685, 2005.

[16] M. Mller, *Information retrieval for music and motion*. Springer-Verlag New York Inc, 2007.

[17] W. Redfern, L. Carlsson, A. Davis, W. Lynch, I. MacKenzie, S. Palethorpe, P. Siegl, I. Strang, A. Sullivan, and R. Wallis, "Relationships between preclinical cardiac electrophysiology, clinical qt interval prolongation and torsade de pointes for a broad range of drugs: evidence for a provisional safety margin in drug development," *Cardiovascular research*, vol. 58, no. 1, p. 32, 2003.

[18] A. W. Fu, E. Keogh, L. Y. H. Lau, C. A. Ratanamahatana, and R. C. W. Wong, "Scaling and time warping in time series querying," *The VLDB Journal*, vol. 17, no. 4, pp. 899–921, 2008.

[19] L. Kover, M. Gleicher, and F. Pighin, "Motion graphs," in *Proc. of SIGGRAPH*, 2002.

[20] J. Gower and G. B. Dijksterhuis, *Procrustes problems*. Oxford University Press, USA, 2004.

[21] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *Readings in speech recognition*, p. 159, 1990.

[22] "Introduction to 3d animation," id: 1.

BIOGRAPHICAL STATEMENT

Harnish Bhatia was born in India on 9^{th} November 1984. She did her Bachelors of Engineering from Rajeev Gandhi Prodyogiki Vishwavidhyalaya, Bhopal in May 2006. She obtained her Master of Science degree from the University of Texas at Arlington in May 2010.

Her current research interests include Computer Animation, Computer Graphics and Vision.