Rhythmic Analysis of Motion Signals for Music Retrieval

By
Qiushi Li

A Thesis Submitted to
Saint Mary's University, Halifax, Nova Scotia
in Partial Fulfillment of the Requirements for
the Degree of Master of Science in Applied Science

October, 2008, Halifax, Nova Scotia

Your file Votre référence
ISBN: 978-0-494-46943-9
Our file Notre référence
ISBN: 978-0-494-46943-9

Certification

Rhythmic Analysis of Motion Signals for Music Retrieval

by

Qiushi Li

A Thesis Submitted to Saint Mary's University, Halifax, Nova Scotia,
in Partial Fulfillment of the Requirements for the
Degree of Masters of Science in Applied Science

October 29, 2008, Halifax, Nova Scotia

Examining Committee:

Approved: Dr. Vlado Keselj, External Examiner
Faculty of Computer Science, Dalhousie University

Approved: Dr. Sageev Oore, Senior Supervisor
Department of Mathematics and Computing Science

Approved: Dr. Norma Linney, Supervisory Committee Member
Department of Mathematics and Computing Science

Approved: Dr. Stephen Brooks, Supervisory Committee Member
Faculty of Computer Science, Dalhousie University

Approved: Dr. Pawan Lingras, Program Coordinator

Approved: Dr. Kevin Vessey, Dean of Graduate Studies

Abstract

Rhythmic Analysis of Motion Signals for Music Retrieval

By Qiushi Li

This thesis presents a framework that queries a music database with rhythmic motion signals. Rather than the existing method to extract the motion signal's underlying rhythm by marking salient frames, this thesis proposes a novel approach, which converts the rhythmic motion signal to MIDI-format music and extracts its beat sequence as the rhythmic information of that motion. We extract "motion events" from the motion data based on characteristics such as movement directional change, root-y coordinate and angular-velocity. Those events are converted to music notes in order to generate an audio representation of the motion. Both this motion-generated music and the existing audio library are analyzed by a beat tracking algorithm. The music retrieval is completed based on the extracted beat sequences.

We tried three approaches to retrieve music using motion queries, which are a mutual-information-based approach, two sample KS test and a rhythmic comparison algorithm. Feasibility of the framework is evaluated with pre-recorded music and motion recordings.

October 29, 2008

ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude to Dr. Sageev Oore, for his supervision, advice, and guidance from the very beginning of this research. He conducted me through this very interesting topic with rich knowledge and intelligence on both science and music. His truly scientist attitude and passions had inspired me and helped my intellectual growth, which I could benefit from for a life long time.

I gratefully acknowledge Dr. Stephen Brooks, Dr. Norma Linney for their advice and instructions. Their involvement on the research helped me open my mind to look wider and further. I would also like to thank the external examiner, Dr. Vlado Keselj, for using his precious time to read this thesis and give his critical comments.

Furthermore, I convey my appreciation to Dr. Pawan Lingras and everybody in the Computing Science Department. Their constructive suggestions and support helped me complete this work through these remarkable and pleasant years.

Finally, I would like to thank my parents and friends for their unconditional support, patience and constant encouragements, which helped me to overcome every difficulty and complete this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

Expanding internet bandwidth and storage media together with other advances have made large multimedia collections prevalent. Technology and algorithms for audio and visual data indexing and retrieval have attracted increasing interest [39].

Exploration of such collections is often based on metadata, such as artist, title or album, if this information is available. Many popular tools, including Media Player, RealPlayer, Winamp and iTunes, use conventional folder-based organization and textual list interfaces to represent the digital library. Although users have become accustomed to searching for music based on textual information, there is clearly an opportunity for non-verbal queries, and there has been a growing body of work in this domain as well [2][22].

Music Information Retrieval (MIR) is an interdisciplinary research area that emerged

to fulfill a wide variety of users' needs. Different from text information retrieval, MIR encompasses a number of different approaches aimed at music management, easy access, and enjoyment. Recently, different content-based MIR technologies have been developed [11][40], intending to data-mine music libraries based on structures or patterns in the music. However, content-based retrieval of music, according to overall sound similarity, is still an area with many challenges. For example, no music identification algorithms are robust to all situations; one solution might be more sensitive for classifying drum rhythms but doesn't work for light music [40]. Furthermore, while processing hi-fidelity audio and video can still be resource-intensive, the music identification system should be scalable to large music databases. This requires the optimization of algorithms in order to achieve a computationally efficient indexing [26]. A variety of research has been done to organize and query video and motion-captured libraries [31].

In this thesis, we consider the problem of being given a motion signal (ideally, a video, but in practice we work with a motion-captured signal), and then finding a piece of music from a given music library that will match the given motion. In other words, the motion signal is used as the "query" for searching in a music library. We focus on one particular element of this problem: rhythmic similarity. That is, we look for rhythmic elements in the motion, and try to find music pieces that have matching rhythmic patterns.

Rhythmic motion can include anything from a bouncing ball to video or animation of characters walking, dancing and certain sports games. Such motion typically con-

tains movement signals with certain repetitive patterns and temporal features, which are the bases of motion analysis and reconstruction. Researchers have developed techniques to extract various features from visual data for the purposes of motion reconstruction/adaptation. Such features include motion boundary [24], motion energy (spectral domain) [29][52], and movement center of mass [46]. In our case, however, we aim to analyze the motion signal for its rhythmic characteristics, and therefore need to extract a potentially different set of features for that purpose.

Our primary goal in this thesis is to develop a framework that allows us to use motion signals as queries to a music library. We do this by extracting "events" from motion signals based on an expandable set of features (defined in Section 3.1). The motion signal thus gives rise to a "vector" of events, related to the underlying rhythm of the motion. We then use this rhythmic representation as a query to the music database, by mining the music library for pieces with a similar rhythmic structure.

The framework presented here matches a motion animation with a music piece by rhythm similarity, which could be used as a video editing tool or music management application, such as the background music selection tool in a video game. Another possible application could include creating soundtracks for home video collages.

1.2 Related Work

In recent years a number of multimedia retrieval systems have been developed. In many of them, the same type of media is presented as both queries and results. For example, the system retrieves a similar audio file for an audio query [2][10], or

a similar image for an image query. Some researchers present methods that retrieve digital files by metadata, such as title or genre. Most of the approaches are based on content similarity such as melody, rhythm or tonality [40]. In our work, we focus on the comparison and retrieval between music and animated motion files based on content similarity.

From a data processing perspective, music data has many features, some of which are clearly "user recognizable" (e.g. certain melodic or rhythmic characteristics), and others of which are less so (e.g. certain spectral characteristics). These various features and characteristics of music data can, in fact, be used as the bases for audio retrieval techniques. For example, several music retrieval systems have been proposed based on vocal percussion [25], or harmony [42]. Doraisamy and Rüger [10] presented an N-gram approach to index data and to improve scalability to retrieve melodies. Their systems are developed for polyphonic MIDI music with pitch and rhythm information. Similar research could be found in [11][35]. An alternative approach is to represent queries by humming and formulate all music in a collection as sequence data, such as symbolic melody sequences and pitch sequences, and then use sequence matching techniques to retrieve a precise musical work [2][22][48]. There is also a music retrieval system built on a geometric model, which is applied to compute the similarity between a music query and the pieces in the library [33].

Several authors introduced distance metrics for the audio data's spectral parameters, or spectral parameters combined with other perceptual information [15][32]. Foote and Uchihashi's proposed the notion of "beat spectrum", a measure of acoustic

self-similarity as a function of time lag [16], and various other subsequent research has used this approach to sort and retrieve audio data [13][15]. Cheng Yang presented an algorithm [51] to retrieve music pieces by spectral similarity. In his work, audio files are represented by spectral vectors calculated from the signal power. After that a specially defined distance function is used to sort the results according to linearity criteria to select the best matched piece.

In this thesis, music retrieval includes both types of music features. Music files are represented by their beat information, while beat sequences are calculated through the audio data's spectral flux [7]. Instead of audio data, queries of our framework are animated motion sequences.

Although some researchers have addressed the issue of matching audio pieces with visual data, very few of them use motion signals as queries. Hua and his colleagues introduced the method of automatic video editing by finding video boundaries and then aligning incidental music to those segments [24]. In their system, video boundaries are detected by segmenting shots and sentences of the sound track, while music files are analyzed and beats are extracted by detecting strongest onsets within a time window. Finally, the system constructs music videos by aligning video shot boundaries with music beats and sentence boundaries. Another example is the model proposed by Yang and Brown which queries a music database with an MPEG video [52]. The algorithm treats both music and video as time series data, and compares these two sequences based on synesthesia effect, a similarity measure for corresponding features in the music and video sequence, which in particular, refer to the tempo of music files

and a motion vector extracted directly from the MPEG compressed video stream. Similar research work also contributed to reveal the relationship between audio and video data [21][14][50], such as music features like the pitch, amplitude or tempo, and color or shape in the video shots. Recent work has presented various algorithms of detecting video boundaries, which could be considered rhythmic elements [1][23][38].

In computer animation, the problem of automatically matching visual representation with audio has not been heavily explored. Among the existing work, a common approach is to generate or modify the animation or music by changing the timing of the object media [30]. Background music is used as a supplementary motion feature to extract motion structure. For example, Shiratori et al. proposed a motion capture and analysis system which records the speed of hands and center of mass [46]. They combine the above features and the background music's rhythm information (beat onset times) to segment dance motion-captured signals, and also synthesis the motion signal to music. Some researchers have proposed algorithms to extract the percussive pattern from motion files. For example, the system presented by Lee et al. extracts repeating rhythmic patterns from the dance motion data using both spectral (frequency impulse analysis) and acceleration-onset analysis [29]. Kim et al. defined the concept of "motion-beat"[27], which are detected by analyzing and grouping repetitive movements from the motion signal. Their proposed system could match newly composed motion to background music based on motion graphs [28]. According to their description, the motion beat is a regular, rhythmic unit of time for a motion and the rhythmic pattern is a sequence of motion beats corresponding

to a motion unit. Another motion rhythmic analysis system was presented in [29], which extracts motion impulse by combining spectral and spatial analysis of motion onset. In this thesis, we proposed a different concept of "motion beats" in Section 3.

In Penasse and Nakamura's study, motion data has been customized so that the motion pattern is synchronized with the background music's rhythm [41], while some other researchers considered the complementary problem of how to modify the music to achieve synchronization with a pre-existing motion [9][19][37][53]. An example is the system that Cardle and Brooks presented [3], which has the purpose of synchronizing motion signals to music. The proposed system modifies existing motion according to perceptual parameters extracted from music, including MIDI sound track and converted analogue audio. Motion curves are then mapped to the multiple parameters to generate animation. They also presented a soundtrack generating system [4] based on the controllable model extracted from the original animation. This work enables a user to select and build the relationship between sound(s) and motions and then generate a soundtrack given new motion files.

1.3 System Overview and Outline of the Thesis

The objectives of this research include analyzing object movement signals, extracting rhythmic parameters from both motion and music sequences, and applying similarity measures on them. The goal of our research is to query a music library with motion captured data.

Our framework has three fundamental components: motion analysis, music anal-

ysis and music retrieval. Rhythmic characteristics are extracted from motion signals and music recordings individually. In the music retrieval phase, a motion file is provided as a raw query. We sort music pieces of the collection by content-based similarity scores, comparing them with query motion file. A conceptual model of the system is shown in Figure 1.1.

1.3.1 Motion Signal Analysis

Motion-captured recordings of 3 degree-of-freedom (albeit complex and noisy) rhythmic motions are provided for analysis in this part (data collection and preprocessing is described in Section 2). Different types of events are extracted from the motion sequences based on characteristics such as velocity, movement directional change, angular velocity and curvature. A motion file is then represented by a series of time-stamped events, which is similar to the structure of MIDI audio (MIDI is the series of time-stamped note events). In terms of this similarity, we convert such event-represented-motion-sequences to MIDI files for further analysis. A detailed explanation of motion signal analysis is given in Chapter 3.

1.3.2 Music Beat Analysis

We apply an existing music beat tracking algorithm "BeatRoot" [7] on both music pieces and motion-generated MIDI files (Section 4). The description of the beat tracking algorithm is in Section A. This algorithm takes wav format music as input. Hence, to use this algorithm, motion-generated MIDI sequences are converted to

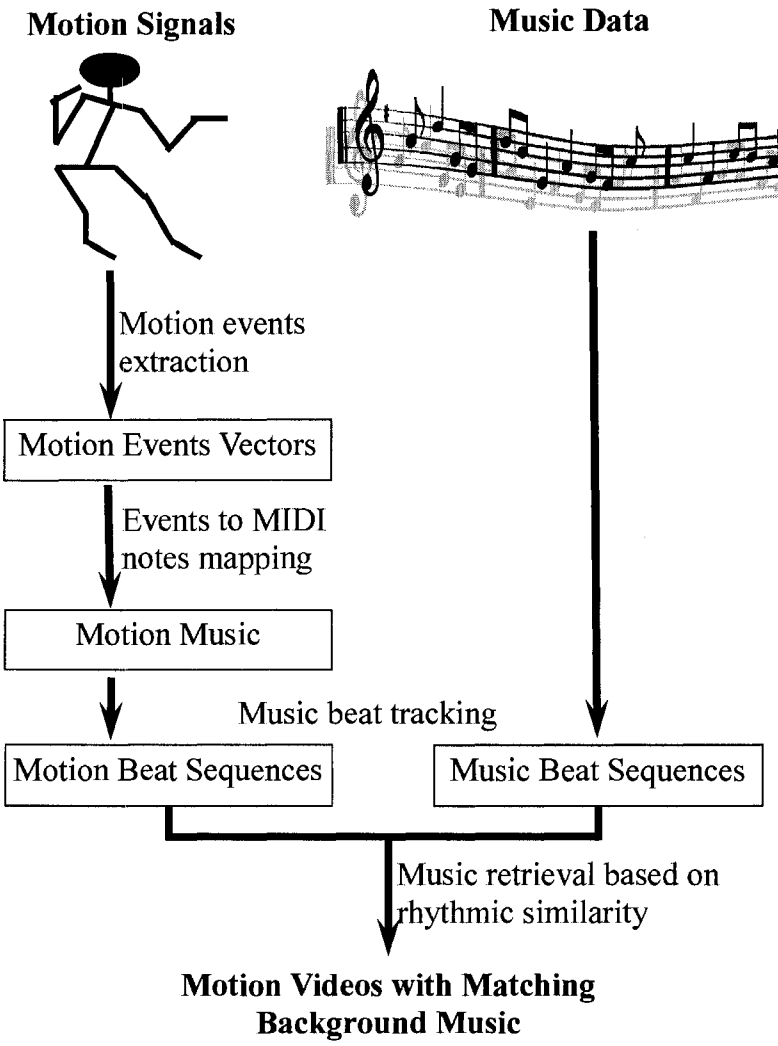

Figure 1.1: An overview of the motion signal analysis and music retrieval process

audio (wav) files before applying the rhythmic analysis algorithm. The outcome of this phase is each music file's corresponding beat sequence. It is represented as a time-series of events (music beats). The rhythmic representation of motion video is generated in the same manner.

1.3.3 Music Retrieval

Rhythmic information of the motion recording is provided as above. The system analyzes and compares it with the beat information of the music in the library, retrieves a single music piece whose beat is most similar to the rhythm of the motion recording. In this thesis, we compared a few different approaches for estimating similarity between the rhythmic features of the motion-generated signal and those of the available music-library data. The approaches we tried included: a mutual-information based metric (Section 4.1), a non-parametric K-S measure (Section 4.2), and a rhythmic comparison metric that we developed (Section 4.3).

In this work, our focus was on demonstrating the potential efficacy of this framework. While outside the scope of this thesis, it is very likely that a more customized rhythmic comparison metric might be even more effective than any of the particular approaches we had tried. Nevertheless, all three metrics demonstrated the validity of the approach.

The system is evaluated on pre-recorded motion recordings and music data. Using each of the motion recordings as a query, a music piece is retrieved. The results of the three different retrieval methods are described in Chapter 5. We also generated some

demonstration animations based on the results so that the results could be visually and aurally evaluated.

Chapter 2

Data Collection and Preprocessing

Our system requires two sources of data: music files and motion files. We collected data both from free on-line resources and also by recording our own. There are two other transitional sources of data: the midi and audio files derived from the provided motion data. However, these will be described in more detail in Sec 3.2.

2.1 Music Data

In order to apply the existing beat tracking algorithm "BeatRoot" [7] which accepts .wav format music files as input, we converted the given music library into a raw wave form format (.wav). The files in the library contain some short clips (several seconds long), and also two to three minute-long recordings. Various styles of music were assembled, including classical, ragtime and popular music.

Besides these existing pieces, we also recorded music ourselves, to ensure, somewhat systematically, a range of different speeds, numbers of tracks and time meters for

testing purposes. For example, we created and/or recorded pieces whose underlying time signatures are: $2/4, 3/4, 5/4$ and $7/4$.

We also combined different tracks and/or different numbers of tracks of melody or rhythm of the same musical piece to generate additional variations. For example, for the piece "Take Five" (a tune by Dave Brubeck in 5/4) we used a metronome to maintain the tempo, and then recorded two different bass lines, a rhythmic accompaniment, and one melody. We then took various permutations of these elements to generate four distinct recorded pieces, e.g. bass-1 only, bass-2 + rhythmic accompaniment, bass-2 + rhythmic accompaniment + melody, etc. The goal of recording variations of the same music piece is to test the beat-analysis algorithms to determine, which rhythmic element or elements of a music piece are more influential in beat induction. For example, if the extracted beat sequence of recording bass-1 and that of bass-1 + rhythmic accompaniment are similar, we can deduce that the accompaniment track is less important in music structure analysis than the bass track.

2.2 Motion Data

As with the music data, we acquired our motion data from two sources: one was a free on-line motion capture library, and the other was by capturing motion data ourselves.

The public library data was obtained from the Graphics Lab Motion Capture Database of Carnegie Mellon University. This type of data is captured human-character motion data, which contains annotated main body joint tracks in 3-D po-

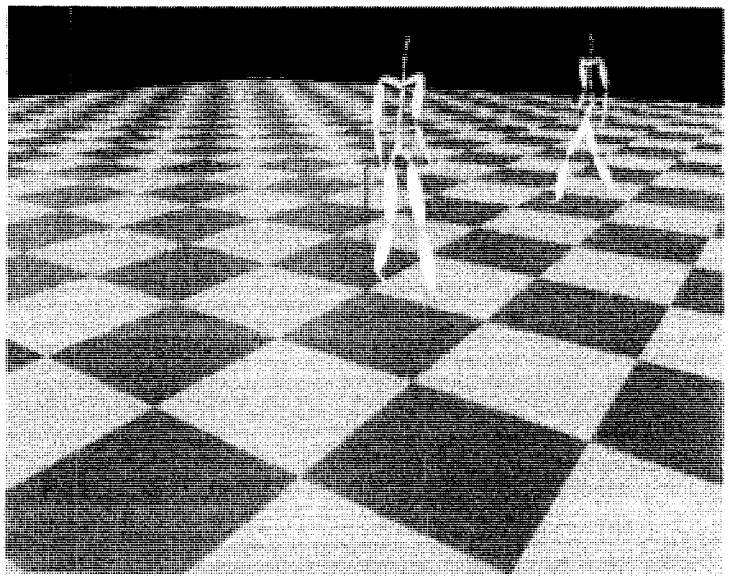

Figure 2.1: Screen shot of skeleton animation from CMU motion capture database

sition, and the corresponding skeleton animation (Figure 2.1).

The rest of our motion data were recorded in our own lab using a Polhemus Isotrak, a motion capture device consisting of two trackers, each one measuring 6-DOF (Figure 2.2). Our recording configurations included sessions with one and both trackers. The trackers were used to record hand movements in synchrony with background music. The "dancer" is familiarized with the music in advance in order to perform the rhythmic movements. Three degrees of freedom from each sensor (the spatial positions relative to a predefined origin) are recorded.

Due to the characteristics of motion capture devices and sources, there could be slight difference of the data format and quality, such as different sampling rates

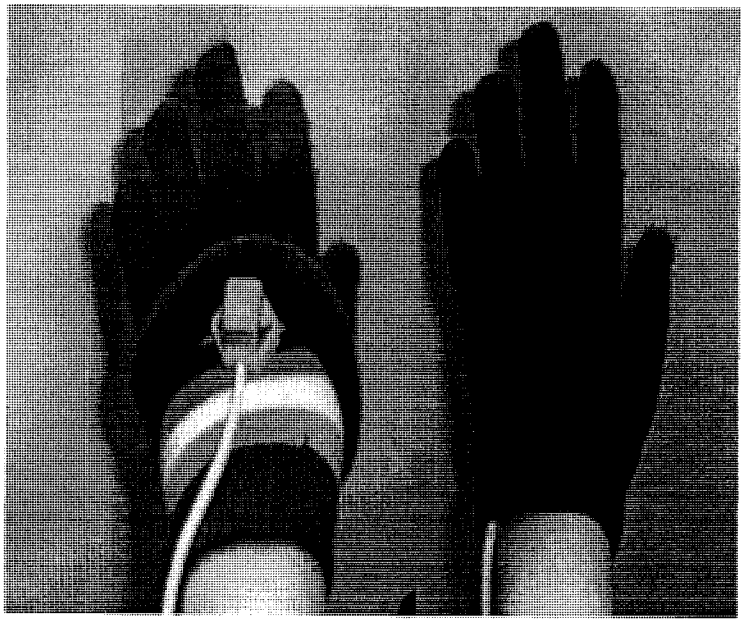

Figure 2.2: Polhemus 3D motion capture sensors are attached to the gloves. (Only the left hand is shown)

and different types of noise. In this project, we format the tracked motion data as 3-D position (x, y, z triplets) together with time information at each point for a single moving object. The motion captured data from Polhemus device is already smoothed. Thus for the noisy data, we smooth it with a moving average filter over a time-window w_1, of which the size is manually decided according to the noise of that data. Generally, the width of w_1 is different for each motion recording device. It is determined by testing and observation, namely, the processed data should be smooth enough to take off the noise while keeping the characteristics of raw data. This principle is employed whenever a new source of data is applied to our system. Figure 2.3 shows an example of raw data we used. It is the plot of the x coordinates over time. Figure 2.4 shows the same data from Figure 2.3 but filtered by w_1 of size 0.2s. The noisy property of the raw data set is removed while the peaks and bottoms of the curve are kept.

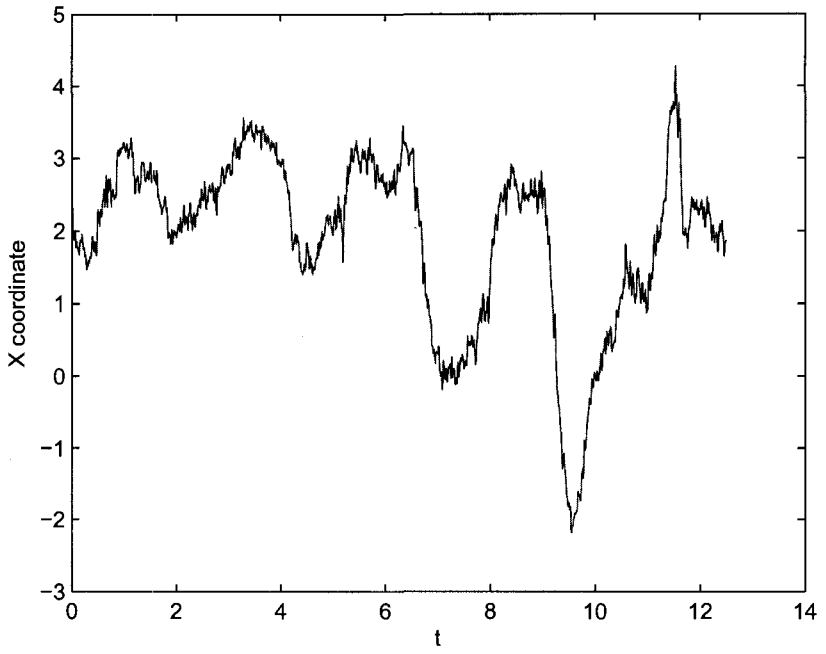

Figure 2.3: Original recorded motion data, the plot of x coordinates with time vector

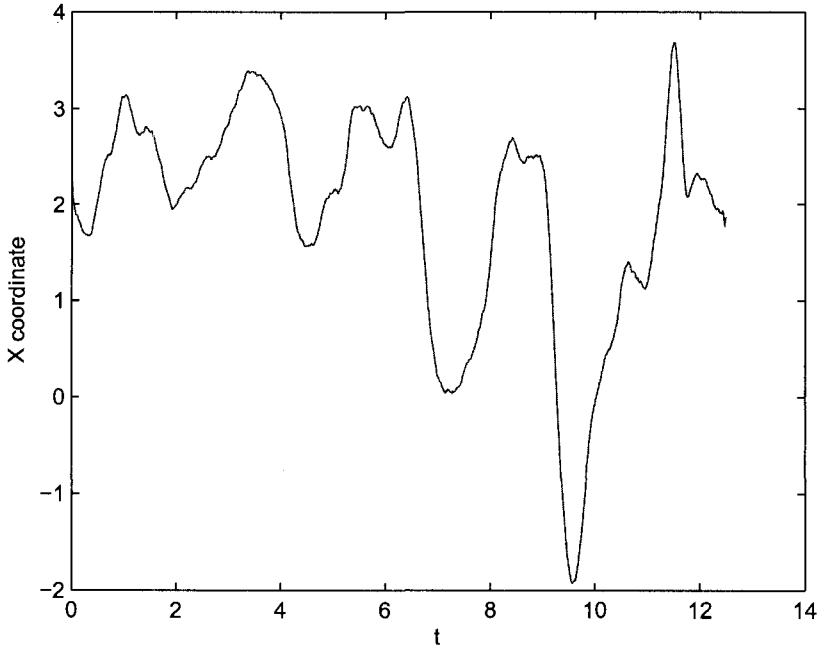

Figure 2.4: Recorded motion data from figure 2.3, smoothed with $w_1 = 0.2s$, the plot of x coordinates with time vector

Chapter 3

Rhythmic Motion Signal Analysis

Rhythm has the attributes of continuity or flow, articulation, regularity, proportion, repetition, pattern, alluring form or shape [20]. Music has an essential connection with rhythm. According to Hasty, music is the "rhythmization of sound". Such structured temporal patterns are also contained in motions, and are especially obvious in human dancing and cyclic motions such as marching, running and walking. All of these could be considered simultaneous responses to external rhythmic signals such as background music or drum beats [27]. Note that it is not necessarily music that actually caused the physical motion, but still there is some connection between them. Based on these observations, we assume that the structured repetitive patterns in a motion file could reveal the underlying rhythm of that motion signal as rhythm of the associated music. Consequently, we could extract the "beats of motion", by analyzing the regular (and irregular) temporal patterns in rhythmic motion signals.

Therefore, we introduce the notion of a "motion event": a motion event represents

the moment of an apparent movement or an evident gesture in a motion, which we interpret as a basic unit of rhythmic motion patterns. From each motion file, we can compute a series of motion events. Hence, we define the concept of "motion beat"[1] to be a regular, rhythmical unit of time for a motion event sequence. That is, while the events themselves may not be regular, just as a rhythmic musical passage is usually not perfectly regular, we assume there is nevertheless an underlying regular pulse of motion beats. We propose that those movements that generate motion events can be detected by analyzing the motion signal and finding certain characteristics or patterns in it. In this thesis, we detect motion events by using the following motion characteristic extraction methods.

3.1 Extracting Motion Events

Our motion signals are discrete samples that describe sequences of moving object positions. Based on the assumption that the object's behavior would be different at some beats than ordinary time points for rhythmic performance sequence, we extract those important behaviors to detect motion beats. In order to detect the underlying rhythm from the motion, we define a set of distinct motion event types, which could be represented by:

$$EVENT = \{event_i, 0 \leq i < M\}, \tag{3.1}$$

[1]Note that Kim et al [27] defined motion beat in a similar, but subtly different way. They described motion beat to be a regular, rhythmical unit of time of a motion signal; whereas our beat is based on the series of events.

where $event_i$ refers to one particular type of event and M indicates the total number of event types. We assume that zero or more different types of events would occur at the candidate motion beats. Each type of event is defined by certain features of the motion data and can be specified by a movement pattern, such as fast directional change, free-falling and so on. For example, an $event_i$ takes place at frame j if the angle between one object's movement directions at frame $j-1$ and j exceeds a certain threshold. More generally, a set of constraints are associated with each event type, and an event is said to occur at the frame at which its constraints are satisfied, as will be seen below.

In our system, these constraints are specified based on our observations and heuristics, rather than by an automated process. We have not tried to create a comprehensive feature list, but rather a sufficient list to demonstrate the feasibility of our framework. To identify the significant motion events, it is helpful to extract a set of basic motion-related features from the captured motion data. We now describe this set of basic features.

3.1.1 Minimum Velocities

The local minimum velocity value can help us identify visible pauses. To find such points in the motion, we calculate the instant velocity at time t by:

$$\vec{v}(t) = \frac{ds}{dt} \tag{3.2}$$

In the equation, ds is separated into the position changes of three spatial directions (dx, dy, dz) and each velocity vector is composed of scalar velocity values in 3 spatial axes (v_x, v_y, v_z).

Local minimum velocities could be found by the zero-crossing points of velocity values' first-derivative $v'(t)$ while $v'(t)$ is in its increasing phase, which is shown in following calculation:

$$v'(t) = 0 \text{ and } v'(t_i - 1) \leq v'(t) \leq v'(t_i + 1) \tag{3.3}$$

A visible pause during a motion is the moment when the instant absolute velocity of the moving object drops to near zero. Hence, we add the condition:

$$|\vec{v}(t)| \leq 10^{-3} \tag{3.4}$$

to detect the such events, in which $|\vec{v}(t)|$ is calculated by:

$$|\vec{v}(t)| = \sqrt{v_x(t)^2 + v_y(t)^2 + v_z(t)^2}$$

3.1.2 Movement Directional Change

Movement directional change is the angular change between an object's movement direction at two time points (Figure 3.1).

Initially, we calculated the object's directional change at frame i using the object's direction of motion at frame i and frame i+1. An object's directional change curve was

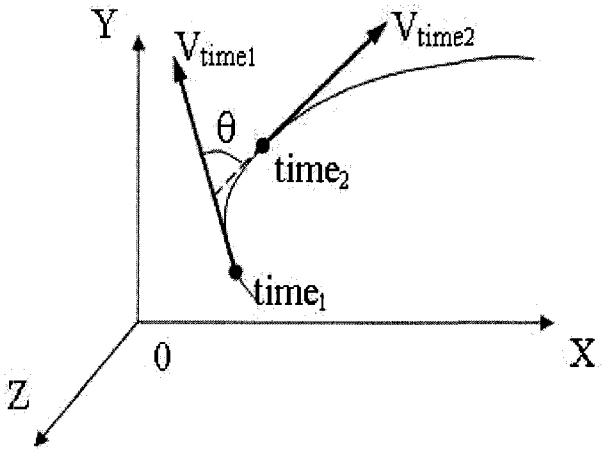

Figure 3.1: Movement directional change θ

formed by calculating the directional changes at each pair of adjacent frames. The time duration between adjacent frames is 1/120 second for CMU motion captured data, and 1/15 second for recorded motion data. Because the object's direction could be expressed by its instantaneous velocity, we used the directional change angle between adjacent frames' velocities to represent directional difference. That is,

$$\theta_i = \arccos\left(\frac{\vec{v_i} \cdot \vec{v_{i+1}}}{|\vec{v_i}| \cdot |\vec{v_{i+1}}|}\right) \tag{3.5}$$

where $\vec{v_i}$ and $\vec{v_{i+1}}$ are the moving object's velocities at frame i and frame $i+1$. The calculation of the velocity vector is the same as in equation 3.2.

However, in some cases the moving object's velocity was low throughout the whole process, which made the curve flat and smooth. Or in some cases, high frequencies or the sensitivity of motion capture device can result in non-continuous motion data. The directional change, which is the difference of motion direction curve, calculated from those data are fluctuating in a narrow range between zero and the actual angle value. For example, Figure 3.2 shows the directional change curve computed between adjacent frames of basketball hand motion data set 1(from CMU mocap database). The value of the object's directional change was fluctuating between very small and very large values. The zero values on the directional change curve were not the necessary representations of the object's movement, which made the curve noisy and inappropriate for further processing.

To deal with this issue, we set up a small time interval $\triangle T$, such as 0.1s or 0.15s, and calculated the angle between velocities spanning the length of that time window.

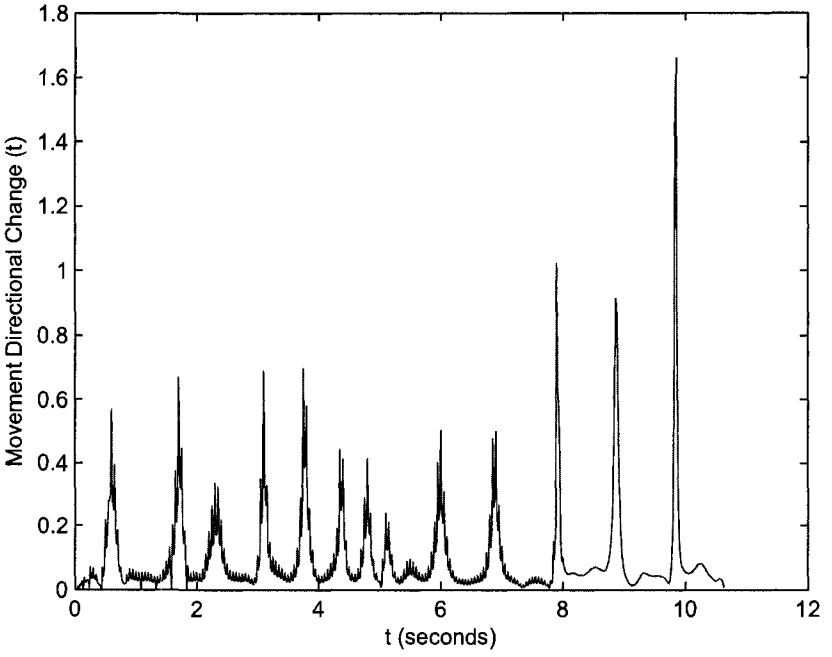

Figure 3.2: Movement directional change curve computed between adjacent frames from basketball human hand motion data set 1, right hand

Thus, we use the equation,

$$\theta_k(\triangle T) = \arccos\left(\frac{\vec{v_i} \cdot \vec{v_j}}{|\vec{v_i}| \cdot |\vec{v_j}|}\right), \tag{3.6}$$

to calculate the movement directional change at time frame k, in which

$$k = int\left(\frac{i+j}{2}\right), \text{ and } \triangle T = time_j - time_i$$

For example, Figures 3.3, 3.4 and 3.5 show the directional change curves calculated with time intervals of 0.05s, 0.1s, and 0.2s respectively of the right hand motion of basketball playing. From these figures we see that increasing the time interval in calculating the directional change curve leads to clearer directional change angles (strong peaks in the curve). Moreover, with the interval increasing, the areas without much directional change, that is, the places where the movement directional change angles have low values, are smoothed.

From the above examples we can see that, a larger duration results in more obvious peaks, higher angle values, and could also remove the "noisy" appearance of some curves. The length of the time interval $\triangle T$ is set up by comparing the different directional change curves resulting from using different time interval lengths. For calculation purposes, we would like to keep the directional change curve smooth but with proper number of directional change peaks. Hence, we pick the values which could generate the suitable curves. The length of $\triangle T$ ranges from 0.08s to 1.5s in our system. For motion recordings that full of fast movement directional changes, the

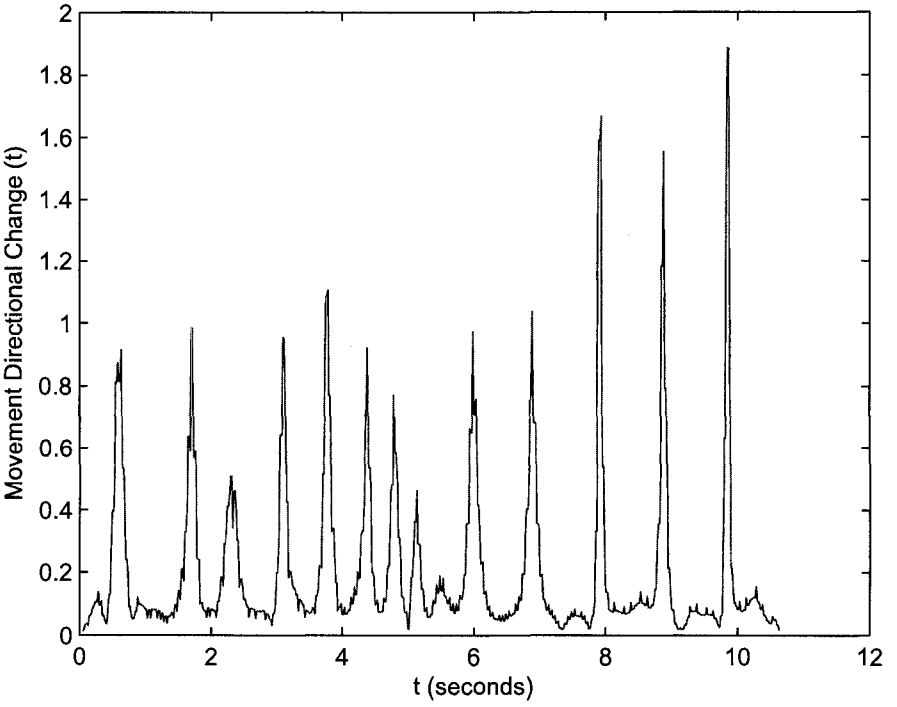

Figure 3.3: Movement directional change curve computed with a time interval of 0.05s from basketball human hand motion data set 1, right hand

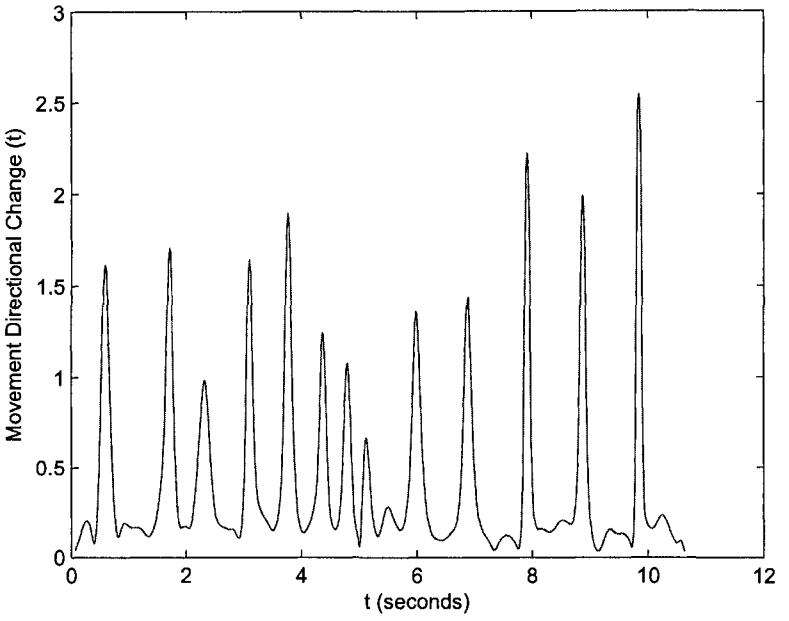

Figure 3.4: Movement directional change curve computed with a time interval of 0.1s from basketball human hand motion data set 1, right hand

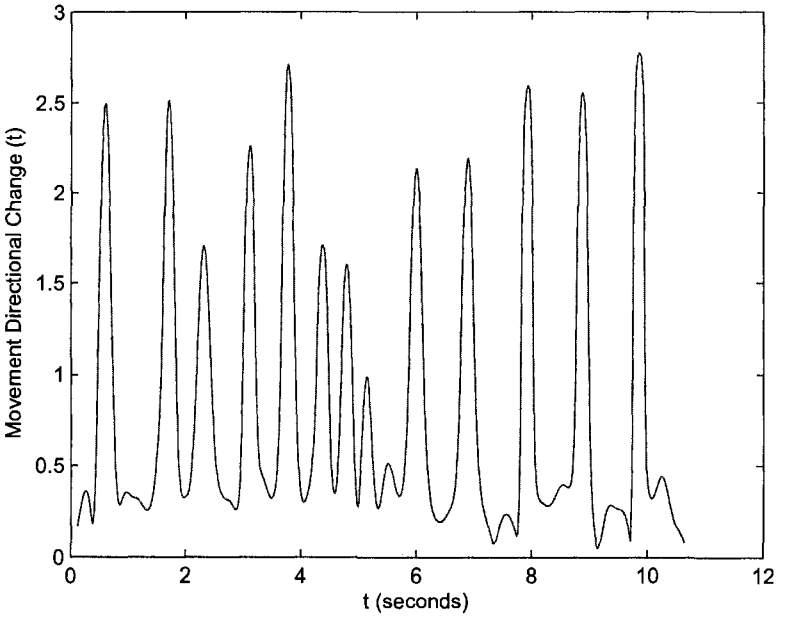

Figure 3.5: Movement directional change curve computed with a time interval of 0.2s from basketball human hand motion data set 1, right hand

time duration should be short to detect fast motion direction changes; for motion data with very flat or noisy movement directional change curve, such as the situation shown in Figure 3.3, we may slightly increase the time interval between target velocities to get more clear peaks on the curve.

Having computed the directional changes throughout the entire motion signal, we can now use the resulting curve to locate the most salient change-of-direction events. That is, the peaks on the directional change curve mark the moments that the moving object has the biggest directional changes. To identify these movements as motion events, a threshold has been set up on the directional change curve. Moreover, we use different levels of thresholds to separate the motion events into different levels by the degree of their directional change angles. These positions should have the property that the corresponding places in the directional change curve are peaks and their values exceed a threshold. Such places can be detected by the zero-crossings of the first derivative of the directional change angles. We use the expression,

$$\theta'_{i-1}\theta'_i < 0, \text{ and } threshold_j < \theta_i \leq threshold_{j+1} \tag{3.7}$$

to denote that $event_j$ occurs at time i. In this expression, $threshold_j$ and $threshold_{j+1}$ are the thresholds for $event_j$ and $event_{j+1}$ respectively. If the directional change curve is noisy, the resulting derivative will have redundant zero-crossing points which may add inaccurate events to the motion events vector and increase the computational cost. To avoid that situation, some noisy directional change curves need to be smoothed with a moving average filter before we do further analysis. The width of

filter is decided under the same consideration as that in the phase of data preparation. The smoothing function is simply,

$$\theta_i = \frac{1}{N} \sum_{k=-\frac{2}{N}}^{\frac{2}{N}} \theta_{i+k} \tag{3.8}$$

where N denotes the filter size.

The thresholds for computing these types of events are set up by the user given different datasets. It depends on the values on the directional change curve and the number of events that user would like to extract from the motion sequence. If the movement directional change angles have low values while we want to extract more events, we can set a lower threshold so that there could be more angle peaks exceeding that threshold. For example, we set the threshold to be $\pi/2$ for the simulated ball-bouncing motion recording, and $\pi/4$ for the basketball human hand motion.

3.1.3 Maximum Acceleration

Human dancers may have significant movements at the musical accents. Hence, there are possibly local maximum forces exerted at the same time as the background music's various metrical events. Computing maximum force can be done by computing maximum accelerations.

According to Newton's well-known second law of motion

$$\vec{F} = m\vec{a}$$

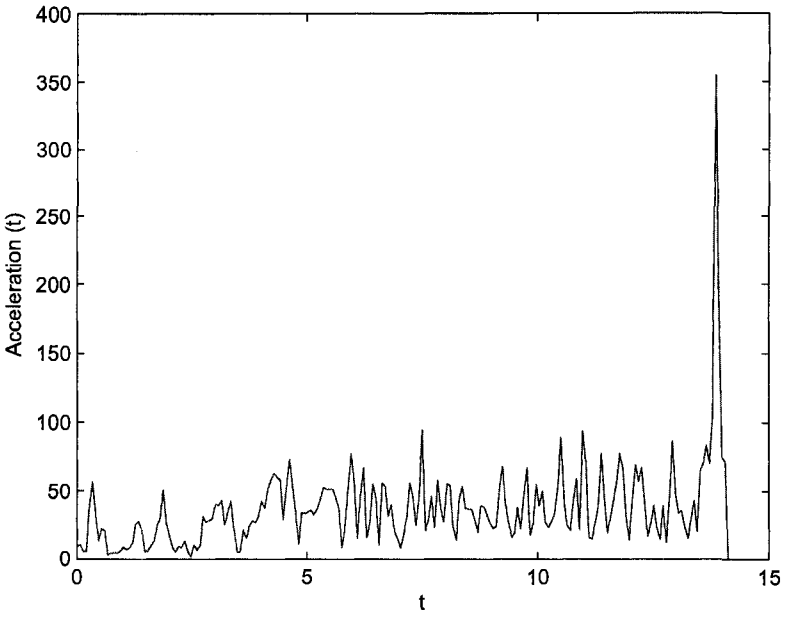

Figure 3.6: Acceleration values from one hand motion of motion capture data

the acceleration of an object is proportional to the resultant force acting on it and is in the same direction. Because a moving object's mass is constant, the force on the object can be presented by its acceleration, which is described by the following expression,

$$\vec{a} = \frac{d\vec{v}}{dt} \tag{3.9}$$

The calculation of $\vec{v}$ is the same as in equation 3.2. An example acceleration curve is shown in Figure 3.6.

Thus, we calculate local maximum accelerations (peaks in the acceleration curve of Figure 3.6) to detect the moments of maximal force. Local maximum values of acceleration can be calculated by the zero-crossing positions of the acceleration's first-derivative (also known as jerk) in the decreasing phase.

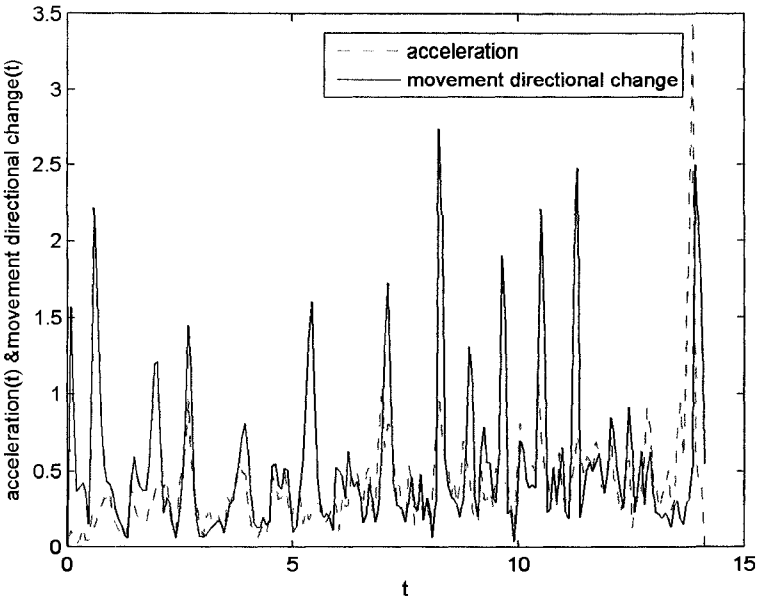

Figure 3.7: Acceleration and movement directional change angle comparison

During the calculation process, we found that the acceleration curve has a high similarity with the directional change angle curve (Section 3.1.2). An example of the comparison between acceleration and directional change is shown in Figure 3.7, in which both curves were calculated from motion data #233234 [2]. The $\triangle T$ used to calculate directional change is 0.15s. This picture indicates that the curves of acceleration and directional change have similar local maximum positions. Thus, the events generated from these two methods are close in occurrence time.

[2]This number refers to the motion ID of recorded motion data. See Table 5.2 for detailed description.

3.1.4 Curvature

The curvature of a smooth curve at a given point is the measure of how quickly the curve changes direction at that point. Great direction changes corresponds to a local maxima curvature values of the motion track. We compare the curvature at each point and mark the places with the local maximum curvature values as an event. The curvature of a point at time t is given by

$$k(t) = \frac{|T'(t)|}{|v(t)|}, \tag{3.10}$$

in which

$$T'(t) = \frac{v'(t)}{|v(t)|}$$

In these equations the definition of $v(t)$ is the same as in equation 3.2.

We calculate the local maximum values of the curvature by calculating the zero-crossing positions of its first-derivative in the decreasing phase. However, because of the fact that the plotted motion tracks from the captured data are not perfectly smooth curves, the resulting curvature values could be noisy in a narrow range like the directional change curve. To avoid the misleading result of detecting all the maximum values from the noisy curve such as in Figure 3.8, we apply a threshold on the calculation of local maximum values,

$$\max(k(t)) \geq threshold_{curvature} \tag{3.11}$$

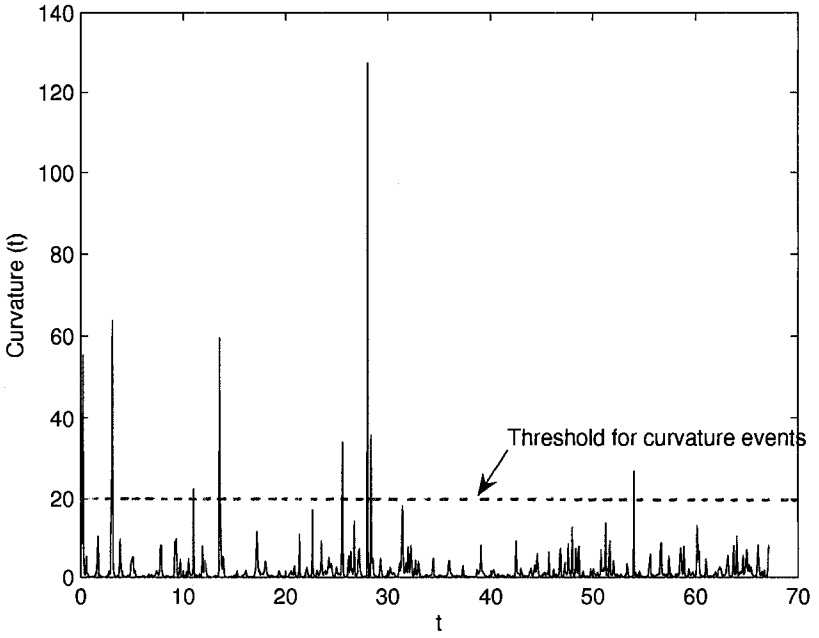

Figure 3.8: Curvature values from one hand motion of motion capture data. The dashed line represents the chosen threshold for generating events by maximum curvature value. The peaks above the threshold are events.

in order to extract the major maximum curvature values such as those shown in Figure 3.8, the curve peaks above the chosen threshold. Similar to the directional change curve, the value of curvature threshold is manually tuned according to each curvature curve to obtain proper number of peaks.

Like the acceleration curve, the curvature also has a similar shape to that of the directional change curve. Figure 3.9 shows the curvature and directional change curve (calculated with $\triangle T = 0.07$s) of motion #233234, from which we can see that the curvature is similar to the directional change. But the curvature tends to be noisier than the directional change. They actually detect very similar sets of events if the curvature is smoothed.

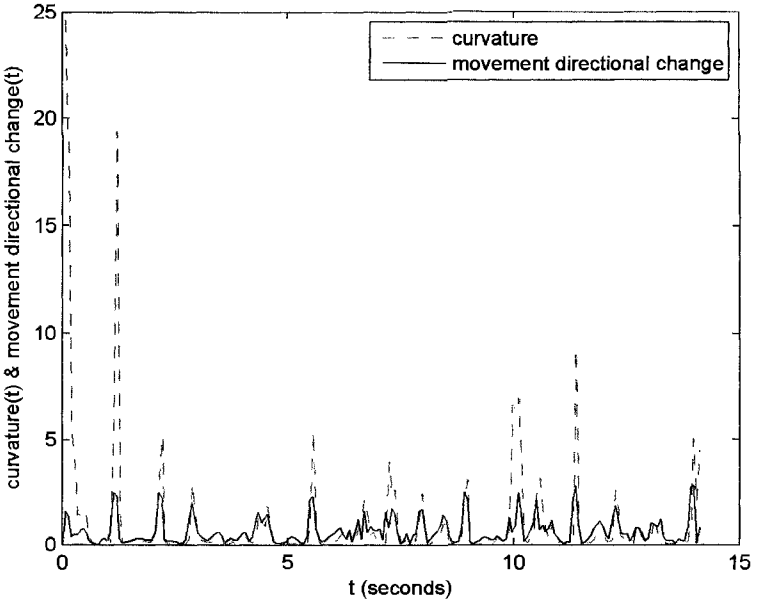

Figure 3.9: Curvature and movement directional change angle comparison

3.1.5 Root-y Coordinate

An important visual characteristic of rhythmic object movement is bouncing, a rhythmic vertical motion. In this project, we identify "bounces" by the moments when the bouncing object reaches its bottom and define these moments as events.

Since the rhythmic vertical movement is a feature of the entire moving object, we calculate the "root position", i.e. position of the root-node, of each object and analyze its motion. For simple rigid objects, the root position is the position of the object itself, for human characters such as in the motion captured data from CMU, the root position represents the body center (shown in Figure 3.10).

We analyze the y-coordinate of the root node, which represents the up and down movement of the object. An example is shown in Figure 3.11, which is the plot

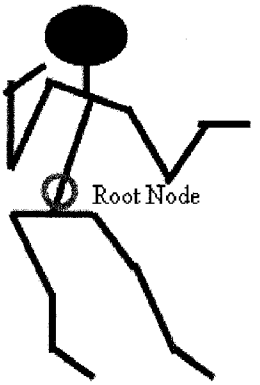

Figure 3.10: Root position (marked with an O) of human-character

of the root-y coordinate of a human walking motion recording. Bottom points of the y coordinates, which are the local minimum values of the curve, are marked as events since they reflect a rhythm of the motion (marked by red circle in Figure 3.11). The rhythmic motion features are more clearly reflected in cyclic motion such as a bouncing ball or human-character walking or running motion.

3.1.6 Constant Angular Velocities

Another special pattern we extract from the movement is when the object is moving in approximately circular motion. We can test for this pattern by looking at the angular velocities of the object. To calculate the angular velocity of the moving object, we use the movement directional change angle $\theta(\triangle T)$ with a time interval $\triangle T$ introduced

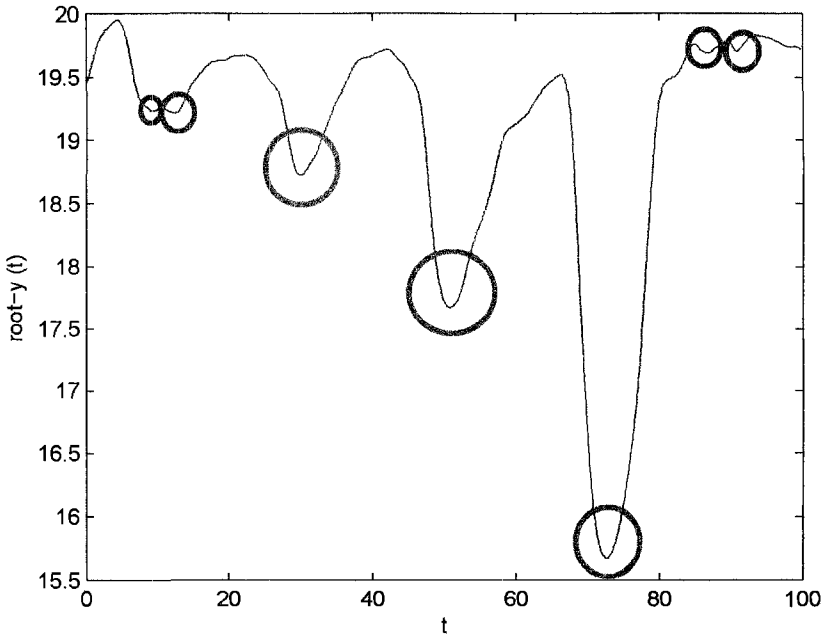

Figure 3.11: Root-Y coordinates, events could be extracted by root-y coordinate bottoms (circled positions).

in Section 3.1.2, for the reason that equal movement directional change angles in a consecutive time denotes constant angular velocities within that time duration. We calculate the difference of directional change for the time duration of k continuous frames. Ideally, the difference between directional change angles is zero for detecting this pattern. But due to the various noises, the difference cannot reach zero under most circumstances. We set up a threshold for the above situation, and use the following equation to calculate the change of moving object's angular velocities at frame i and $i+1$,

$$|\theta(\triangle T)_i - \theta(\triangle T)_{i+1}| < threshold, \quad i \in k \tag{3.12}$$

3.2 Motion Music Generation

We will use the term *motion music* to refer to the music generated from the motion recordings. This could be seen as an acoustic representation of the motion, or rather, of certain features of the motion. This concept is proposed based on our hypothesis that there is an underlying rhythmic pattern in a rhythmic motion signal. We define motion events as basic rhythmic elements of the motion, as music notes are to music. Just as an underlying beat can be inferred from a sequence of musical notes, we propose that an underlying beat can also be inferred from the sequence of rhythmic motion events. Therefore, the problem of extracting the underlying beats from a motion recording is converted to one of music beat tracking. Given the similarity in "rhythmic function" between motion events and music notes, we convert all the

motion events (in a motion event vector) into music notes, to generate motion music for that motion file. That is, *we create an audio representation of the rhythmic properties of a motion signal.* The purpose of this is that, we can use the existing beat tracking algorithm to analyze motion music, and then find similar music pieces in the collection by applying rhythmic similarity comparison methods between audio data. The detailed method of generating this is introduced in the following sections.

3.2.1 Testing with Motion Music

The features listed from Section 3.1.1 to 3.1.6 allow us to identify a large set of possible motion events, which might be caused by the background music or the rhythm in the movement itself. Events generated from different motion characteristic extraction methods might be very close in occurrence time to each other, which will make the resulting motion events vector overly dense. However, we would like to keep the generated motion music uncluttered to represent the underlying motion rhythm as accurately as possible, which requires the motion event vector to be as sparse as possible while maintaining salient rhythmic features. To test the above approaches and choose the ones that more accurately present the motion rhythm, we applied them to our captured motion data individually and in different combinations. Corresponding motion event vectors are generated each time, and then converted to music files as described below. Hence, we could listen to the motion music and watch the motion recording synchronously to subjectively distinguish between the qualities of the various combinations of motion characteristic extraction methods.

Motion characteristics	Music characteristics
Motion event	Music note
Motion event occurrence time	Music note onset
Motion event level	Music note amplitude, duration and pitch
Joint (for human-character motion data)	Timbre and pitch

Table 3.1: Mapping relationship from motion characteristics to music characteristics

In our implementation, we mapped motion characteristics and patterns into the basic characteristics of MIDI-formatted music, such as amplitude (volume), pitch and timbre (by choice of instrument). The mapping scheme is shown in Table 3.1.

By listening to the motion music synchronized with corresponding motion video animation, we found that the best approach to representing the original motion was by combining several motion feature extraction methods. The features we extracted included: minimal velocities, movement directional change angle, root-y coordinate (for human-character movement) and constant angular velocities. The generated music notes could basically match the moments when the moving object is making significant movements.

3.2.2 Converting A Motion Event Vector to Motion Music

Given the definition of a motion event and the chosen event extraction methods, we can analyze the motion data and mark most of the moments when special movements take place. Moreover, sometimes certain gestures are more evident than others. Hence we would like to distinguish the events generated at those moments from the others since they are more important. For example, suppose that at time t_1 the moving

object changed direction by 180 degrees, while at time t_2 the object changed direction by only 20 degrees. Then we hypothesize that the "event" that occurred at t_1 might make a stronger visual impact than the event at t_2. To differentiate motion events by their significance, we set up several levels of conditions corresponding to the significance of events, and computed an event's level based on the criteria that the motion satisfies. For example, we define the condition of events of level-4 to be that "the movement directional change angle is greater than $\pi/2$, or the movement directional change angle is greater than $\pi/5$ and the velocity is at its local minimal value".

Therefore, the motion events are divided into several levels according to the significance of the corresponding movements' visual impact. For each motion file, there will be relatively stronger movements and also weaker ones. Hence, each motion recording has different levels of motion events, which later we transferred into different music notes. Motion events with strong visual impact are converted to music notes with strong aural impact. Currently, conditions of all the levels are described in Table 3.2. Note that for the purposes of developing our proposed framework, we defined the various levels heuristically, based on our own observations and trial-and-error. It would be an interesting future research project, outside our current scope, to find more systematic ways of determining the visual significance and impact of such motion properties.

When calculating if an event satisfies a level-6 condition, we apply a moving window with a width of 10 frames, and calculate the difference of each pair of adjacent

Motion event level	Condition	Event value
Level-1	(Movement directional change $> \pi/5$) OR (Local minimal value of root-y coordinate)	1
Level-2	Movement directional change $> \pi/4$	2
Level-3	(Movement directional change $> \pi/3$) OR (Joint is foot) AND (Movement directional change $> \pi/5$) AND (Local minimal velocity)	3
Level-4	(Movement directional change $> \pi/2$) OR (Movement directional change $> \pi/5$) AND (Local minimal velocity)	4
Level-5	(Movement directional change $> 2\pi/3$) OR (Movement directional change $> \pi/5$) AND (Local minimal velocity) AND (velocity $= 0$) OR (Movement directional change $> \pi/4$) AND (Local minimal velocity)	5
Level-6	$(\text{difference}_{angularvelocity}(window) < 10^{-3})$ AND $(length(window) = 10 frames)$	10

Table 3.2: Motion event level definition

frames' angular velocities. The Level-6 condition requires all differences of adjacent angular velocities within the moving window to be less than 10^{-3}. Although some of the conditions are set up for certain types of data (certain joints for human-character movement), they can be applied to all motion capture data.

Consequently, for a single motion signal, we combine all the events values and their timestamps together to build an event vector. An example is shown in Figure 3.12. Therefore, a motion signal is transformed into an event vector, which reflects the visually salient moments in this motion.

Finally, the motion event vectors are converted to music pieces, represented in MIDI format, using the mapping relationships in the above section. A MIDI file is generated note by note with different note attributes using the following conversion:

$$Event(time, value, joint) \Rightarrow MIDInote(onset, duration, pitch, amplitude, timbre)$$

For each music note, its onset is the same time as that of the motion event it corresponds to. The duration, pitch, amplitude and timbre mapping relationships are shown in Table 3.2.2.

We also tried other variations of mapping relationships between motion events and MIDI notes. Those variations of motion music give different acoustic effects, but in our experience, they did not make much difference when the audio beat tracking algorithm was applied.

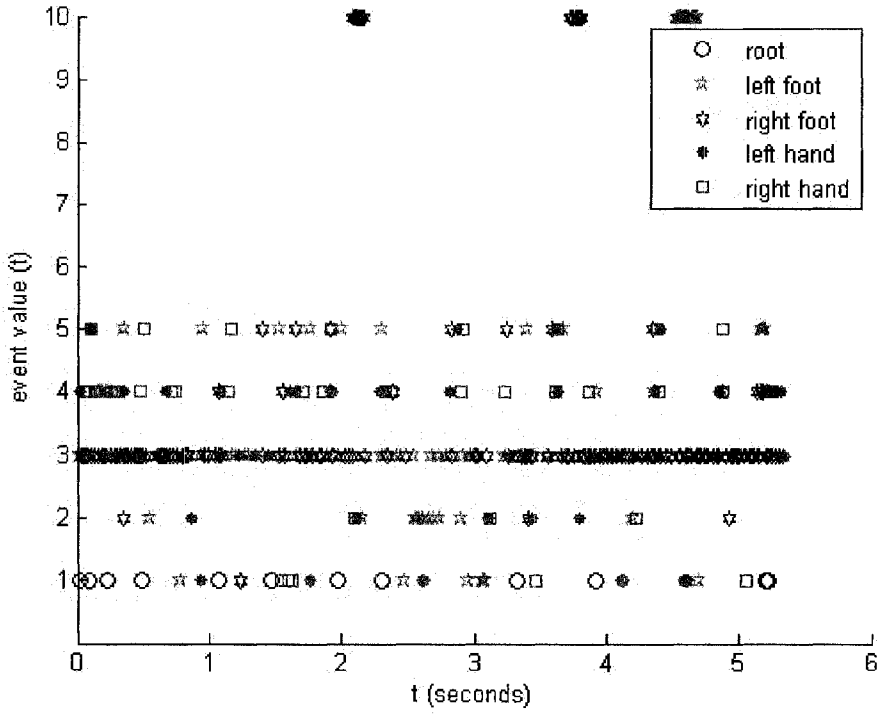

Figure 3.12: Motion event vector computed with a time interval of 0.15s from CMU motion captured data #0513, both hands, both feet and root, x axis refers to the time axis and y axis is the value of events. A black circle refers to an event generated by the root; a green pentagram is an event generated by the left foot; a blue hexagram reflects an event generated by the right foot; a red dot is an event generated by the left hand; a magenta square refers to an event generated by the right hand.

MIDI note		Motion event
Pitch value	36 (C2)	$Event_{root}$ value = 1
	47 (B2)	$Event_{foot}$ value = 1
	48 (C3)	$Event_{foot}$ value = 2
	49 (C3#)	$Event_{foot}$ value = 3
	50 (D3)	$Event_{foot}$ value = 4
	52 (E3)	$Event_{foot}$ value = 5
	54 (F3#)	$Event_{foot}$ value = 10
	57 (A3)	$Event_{hand}$ value = 1
	58 (A3#)	$Event_{hand}$ value = 2
	59 (B3)	$Event_{hand}$ value = 3
	60 (C4)	$Event_{hand}$ value = 4
	62 (D4)	$Event_{hand}$ value = 5
	64 (E4)	$Event_{hand}$ value = 10 if $Event_{hand,previous} \neq 10$
	$Pitch_{previous} - 1$	$Event_{hand}$ value = 10 if $Event_{hand,previous} = 10$
Channel	10 (drum sound)	$Event_{root}$ value = 1
	1 (piano sound)	$Event_{hand}$ Or $Event_{foot}$
Volume	40	$Event$ value = 1
	60	$Event$ value = 2 Or 10
	70	$Event$ value = 3
	90	$Event$ value = 4 Or $Event_{root}$ value =1
	110	$Event$ value = 5
Duration	frame rate^{-1}	$Event_{hand}$ value = 10
	0.5s	$Event$ value = 1
	0.6s	$Event$ value = 2
	0.7s	$Event$ value = 3
	1.0s	$Event$ value = 4
	1.5s	$Event$ value = 5 Or $Event_{foot}$ value =10
	2.0s	$Event_{root}$ value =1

Table 3.3: Motion music note attributes converting relationship with motion events

Chapter 4

Music Query with Motion Recordings

The second part of our system matches a motion signal with corresponding music. Once motion data has been analyzed and converted to a motion music file, the problem becomes one of matching the resulting motion music with a similar music piece from the library. The approach is to treat motion data as music and match music pieces according to rhythmic similarity metrics.

Currently, most of the music rhythmic similarity analysis methods are based on music structure analysis, that is, music beat sequence or regular temporal patterns. In this thesis, we use an existing algorithm to extract the beat sequences from music pieces and then apply similarity comparison algorithms on the extracted beat sequences. A number of music beat tracking algorithms have been developed [18][47], some based on the acoustic information of the sound signal [17][45], and others focused

on music score analysis, such as that available in MIDI representation [6][54].

After comparing some beat tracking algorithms on their efficiency and input / output data format, we finally settled on the beat tracking algorithm developed by Simon Dixon in 2001 [7]. Those algorithms also include Meudic's casual beat tracking algorithm [36] and Eck's autocorrelation phase matrix [12], which we implemented for testing. The beat tracking algorithm we applied is an interactive system called "BeatRoot" provided by Dixon. Incidentally, this algorithm also performed the best on the Audio Beat Tracking task presented by ISMIR 2006 [8]. In order to use this, we convert the music input from our MIDI representation into a .wav format. The structure of the beat tracking algorithm is described in Appendix A.

BeatRoot can detect the beats of the input music file and output the beat times in text-MIDI format, which is a text file containing the list of beat onset times and note information. An example of beats detected by BeatRoot is shown in Figure 4.1. We then read in the text-MIDI files and change them into vectors containing the timestamps of beat information. Therefore, the music rhythmic similarity analysis problem is converted to a vector comparison and matching problem.

In this thesis, we applied three different approaches for matching beat vectors. They are mutual information combined with a window-based matching parameter, two-sample Kolmogorov-Smirnov test, and a rhythmic comparison algorithm that we developed.

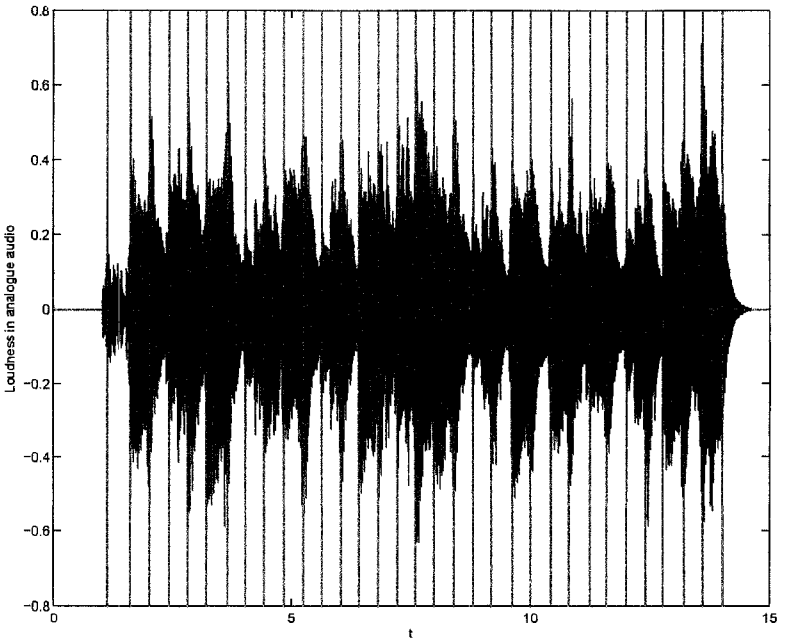

Figure 4.1: Beat sequence detected by BeatRoot and the corresponding analogue audio file "cancan(track #8)". Red vertical lines represent the beats.

4.1 Mutual Information Incorporated with Window-based Matching

The first approach we shall describe evolved over several stages. It began as a mutual-information-based metric, which is described in Section 4.1.1. We then added a gradient term to incorporate temporal constrains with mutual information (Section 4.1.2). Finally we explain a more intuitive interpretation of this approach (Section 4.1.3), which effectively corresponds to a window-based matching together with mutual information.

4.1.1 Mutual Information Based Matching

In information theory, mutual information is a quantity that measures the interdependence of two random variables. It is defined using Shannon entropy, a measure of the uncertainty associated with a random variable. Shannon entropy, or information entropy, quantifies the information contained in a signal, usually in bits (logarithms to the base 2). The information entropy of a discrete random variable X, that can take on possible values $x_1, ..., x_n$ is

$$H(X) = E(I(X)) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i) \tag{4.1}$$

where $I(X)$ is the information content or self-information of X, which is itself a random variable; and $p(x_i)$ is the probability of a given value x_i.

From the definition of Shannon entropy, the mathematical description of mutual

information between two random variables is given by

$$I(X;Y) = I(Y;X) = H(X) + H(Y) - H(X,Y) \tag{4.2}$$

In this equation, $H(X)$ and $H(Y)$ are the Shannon entropy of sample X and sample Y, respectively. $H(X,Y)$ is the joint entropy between sample X and Y, which is defined with the following equation,

$$H(X,Y) = -\sum_{x,y} p(x,y) \log_2(p(x,y)) \tag{4.3}$$

The required entropies $H(X)$, $H(Y)$, $H(X,Y)$, can be computed by estimating the probability distribution of the samples. Since $H(X,Y) \leq H(X) + H(Y)$, this characteristic indicates the non-negativity property of mutual information. Hence, $I(X;Y) \geq 0$.

One of mutual information's important properties is $H(X) = I(X;X)$, that is, mutual information between a variable and itself is the Shannon entropy of the variable. Thus $I(X;X) \geq I(X;Y)$. The information amount carried between two different variables cannot be greater than the amount between one variable and itself. According to this property, the more similar vector Y is to X, the greater their mutual information is.

In this project, we convert the beat vectors to be sequences of 0s and 1s (1 represents the beat time and 0 otherwise) in the resolution of milliseconds. We use the desired motion music beat vector and a beat vector of a music piece from our collec-

tion as variables X and Y respectively, and calculate the mutual information between them. In our case, the more similar a music beat vector is to the motion music beat vector, the greater their mutual information value will be. Hence, we select music pieces by the mutual information between their beat vectors and the desired motion music beat vector. On account of the above properties of mutual information, one can formulate the beat vector matching criterion as selecting the music beat vector Y that maximizes the mutual information $I(X;Y)$ between itself and motion-generated music beat vector X:

$$i_0 = \underset{1 \leq i \leq N}{\operatorname{argmax}} I(X;Y_i) \tag{4.4}$$

in which X and Y_i represent the beat vectors of the motion music to be compared and that of music piece $\#i$ in the library. N refers to total number of music pieces.

4.1.2 Mutual Information Incorporated with Gradient Information

Standard mutual information only considers the distributions' statistical information, which is the number of 0s and 1s in our case. However, the variables' temporal relationship is ignored, which is inappropriate for our situation. For example, if beat vectors A and B have the same number of 0s and 1s, the mutual information of them with another vector $I(C;A)$ and $I(C;B)$ are the same.

To solve this problem, we incorporate temporal constraints by additionally considering gradient information. This idea was inspired by the work of Pluim et al. and

others, who applied this to medical images [43][34].

In order to solve the information limit of standard mutual information when working with medical images, they incorporated gradient information to provide spatial constraints. Their gradient term $G(X, Y)$ between two images X and Y not only seeks to align locations of high gradient magnitude, but also orientation of the gradients at these locations.

In the calculation of gradient, a medical image is considered to be a matrix, while the beat information is given by a vector. As was mentioned previously, the beat vector in our project is a sequence of 0s and 1s (1 represents the beat time and 0 otherwise) in the resolution of milliseconds. For example, if the beats occur at 0, 4, and 6 milliseconds of a 10 milliseconds' music piece, its beat vector will be

$$[1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0]$$

The gradient direction of the beat vector is reduced from within a plane (an image) to the possible directions within one dimension. Hence, considering the similarity of beat vector and image in the process of gradient information calculation, the above method is applied in this project with minor adaptations. In our case, the beat vector is one-dimensional rather than a two-dimensional image, and the gradient (derivative) provides us with temporal rather than spatial constraints. Hence, we calculate the beat vector's first-derivative to measure how beat changes over time.

The gradient for each beat vector is computed by the difference of adjacent values, or set to zero if it is the very last point in the vector. Because the beat vector is one-

dimensional, the gradient direction for each point is considered as $-\pi/2$, 0, or $\pi/2$ if the gradient value is -1, 0 or 1 correspondingly. Then, the angle $\alpha(\sigma)$ between the gradients of the corresponding points on two beat vectors is defined by

$$\alpha(\sigma) = \nabla x(\sigma) - \nabla y(\sigma), \alpha(\sigma) \in [0, \pi] \tag{4.5}$$

where $\nabla x(\sigma)$ and $\nabla y(\sigma)$ refer to the gradient directions in the motion music beat vector and the music beat vector respectively. Because we only need to use angle $\alpha(\sigma)$'s cosine value, we shift it into the range of 0 to π if it is not by adding or subtracting 2π. In practice, $\alpha(\sigma)$ will be 0 if both vectors have the same gradient; π if they have the opposite gradient; or $\pi/2$ otherwise.

A weight function is also set up for each point using the angle,

$$f_{weight}(\alpha) = \frac{\cos(2\alpha) + 1}{2} \tag{4.6}$$

Only gradients that appear in both vectors are counted. This refers to the situation that the two beat vectors both have a beat at the same time ($\alpha(\sigma)$ is 0) or each has a beat at two adjacent time points ($\alpha(\sigma)$ is π). For each point, the angle function is multiplied by the minimum of the gradient magnitudes. The final gradient term is calculated by summing the resulting product for all points,

$$G(X, Y) = \sum_{(x,y)\in(X\cap Y)} f_{weight}(\alpha_{x,y(\sigma)}) min(|\nabla x(\sigma)|, |\nabla y(\sigma)|) \tag{4.7}$$

When there is non-zero gradient values in both X and Y vectors, the gradient term will count that in. Hence, in our case, the gradient term can be considered as the score of counting the non-zero gradient magnitudes that appear in both X and Y vectors at nearly the same time.

The final comparison score is composed of two parts that complement each other: mutual information term (I), which compares the two beat vectors' statistical information, and the gradient term (G), which considers how the beat vectors change. Finally, we combine the two terms together by multiplying the gradient term to the normalized mutual information (NI), which are defined by the following equations,

$$I_{new}(X,Y) = G(X,Y)NI(X,Y) \tag{4.8}$$

and

$$NI(X,Y) = \frac{I(X)+I(Y)}{I(X,Y)}. \tag{4.9}$$

4.1.3 An Alternative Interpretation: Matching Beats within Small Windows

The gradient term is typically used in image processing with a range of possible grayscale values, whereas in our application, we have only two binary values: 0 (no beat detected) and 1 (beat detected). Thus, the "gradient" is in fact being used in a limited situation, and has a simpler interpretation in this case. In particular, consider as a simple example, the beat vector $[0,0,0,0,1,0,0,0,0]$. The gradient for this vector

is $[0, 0, 0, 1, -1, 0, 0, 0]$. Hence, computing the gradient term effectively just expanded the region of non-zero elements in the vector, allowing a slightly larger time window for matching beats. Thus, we can reinterpret the above gradient-based approach as a window-based comparison method. This approach obtains the beats' temporal information by counting how many beats of the motion music are matched by the candidate music's beats. This temporal information is represented by a parameter T. It is accomplished by simply setting up an analysis window (3 frames' width) around each beat of the motion beat vector, and determine if there is a music beat located in that window. The window-based beat matching approach can be described by a weight function on each beat of the motion beat vector X, as the following,

$$f_w(beat_X(k), Y) = \begin{cases} 0, & \text{if no music beat in } beat_X(k)\text{'s analysis window} \\ 1, & \text{otherwise.} \end{cases} \tag{4.10}$$

$beat_X(k)$ refers to the motion beat vector's kth beat, and $beat_X(k)$'s window is the 3-frame width's analysis window centered at $beat_X(k)$. Y is the beat vector of a candidate music piece.

Hence, the temporal parameter T for motion beat vector X (with a total of N beats) and music beat vector Y can be defined as

$$T(X, Y) = \sum_{k=1}^{N} f_w(beat_X(k), Y) \tag{4.11}$$

Finally, we implement this mutual information combined with a window-based

temporal parameter matching algorithm by multiplying the temporal term (T) to the normalized mutual information (NI). The comparison process between a motion music beat vector X and a candidate music's beat vector Y is defined by the following equations,

$$I_{new}(X,Y) = T(X,Y)NI(X,Y) \tag{4.12}$$

and

$$NI(X,Y) = \frac{I(X)+I(Y)}{I(X,Y)} \tag{4.13}$$

Because the music pieces vary in length, the number of points in their corresponding beat vectors also vary. To fully make use of the length of the beat vectors, we truncate the music beat vector if it is longer than the motion music beat vector, or duplicate it until its length is not less than that of the motion beat vector and then truncate the extra part to match with the motion beat vector otherwise.

A music piece may not start exactly on what is referred to as the "downbeat" (the first beat of a bar), and furthermore, the recorded motion may begin with some amount of "waiting". These, and other factors, may lead to different offsets between the starting points of music and motion data. Considering that, we set up a window w_2 and move the music piece within that window when comparing it to a motion beat vector until we find the best possible match between the given pair of musical pieces. The Figure 4.2 indicates the position of w_2 and how the music piece is moved within it.

Therefore, the new mutual information combined with window-based temporal

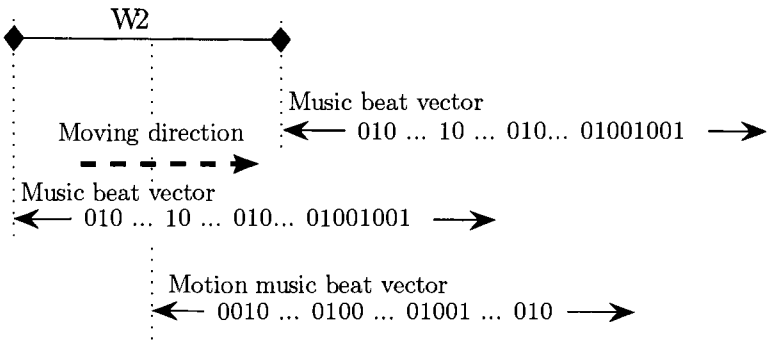

Figure 4.2: Moving window of w_2 and the motion and music beat vectors' comparison

parameter matching result should be revised to be the maximum value of music beat vector within the window w_2, which is shown by

$$MIT(X, Y_{(w_2)}) = \max(I_{new}) = \max(T(X, Y_{(w_2)})NI(X, Y_{(w_2)})) \tag{4.14}$$

Hence, the revised music and motion music beat vector matching criterion is to find out the music piece Y_{i_0}, by the following equation:

$$i_0 = \underset{1 \le i \le N}{\operatorname{argmax}} MIT(X; Y_i) \tag{4.15}$$

in which X and Y_i represent the beat vectors of the motion music to be compared and that of music piece $\#i$ in the library. N refers to total number of music pieces.

4.2 KS Test

The Kolmogorov-Smirnov test (K-S test) is a goodness of fit test used to determine whether two underlying one-dimensional probability distributions differ significantly. Because our data to be tested are beat vectors, which are random samples whose numerical interpretation are unclear, we apply the KS test on our beat sequences. The advantage of the KS-test is that the distribution of its test is non-parametric and distribution free, which means the test statistic itself does not depend on the underlying cumulative distribution function being tested. Another attractive feature is that it is a statistical technique that performs well under a wide range of distributional

assumptions (robust).

The two-sample KS test is one of the most practical and commonly used methods to determine the statistical difference between samples.

KS test uses the maximum vertical distance between the cumulative distribution function curves of the two samples, X_1 and X_2 with length n_1 and n_2 respectively. The test statistic can be written as

$$D_{n_1,n_2} = \max\left(|F_{n_1}(x) - F_{n_2}(x)|\right) \tag{4.16}$$

where $F_n(x)$ is the empirical cumulative distribution function ($ECDF$) of the distribution n being tested. Given dataset X with n ordered data points $X_1, X_2 \ldots X_N$, the $ECDF$ is defined as

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} I_{X_i \leq x} \tag{4.17}$$

$I_{X_i \leq x}$ is the indicator function. It uses 1 and 0 to determine the membership of an element in a set $X_i \leq x$ (1 for yes and 0 otherwise). That is, for each x value, $\sum_{i=1}^{n} I_{X_i \leq x}$ counts the number of points in dataset X that are less than x.

In practice we use an existing function in Matlab to perform a two-sample Kolmogorov Smirnov test to compare the distributions of values in the two vectors X_1 and X_2 of length n_1 and n_2. In our case, X_1 is the beat vector derived from the motion-generated music, and X_2 is the beat vector derived from the candidate piece of music currently being tested as a possible soundtrack. Note that unlike the beat vectors of 0's and 1's that we have used in previous sections, we now represent the

beat vectors as a series of beat onset times. As a simple example, if the time between samples were 1 millisecond, then the vector

$$[1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0]$$

would now be represented as

$$[0, 4, 6]$$

since the beat onset times occur at 0, 4 and 6 milliseconds.

The null hypothesis for this test is that X_1 and X_2 are drawn from the same continuous distribution. The result H is 1 if the null hypothesis can be rejected or 0 otherwise. The Matlab function also returns the asymptotic p-value to report if the numbers differ significantly. A smaller p-value means we can reject the null hypothesis with greater certainty. In our case, we reject the null hypothesis for p values less than 5%. The asymptotic p-value is believed to be reasonably accurate for sample sizes n_1 and n_2 such that $\frac{n_1 n_2}{n_1+n_2} \geq 4$.

Because the beat sequences of our music files are large (a beat vector usually contains some dozens of points, i.e. a beat vector could contain from 20 to hundreds of points), the sample size condition for accurate p-value is always satisfied. In our comparison process, we first sort all music pieces by whether the null hypothesis (i.e. that the two beat distributions are the same) can be rejected. Having found those music pieces for which the null hypothesis cannot be rejected, we then sort them by p-values and retrieve the one which is closest to the motion music beat vector as the

result.

4.3 Rhythmic Comparison

Rhythm is considered one of the fundamental elements of music[20]. It contains the duration and accentuation information of music (beat sequence and meter). For example, meter information is very important in rhythm, which is the description of how the beats are stressed and grouped. Based on our assumption, the matched music piece to a motion signal should not only have similar beats, but also similar rhythm. However, the above two approaches that we developed only compare the two beat sequences of music and motion-generated music. Consequently, if different music files have similar beats but vary in their meter, the retrieved music piece for a motion signal might be a mismatched result. To achieve a more accurate matching, we would like to incorporate the music rhythm information into the comparison process. Hence, we designed a rhythmic comparison algorithm to search for rhythmically similar results. Not only are the beat vectors of the motion music and music pieces used, original music wave forms and motion event vectors are also employed.

Comparing the rhythms of two pieces of music is a potentially very complex (and subjective) question. Nevertheless, in matching a motion-based music signal, M_{motion}, with a candidate piece of music, M_{music}, we wished to explore more than just comparing their respective beat vectors. In this section, we develop a heuristic approach to comparing the "rhythmic activity" in M_{motion} that coincides with the beats of M_{music}. We do this in two steps:

Step 1: We compute a score S indicating the total rhythmic activity in M_{motion} occurring near the beats of M_{music}.

Step 2: We re-assess S, this time accounting for the *meter* of the piece, i.e. 5/4, 3/4, etc, by giving different weight to the different beats.

4.3.1 Step 1: Scoring the Rhythmic Activity

One of the basic aspects of rhythm is tempo, which corresponds to the duration between beats. That is, a fast tempo means a short duration between beats. Accordingly, we assume that when a motion signal M_{motion} matches a music piece M_{music} rhythmically, there should be strong rhythmic elements of the motion music occurring around many of the beats of the music piece[1]. For our motion-generated music, the events generate music notes (described in Section 3.2), which constitute the basic rhythmic elements of the piece. We have defined our events such that higher event levels can roughly correspond to louder note volumes, and multiple events happening at the same time can also increase the overall volume. To compute the score of how well the M_{motion} is matched with M_{music} on the beats of the candidate music piece, we perform the following:

1. First, we set a small time window w_3 around every beat in M. This gives us a series of N windows, where N is the number of beats in the piece.

2. Within each window, find the note in M_{motion} with the strongest volume. This

[1]Note that this assumption need not always be true, but it is simply a heuristic starting point for demonstrating the potential application of the framework we are developing in this thesis.

gives us a set L, consisting of N "loudest notes".

3. Find the average volume of L.

If the music piece matches the motion signal on the beat times, the average volume value is higher than the pieces that do not match.

The loudness of motion music note is proportional with the level of the motion event from which the note is converted. Hence, we use motion event values in this step to represent loudness of motion music. Motion event vectors are applied in the calculation instead of motion music. Thus the problem of comparing average motion music volume becomes the comparison of average event values. Consequently, the tempo comparison process of music number j regarding motion recording i can be defined as,

$$score_i(j) = \frac{1}{N}\sum_{k=1}^{N} \max(event_i(beat_j(k) - \frac{w_3}{2}, beat_j(k) + \frac{w_3}{2})) \tag{4.18}$$

where N is the number of beats for music piece number j, and $beat_j(k)$ means the kth beat of music j. The example of setting window w_3 is shown in Figure 4.3. The different shapes and colored dots, the same as that in Figure 3.12, denote the motion events of different joints respectively. The black vertical lines refer to the detected beats and the dashed lines denote the range of analysis window w_3 around the beats.

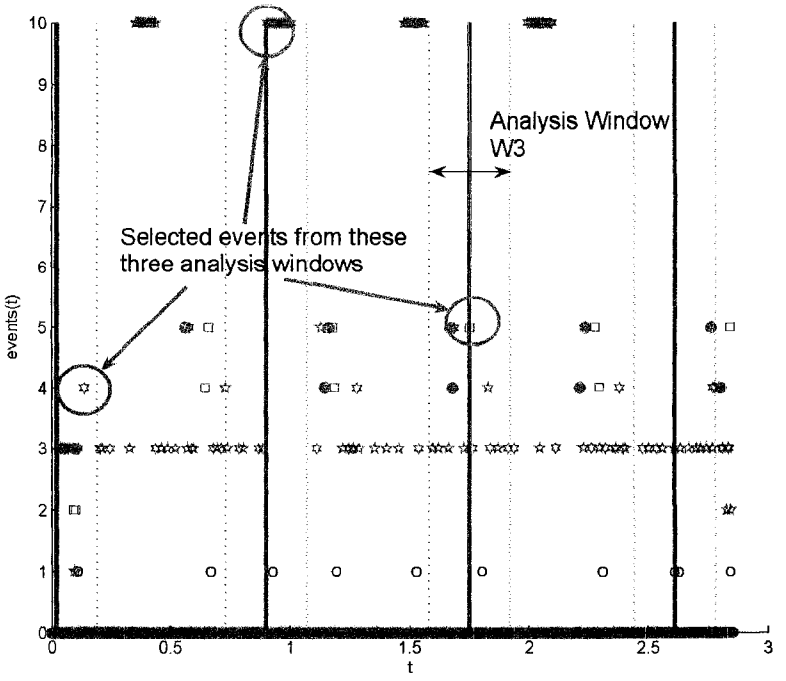

Figure 4.3: Moving analysis window w_3

4.3.2 Step 2: Weighting According to the Meter

Music meter measures how the musical lines are divided into bars of stressed and unstressed beats. We would like to detect the rhythmically stressed beats on the beat sequences extracted by BeatRoot. Normally, such beats can be reflected by the loudness of analogue audio format. Music notes with accentuation have higher amplitude than ordinary notes. Hence, in the rhythmic comparison algorithm, music wave files are loaded for meter comparison, as opposed to the processed beat vectors.

Figure 4.4 shows comparison between a simple two-hand movement motion event vector and a music piece (in analogue audio format). In this figure the black vertical lines denote the music's beats detected by BeatRoot. The motion events are presented by green and blue dots, and the red wave along t axis is the analogue audio.

For music files that are stereo (have two sound channels), we first force them to be mono by calculating the average loudness of analogue audio of the two channels. Usually, the waveforms are recorded at sampling rate of 44100Hz or 22050Hz. However, in the rhythmic comparison process only the music samples that were taken at the same time with the beats are involved. Accordingly, we down-sample the waveform with its beats to largely reduce its data size. Thus, only the loudness values at the beats are retained.

For most music works, the numbers of beats in a measure range from 2 to 7 and the most common note values are half, quarter and eighth notes, corresponding to different degrees of subdivision of each measure. To find out the meter for each music piece, the algorithm compares the average loudness of the first beat in each

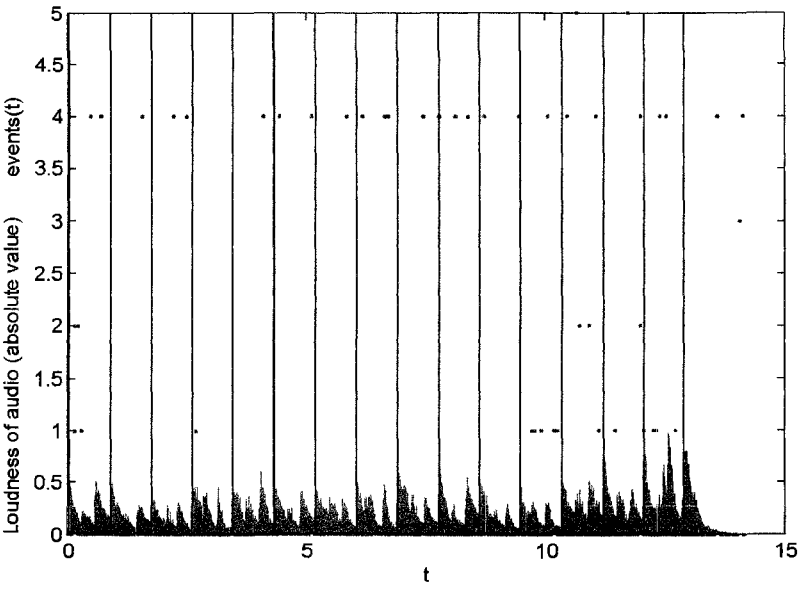

Figure 4.4: Wave form of music piece "waltz (track #42)" with motion events of #233234

measure under assumptions of 2 to 7 beats per measure respectively. The number that generates the highest average loudness value is considered to be a likely rhythm. The meter M_j for music piece number j can be calculated by,

$$M_j = \underset{2 \leq m \leq 7}{\operatorname{argmax}}\left(\frac{1}{N_m} \sum_{i=0}^{N_m - 1} loudness(beat_j(mi + 1))\right) \tag{4.19}$$

in which, m denotes the possible meter value which ranges from 2 to 7, and N_m is the total number of measures in the music piece according to m value.

Having a music piece's meter information and beat sequence, we could differentiate the beats of that music into two types, rhythmic beats and ordinary beats. A

rhythmic beat refers to the first beat in each bar, typically with some accentuation[2]. Considering the importance of rhythmic beats, we combine the meter parameter with the tempo comparison algorithm of the previous step to form a rhythm comparison algorithm. It is achieved by adding rhythmic parameter to the motion music notes that are located in the windows of rhythmic beats. This can be done by weighting the analysis windows according to the type of music beat corresponding to each window. Consequently, we use the following weight function wt,

$$wt_{beat(k)} = \begin{cases} 2, & \text{if } beat(k) \text{ is rhythmic beat,} \\ 1, & \text{otherwise.} \end{cases} \tag{4.20}$$

Then, by simply multiplying the values of events with the weights of the corresponding beats, and then dividing by the weighted sum of the number of beats, we have the rhythmic comparison score. Finally, the new rhythmic comparison criteria for motion signal i and music piece j can be presented as,

$$r_i(j) = \frac{\sum_{k=1}^{N} wt_{beat_j(k)} \max(event_i(beat_j(k) - \frac{w_3}{2}, beat_j(k) + \frac{w_3}{2}))}{\sum_{k=1}^{N} wt_{beat_j(k)}} \tag{4.21}$$

where N is the total number of beats for music piece j.

[2]Again, there are numerous exceptions to this, but developing more sophisticated musical models is outside the scope of this current project.

Chapter 5

Results and Discussion

5.1 Implementation Overview

The model described above is implemented under Matlab7.0 with programming language C on an Intel Pentium PC (P4 3.2GHz processor and 1GB memory). See figures 5.1, 5.2, and 5.3 for further details.

5.2 Testing Data

To test our system, we created a dataset consisting of both recorded music and recorded motion. Each motion was recorded in synchrony with one of the music pieces, as described below.

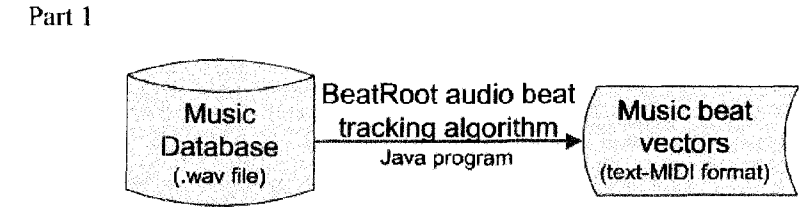

Figure 5.1: System implementation part 1, music data analysis

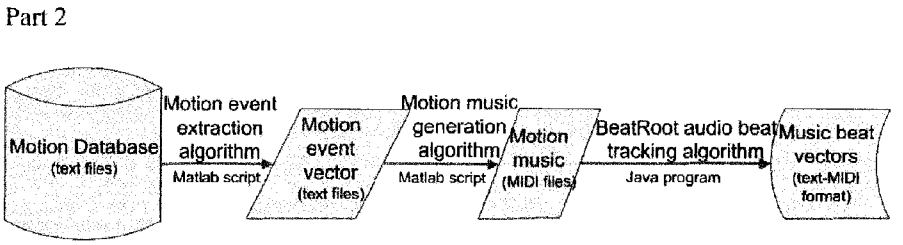

Figure 5.2: System implementation part 2, motion data analysis

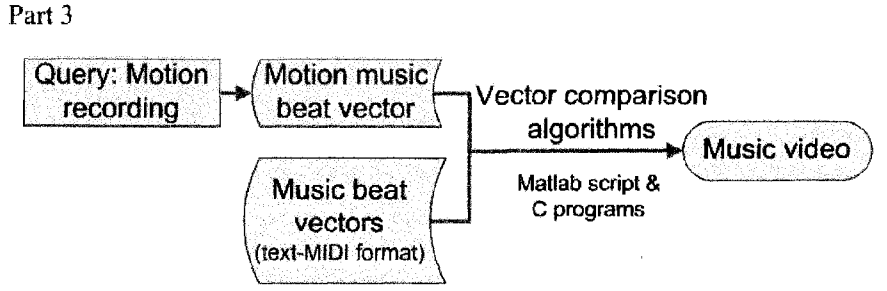

Figure 5.3: System implementation part 3, music retrieval using motion queries

Music Data

Each music piece was recorded as a set of individual tracks, such as a bass line, a rhythmic harmonic accompaniment or melody. Furthermore, alternate takes were recorded. So, for example, the piece called "Minor Swing" may have had 2 different bass lines, 2 counter-melodies, etc. We recorded a total of 11 basic musical pieces (music database in Figure 5.1), and the tracks of each piece were then recombined to create a set of variations (e.g. bass1 and melody2, bass2 with rhythmic accompaniment with melody1, etc). Some of the recorded music pieces' tempos are full of rubato and accelerando, which mean the slightly slowing down or speeding up of the tempo. Such example can be found in Abstract1, Abstract2, Beethoven or Titina. Some pieces are unclear in meter. For example, the meter of music piece Abstract2 could be 3/4, or slow 4/4, or 2/4; that of Titina can be 4/4 or 2/4; the meter of Montuno can be seen as 4/4 but not in any clear way. It comes from the style of Latin music which use a different rhythmic pattern than the "1, 2, 3, 4" of western music. Table 5.1 shows all the music tracks and their description. For those pieces that have unclear meters, only the most likely meter value or main meter is shown on the table.

#	Music ID	Meter	Description
1	Abstract1	4/4	Full track
2	Abstract2	3/4	Full track
3	Beethoven	2/4	Full track
4	CanCan	2/4	Base track
5		2/4	Bass + Accompaniment
6		2/4	Bass + Accompaniment + Grace notes
7		2/4	Bass + Accompaniment + Melody + Grace notes4
8		2/4	Bass + Accompaniment + Melody + Grace notes5
9	March	4/4	Bass + Harmony
10		4/4	Melody
11		4/4	Counter-Melody
12	Minor Swing(Straight)	4/4	Base track1
13		4/4	Bass1 + Accompaniment2
14		4/4	Bass1 + Accompaniment5
15		4/4	Accompaniment2 + Bass4
16		4/4	Bass4 + Accompaniment5
17		4/4	Bass1 + Accompaniment2 +Solo
18		4/4	Accompaniment2 + Solo +Bass4
19		4/4	Full track
20	Minor Swing(Swing)	4/4	Bass track1
21		4/4	Bass1 + Accompaniment2
22		4/4	Bass1 + Accompaniment5
23		4/4	Accompaniment2 + Bass4
24		4/4	Bass4 + Accompaniment5
25		4/4	Bass1 + Accompaniment2 +Solo
26		4/4	Accompaniment2 + Solo +Bass4
27		4/4	Full track
28	Montuno	4/4	Track1 + Montuno
29		4/4	Track1 + Montuno + Bass
30		4/4	Track1 + Montuno + Bass + Melody

Continued on Next Page...

#	Music ID	Meter	Description
31	Seven	7/4	Bass2
32		7/4	Accompaniment1 + Bass2
33		7/4	Bass2 + Accompaniment3
34		7/4	Accompaniment1 + Bass2+ Accompaniment3 + Melody4 + Melody5
35	Take Five	5/4	Bass track1
36		5/4	Bass track3
37		5/4	Bass3 + Accompaniment
38		5/4	Melody + Bass3 + Accompaniment
39	Titina	2/4	Full track
40	Waltz	3/4	Bass track
41		3/4	Bass + Accompaniment
42		3/4	Bass + Accompaniment + Melody
43		3/4	Melody + Bass accompaniment

Table 5.1: The list of music tracks used to test the model, which contains 11 different pieces and a total of 43 distinct music tracks.

Motion Data

Our recorded motion files (motion database in Figure 5.2) are the two-hand movements of a dancer, who created the motions intuitively according to the background music. We recorded the 3D positions of the two Polhemus sensors, which were attached to the two hands of the dancer. The position of a sensor is measured relative to the position of a fixed origin. Each motion file is recorded in synchrony with a music piece. Table 5.2 shows the motion files and the corresponding music tracks to which they were recorded. In this table, the "Music Track #" under "Retrieved Music" column is taken from the corresponding # from Table 5.1.

Motion ID	Background Music	
	Music ID	Music Track #
232432 23267	Abstract1	1
232234	Abstract2	2
231821	Beethoven	3
23345 205138 20524	CanCan	8
231243 231351 231456 23175	March	9 10 11
233147	Minor Swing (Straight Version)	18
23288 232941	Minor Swing (Swing Version)	26
233522	Montuno	30
23742 23943 231121	Seven	31 33 34
23371 233826 233958 23538	Take Five	35 36 38
232027	Titina	39
233234 23336	Waltz	42

Table 5.2: The list of two-hand movements motion files and the corresponding music tracks that they used as background music.

5.3 Testing Results

In this section, we first describe the results of motion music generation, and then that of music retrieval using motion queries. With these predefined motion–music relationships, we could compare the results of our system with the properly matched results and then evaluate the effectiveness of our system. However, the dance motions were created based on human perception of the background music. Variations of the same music piece or even different music may generate motion recordings with similar underlying rhythm. Therefore, there is no absolute answer of right or wrong. A motion recording might have been recorded with one music track, but might indeed have a better match with another variation of that same music piece at the same tempo. Hence, a reasonable goal of the result testing is that the system could retrieve at least one of the variations of the music with which the motion file was recorded, or a music piece with similar rhythm, rather than the correct music and correct variation track. Generally, we can identify music pieces with similar rhythm by listening. Technically, those music pieces should have similar tempo or beat sequences to the correct track. These results should be interpreted with some caution, as described in Section 5.4.

5.3.1 Motion Music Generation Results

During the motion event extraction process, we specified the following parameters: time interval $\triangle T$ (see Section 3.1.2), number of consecutive frames k (Section 3.1.6), and several thresholds (Section 3.1) to extract events from the motion signal. Cur-

rently, the setting of window size or threshold value is based on empirical trials to optimize the performance. However, tuning the threshold setting and window size may result in variation of generated motion event number and occurrence time.

Figure 5.4 and 5.5 show two motion event vectors generated from the same motion recording but with different settings of the parameter $\triangle T$ when calculating the movement directional change angle, and different values of k when detecting segments during which angular velocity remains constant. This example shows that the system generates different motion event vectors under different settings; consequently, the resulting motion music will have different notes.

These different music notes mean that the detected motion music beat sequences will be different as well. The difference could be slight in some cases but more significant in some other situations.

Figure 5.6 shows an example of different motion music beat sequences of the same motion file (CMU motion capture data #0201). One beat sequence is displayed by red dots and the other one by blue "X"s. The two motion music files used to track these beat sequences are generated based on motion event vectors of different parameters settings shown in Figure 5.4 and 5.5 respectively.

Figure 5.7 shows another motion file's (CMU motion capture data #0912) two beat sequences extracted from its two different motion-generated music pieces. The parameters settings which are used to calculate its motion event vectors are the same as the previous example respectively. As we can see from the example of motion data #0201 (Figure 5.6), the dot sequence basically matches the "X" sequence, which

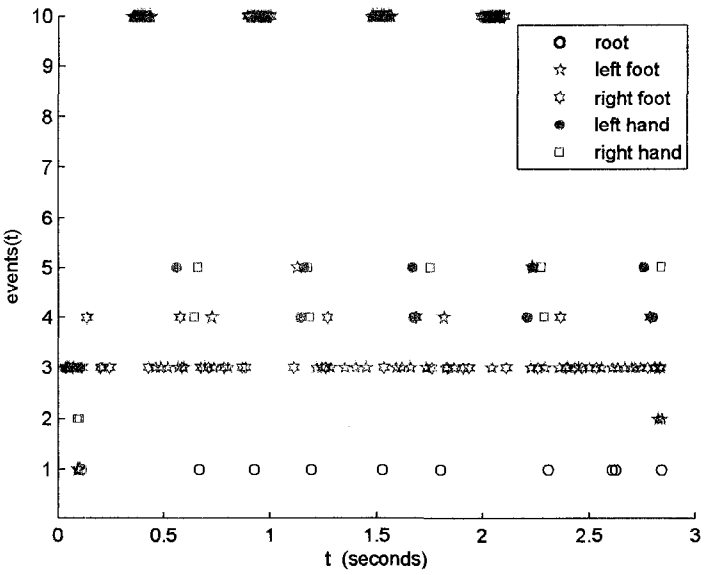

Figure 5.4: Motion event vector of CMU motion capture data 0201, with the ΔT set to 0.15s, k set to 10 frames. Event levels are defined according to Table 3.2

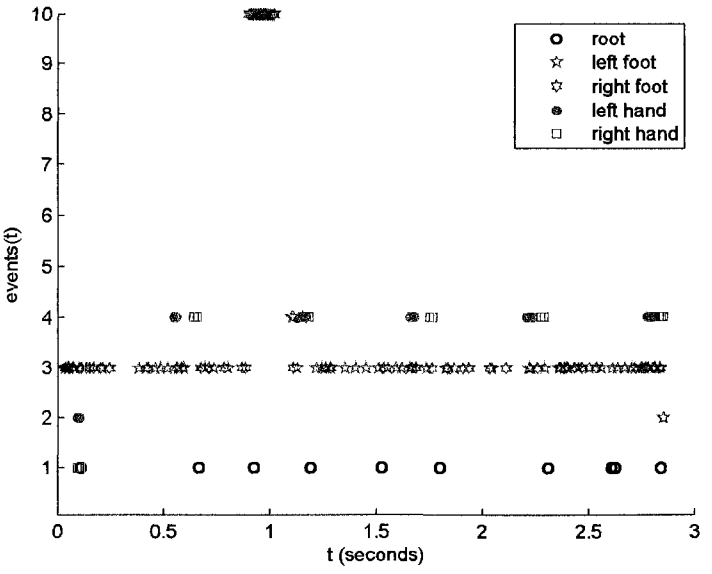

Figure 5.5: Motion event vector of CMU motion capture data 0201, with the ΔT set to 0.1s, k set to 12 frames. Event levels are defined according to Table 3.2

means the beat sequence of one motion music is basically the same with the other one. However, in the second example(Figure 5.7), the two beat sequences barely match each other, and the "X" sequence has a higher density. Hence the two versions of motion data #0912's motion music have different beats. Moreover, one version of the motion music has faster beats than the other one.

The above examples display the situation that with different parameter settings, the extracted underlying rhythm would be different for the same motion file, and the significance of difference varies according to different motion data. Based on our experiment, motion signals with obvious repetitive movement patterns will have more robust beat sequences than free style motion recordings. That is to say, motion files with cyclic signals are more likely to generate beat sequences with slight difference under different parameter settings. On the other hand, motion files with unclear movement patterns are more likely to have beat sequences with significant differences detected using different parameters. Although there is the fact that the music beat tracking algorithm won't be able to detect the exact beat sequence for each music piece, these examples reveal the uncertain factor of the system. Further discussion is in Section 5.4. In practical experiments, motion events are extracted under the same parameters settings.

The system is capable of handling a wide range of motions; our analysis included human-character motions, human-character two-hand movement motions and single object motions. Figure 5.8, 5.9, 5.10 and 5.11 show the motion tracks in 3D space with detected motion music beats of the motion-generated music plotted at the correspond-

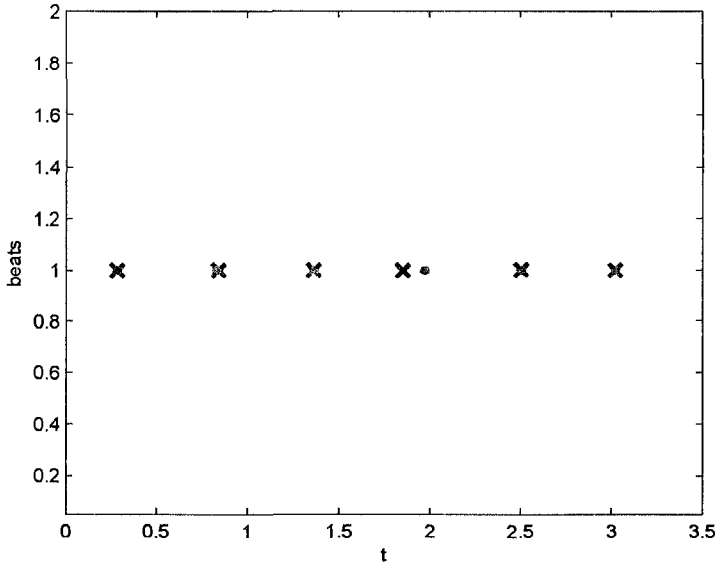

Figure 5.6: Detected motion music beats of CMU motion capture data #0201. Blue crosses denote the beats of motion music generated from motion event vector with $\triangle T = 0.15s$ and $k = 10frames$, and red dots refer that generated with $\triangle T = 0.10s$ and $k = 12frames$

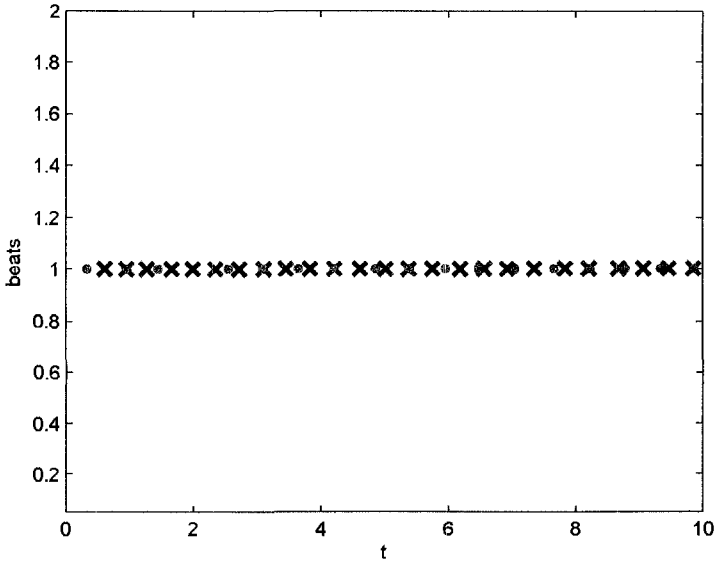

Figure 5.7: Detected motion music beats of CMU motion capture data #0912. Blue crosses denote the beats of motion music generated from motion event vector with $\triangle T = 0.15s$ and $k = 10frames$, and red dots refer that generated with $\triangle T = 0.10s$ and $k = 12frames$

ing places. Blue stars in those figures refer to the motion music beats. Figures 5.8 and 5.9 draw the root tracks of the human-character with beats, Figure 5.10 and 5.11 plot one of the hand tracks with beats.

Examining those figures, the moving object's motion track may not reflect the motion beats in an obvious way, i.e. the beats do not necessarily occur at either peaks or bottoms of the motion curves. There are two reasons for this,

1. In a picture, only a certain joint's motion track is plotted, which may not fully represent the motion features of other joints. While the beats of the motion data are extracted from the movement of all joints, they do not necessarily appear at peaks or bottoms of every joint's track.

2. In our system, motion events, that is, the corresponding music notes are detected using several motion characteristic extraction methods (described in Section 3.1). There are other factors determining the occurrence of motion events— and hence motion beats— than the position of one joint. For example, directional change, minimal velocity and acceleration all may contribute to motion events. The shapes of acceleration or moving directional change curve are not the same as the joint's position curve. Therefore, the motion curve of a single joint cannot fully highlight all of the motion data's features, as well as the occurrence of motion beats.

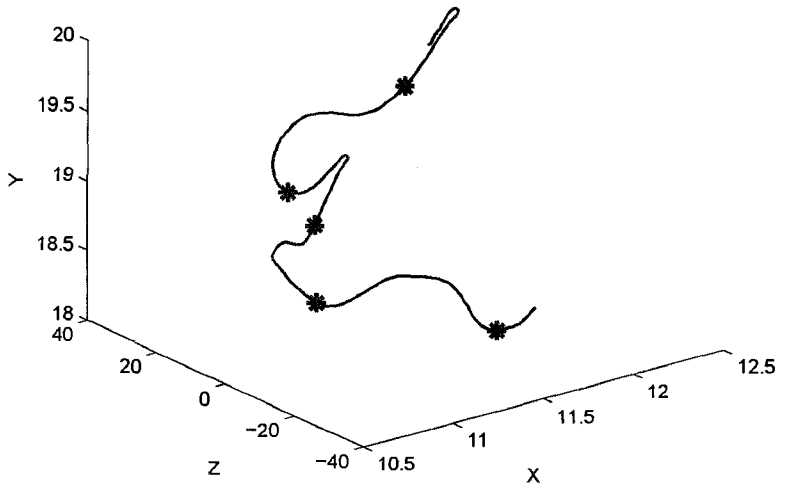

Figure 5.8: Motion track of CMU motion capture data #0201's root node with detected motion music beats

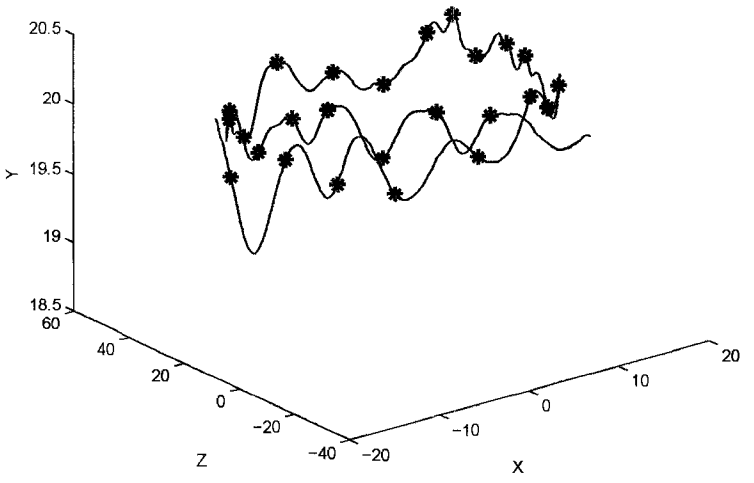

Figure 5.9: Motion track of CMU motion capture data #0912's root node with detected motion music beats

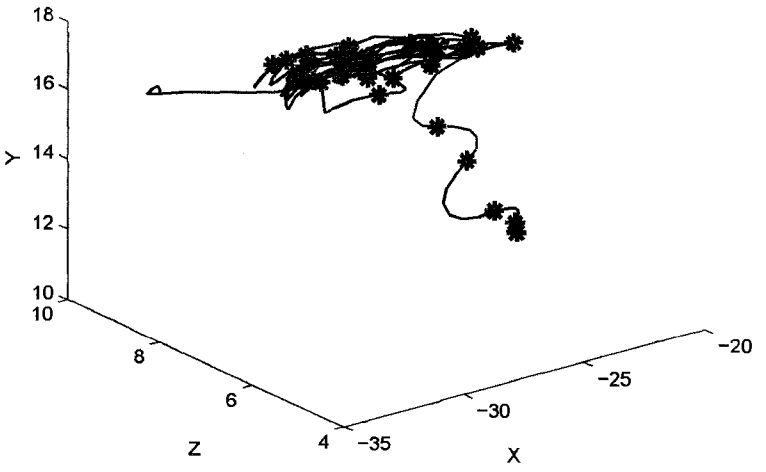

Figure 5.10: Human-character two-hand movement motion data #20524's left hand motion track with detected motion music beats

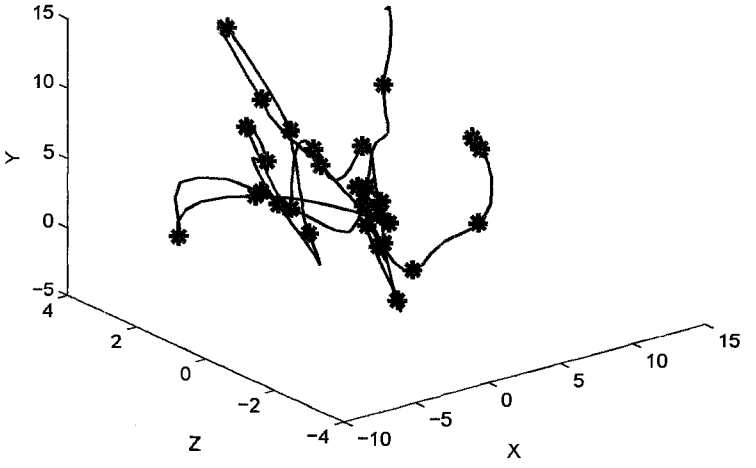

Figure 5.11: Human-character two-hand movement motion data #20524's right hand motion track with detected motion music beats

5.3.2 Music Retrieval Results

After applying the BeatRoot beat tracking algorithm, all the motion music and candidate music pieces are represented by corresponding beat sequences. We tested our model with the three music retrieval methods (described in Section 4) and the results are shown below. Table 5.3 displays the result of using mutual information combined with a window-based temporal parameter, Table 5.4 shows the music retrieval results of applying the K-S test, and the results of the rhythmic comparison algorithm are listed in Table 5.9. In these tables, the Motion # is the same with that in Table 5.2, and the retrieved "Music Track #" is the corresponding music piece's # in Table 5.1.

Comparing the retrieval results of Table 5.3 with the motion recordings' background music in Table 5.1, we note there are seven correct matches out of 25 motion data. By correct match we mean that the retrieved music file is any one of the variations of the music piece, to which the motion was recorded. The detailed result statistic is listed in Table 5.5. On the list of the query motion's possible matches, although the correct music pieces are ranked high in most of the time, this method is not likely to find out the exact matches. By exact match we mean the method not only retrieves the correct music, but also the correct variation track of that music piece.

The same analysis has been done on the music retrieval results using K-S test. The statistics are listed in Table 5.6. Compared with the results of mutual information with a window-based temporal parameter, the K-S test not only provides a higher correct match number, but also the exact match rate is improved. This is because

Query Motion ID	Retrieved Music	
	Music ID	Music Track #
232432	Montuno	30
23267	Minor Swing (Straight Version)	18
232234	Minor Swing (Straight Version)	18
231821	Beethoven Minor Swing (Straight Version)	3 18
23345	March	11
205138	CanCan	8
20524	CanCan	8
231243	March	11
231351	Minor Swing (Swing Version)	26
231456	Take Five	38
23175	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
233147	Take Five	38
23288	CanCan	8
232941	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
233522	Waltz	42
23742	Seven	34
23943	Seven	34
231121	Montuno	30
23371	Seven	33 34
233826	Titina	39
233958	Minor Swing (Straight Version)	18
23538	March	11
232027	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
233234	March	11
23336	Minor Swing (Swing Version)	26

Table 5.3: Test results using mutual information incorporated with a window-based temporal parameter

Query Motion ID	Retrieved Music	
	Music ID	Music Track #
232432	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
23267	Take Five	36
232234	Titina	39
231821	Beethoven	3
23345	Waltz	42
205138	Waltz	42
20524	Waltz	42
231243	March	9 10 11
231351	March	10
231456	March	9 11
23175	March	10
233147	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
23288	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
232941	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
233522	Montuno	30
23742	Take Five	38
23943	Seven	31 34
231121	Seven	31 33
23371	Take Five	35 36 38
233826	Take Five	38
233958	Take Five	38
23538	Take Five	36
232027	Titina	39
233234	Waltz	42
23336	Waltz	42

Table 5.4: Test results using K-S test

	Number of Matches
Incorrect Match	18
Correct Match	7
Exact Match	4
Total Number of Motion File	25

Table 5.5: Music retrieval results of using mutual information incorporated with a window-based temporal parameter

	Number of Matches
Incorrect Match	7
Correct Match	18
Exact Match	11
Total Number of Motion File	25

Table 5.6: Music retrieval results of using K-S test

a large portion of the retrieved results contain multiple music tracks. In K-S test music retrieval process, several music tracks can be selected if they all have the same (highest) probability that the hypothesis is true.

To improve the result of music retrieval, we considered combining mutual information with a window-based temporal parameter and K-S test together, so that the two methods could complement each other. Therefore, we use K-S test in the first stage on all the candidate music tracks to select the ones that have the probability to match the motion, and then apply the mutual information with a window-based temporal parameter method. Hence we could further compare the probabilities of the selected music tracks matching the motion signal rhythmically and narrow down the final retrieval range (having single music piece as retrieval result). This method has been implemented and the results are shown in Table 5.7. The corresponding result analysis is displayed in Table 5.8.

Query Motion ID	Retrieved Music	
	Music ID	Music Track #
232432	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
23267	Minor Swing (Straight Version)	18
232234	Minor Swing (Straight Version)	18
231821	Beethoven	3
23345	Waltz	42
205138	CanCan	8
20524	CanCan	8
231243	March	11
231351	March	10
231456	March	10
23175	March	9
233147	Minor Swing (Straight Version)	18
23288	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
232941	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
233522	Montuno	30
23742	Seven	34
23943	Seven	34
231121	Seven	33
23371	Seven	34
233826	Take Five	35
233958	Take Five	35
23538	Take Five	35
232027	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
233234	Waltz	42
23336	Waltz	42

Table 5.7: Test results using mutual information with a window-based temporal parameter combined with K-S test

	Number of Matches
Incorrect Match	6
Correct Match	19
Exact Match	11
Total Number of Motion File	25

Table 5.8: Music retrieval results of mutual information with a window-based temporal parameter combined with K-S test

From the numbers in Table 5.8 we can see that, applying the music retrieval approach of combining mutual information with a window-based temporal parameter with K-S test on our testing data, although the number of exact matches doesn't increase, the number of correct matches increases and the situation of multiple retrieved music tracks in Table 5.4 has been reduced. As explained in Section 5.3, the method of combining mutual information and a window-based temporal parameter with KS test performs better than each of them individually.

Similarly to the previously displayed three music retrieval methods, we analyze the results of rhythmic comparison algorithm and list the analysis results in Table 5.10. These data show that the number of correct matches of rhythmic comparison results is lower than that of the K-S test and K-S test combined with mutual information and a window-based temporal parameter, but most of the correctly matched results are also exact matches.

Finally, the results of music retrieval by using the motion signal as queries are represented by creating motion animations with retrieved music tracks playing as background music. When integrating the motion recording with a music track, we either concatenate duplicates of the music piece, or truncate the music, depending on

Query Motion ID	Retrieved Music	
	Music ID	Music Track #
232432	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
23267	Take Five	36
232234	Titina	39
231821	Beethoven	3
23345	Waltz	42
205138	Waltz	42
20524	Waltz	42
231243	March	9 10 11
231351	March	10
231456	March	9 11
23175	March	10
233147	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
23288	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
232941	Minor Swing (Straight Version) Minor Swing (Swing Version)	18 26
233522	Montuno	30
23742	Take Five	38
23943	Seven	31 34
231121	Seven	31 33
23371	Take Five	35 36 38
233826	Take Five	38
233958	Take Five	38
23538	Take Five	36
232027	Titina	39
233234	Waltz	42
23336	Waltz	42

Table 5.9: Test results using rhythmic comparison algorithm

	Number of Matches
Incorrect Match	17
Correct Match	8
Exact Match	7
Total Number of Motion File	25

Table 5.10: Music retrieval results of rhythmic comparison algorithm

whether the music is shorter or longer than the target motion.

5.4 Discussions

Extracting the structural information from rhythmic signals such as human motion or regulated swinging object movement is a difficult problem, because these signals contain various patterns and representations which make the candidate beats ambiguous. Besides these difficulties of trying to extract the structural information, there might not always be a unique interpretation of the positions of rhythmical events. To recognize these patterns and learn the potential rules, we need to test different processing schemes to different data types, and discover a generally effective method. The procedure of extracting these patterns requires manual control over some parameters, such as in data pre-processing and threshold setting, which would become significant conditions that affect the detection results.

From the experimental data, we found several important parameter and threshold settings that can make a difference to the preliminary results and moreover, influence subsequent computations. These parameters are all independent and applied in different stages of the framework individually. Tuning the values of the following

parameters will result in different motion event vectors and therefore different music retrieval results:

- The filter size w_1. We use this to smooth the original data during the data pre-processing phase, as described in Section 2.2. It contributes to the rhythmic motion signal analysis and motion event extraction. In practice, we need to tune the filter size so that it is large enough to smooth out the noise, while small enough to keep the possible events detectable. The size of w_1 will directly influence the amount of extracted motion events, and further, may influence the motion rhythmic information detection process.

- The time window, $\triangle T$. This is used when calculating the angle between velocity vectors and generating the directional change curve. Variation of its length would result in different numbers and values of peaks in the directional change curve, and consequently different number of events which can be extracted from the curve. Its length could also influence the calculation of constant angular velocities motion events, which is also based on the motion directional change over $\triangle T$. As described in Section 3.1.2, by enlarging the length of $\triangle T$, we could obtain more obvious peaks and higher peak values.

- The number of continuous frames k for extracting constant angular velocities motion events (Section 3.1.6). We calculate the differences of movement directional changes for k consecutive frames, and if all of the values are less than a threshold, we mark all k of those frames as having constant angular velocities

(one type of our motion events). The larger k is, the fewer the number of such regions of constant angular velocities motion events. However, k should be kept large enough so that the motion event over the time duration of k frames is significant.

- Moving analysis window w_2 as introduced in Section 4.1. This is used to shift the music beat vector when comparing and aligning it with the motion music beat vector during the music retrieval process (when using mutual information with a window-based temporal parameter method). Currently, w_2 is set to be the average beat length of the current candidate motion music beat vector. Variation of w_2 could result in different music retrieval results (when using mutual information with a window-based temporal parameter).

- Analysis window w_3, for calculating the rhythmic comparison music query score in Section 4.3. If w_3 is too small, we may neglect the contribution of good rhythmic matches that are "close but not close enough". If w_3 is too large, then too many rhythmic events may all be considered "good matches", and we won't discriminate sufficiently between them.

Currently, the values of all the parameters discussed above, which are listed in Table 5.11, are determined by experimentation. Other parameter settings in our system are influential on the final results as well, for example the thresholds used to extract different levels of motion events, which is an important factor of motion music generation.

Variable Name	Place Applied	Value
w_1	Data Pre-processing (Section 2)	20 frames for human-character two-hand movement motions; 150 frames for CMU motion capture data
$\triangle T$	Movement directional change angle calculation (Section 3.1.2)	0.15s
k	constant angular velocity calculation (Section 3.1.6)	10 frames
w_2	Music retrieval by mutual information with a window-based temporal parameter (Section 4.1)	length of the average motion beats
w_3	Music retrieval by rhythmic comparison (Section 4.3)	0.15s

Table 5.11: Analysis windows and thresholds setting

Other than those crucial parameters, the beat tracking algorithm BeatRoot may also require manual tuning under some situations. As Dixon described [8], sometimes the BeatRoot system loses synchronization with the beat. For example, some Jazz music is full of variations and the rhythm changes from time to time, or the music changes in tempo. The beat tracking algorithm cannot always adjust its beat detection to these irregular patterns, hence, the manual control is necessary to help detecting beats more accurately. An example is shown in Figure 5.12, in which the red dots and blue crosses are two sequences of beats detected by BeatRoot for the same music piece "Abstract". Blue crosses refer to the beats detected by the algorithm originally and Red dots are the beat sequence detected with manual tuning.

As was mentioned in Section 5.3, a reasonable goal of the result testing is that the system could retrieve one of the variations of the music with which the motion file was recorded, or a music piece with similar rhythm. However, it is still difficult to measure the system's performance. Because the motion data and music are both perceptual, currently we evaluate whether a match is good or not by watching the animation. Good matches can be distinguished comparing to the "bad matches" (randomly selected music piece). In spite of that, some good matches might be rejected because of the observer's opinion. Hence, merely rely on those predefined "motion-to-music" relationships to categorize matches is not nearly enough. Those music pieces that could be good matches to a particular query motion but not its background music will be considered "incorrect matches" in the results summarization. To evaluate the system's performance more accurately, we could analyze the candidate music's rhyth-

mic relationships in advance to find music that is rhythmically similar. Hence each retrieved music could be categorized to see whether it belongs to the "good-match category" or not.

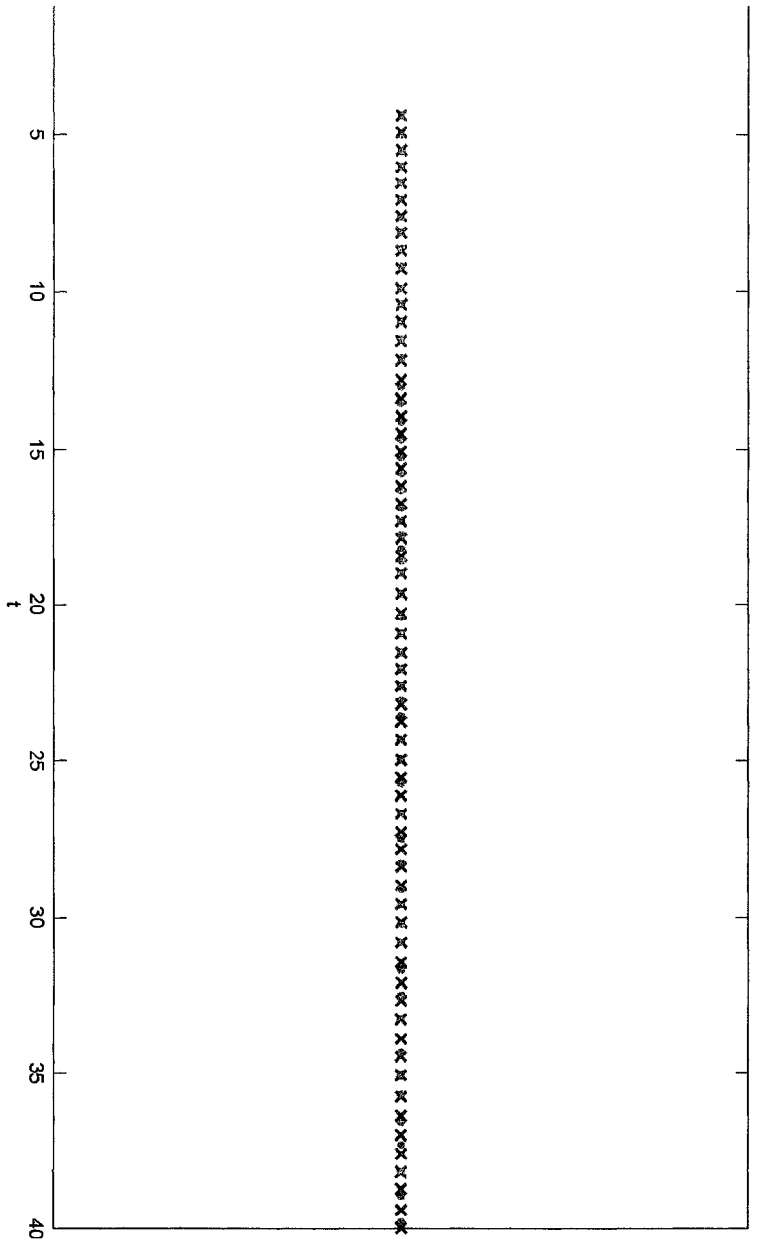

Figure 5.12: Comparison between originally detected beats (blue crosses) by "BeatRoot" and beats detected with manually tuning (red dots)

Chapter 6

Conclusion and Future Work

We have proposed a novel approach for querying music databases by using recorded motion input. This idea is demonstrated based on the assumption that the given motion signal can be related to an underlying rhythmic pattern similar to the rhythm of music.

Our scheme extracts the motion signal's rhythmic information as well as the music beats, and then with several similarity comparison methods, the music piece with highest rhythmic similarity score is retrieved as the match for the motion query. Given the captured motion signal, our scheme captures the significant movements and special patterns to detect "events" in the motion and then create a motion event vector. We build up a mapping relationship between motion event vector and MIDI music to compose motion music files. Together with the original music in the database, motion rhythm information can be extracted from those motion music files.

In this thesis, we defined the notion of motion events (3.1), displayed five dif-

ferent methods to explore motion characteristics and adopted four of them in the implementation to extract rhythmic motion events. We heuristically differentiate the motion events into six levels by their degree of significance. The thresholds which are used to distinguish different levels of events are hand tuned based on our experience. This scheme can successfully detect apparent movements for simple motions like human-character walking, or single ball bouncing. However, compared with the unlimited patterns and characteristics in human-character movements, our extracted motion features are not sufficient. We are planning to extend our system to include more parameters to capture more motion features and more patterns of movements. For human-character motion data, different feature extraction schemes regarding different parts of the body and relationship analysis between different joints could be added. Currently, our system only takes the human-character's hands, feet and root node into consideration. Analysis could also be extended to include more joints of the body, for example knee, elbow and head. Moreover, there could be more detailed definitions on different levels of motion events. We would like to add more parameters and conditions rather than separate motion events with a threshold value.

Another aspect to be explored is the corresponding features between the motion and music. Research has been done indicating the relationship between human motion and background music [52]. Currently, our scheme converts motion event vector to motion music directly with the "event-music note" mapping relationship (Table 3.1). More subtle and sophisticated music could be generated through learning the perceptual features contained in the motion signal.

Our work could be used to automatically match the query motion recording with a music piece. This idea could be extended to the inverse direction of querying a motion database with music files. Our system focuses on the comparison between overall motion/music rhythms but ignores the various tempo or meter changes in the middle of the music or motion. One possible future direction is to break the rhythmic analysis down into small sections of the music recording and of the motion. The retrieval of music/motion will be based on the global rhythmic similarity but the retrieved piece's play back rate can be adjusted to match with the query's partial tempo changes. Ultimately, our framework could be applied to synchronize music with video data (once similar features can be extracted from a video stream).

Content-based audio/video retrieval is a complicated and relatively new area that continues to be explored. The idea of detecting the rhythmic information from motion data is still new. Among the numerous possible solutions for this question, this work is only intended to demonstrate the feasibility of our proposed framework and ideas. There are various potential future directions, for which more extensive experiments and research will need to be conducted.

Appendix A

Beat tracking algorithm

This section briefly introduces the audio beat tracking method called "BeatRoot", based on the published paper by Dixon [7]. BeatRoot is an interactive beat tracking system written in Java. It has two interacting processes to detect the perceptual occurrence of beats: tempo induction, and beat tracking. The architecture of the system is shown in Figure A.1.

The system takes digital audio as input, and first processes the data to detect salient rhythmic events, which can be represented by musical notes. In the newest version of BeatRoot, an onset detector is applied to find peaks in the spectral flux. Then, a clustering algorithm is applied to calculate the clusters of inter-onset intervals (IOIs). Similar IOIs, simple integer multiples or fractions of each other representing various musical units, are also grouped for further analysis. With the information of groups and number of IOIs, clusters are ranked and a list of tempo hypotheses is generated.

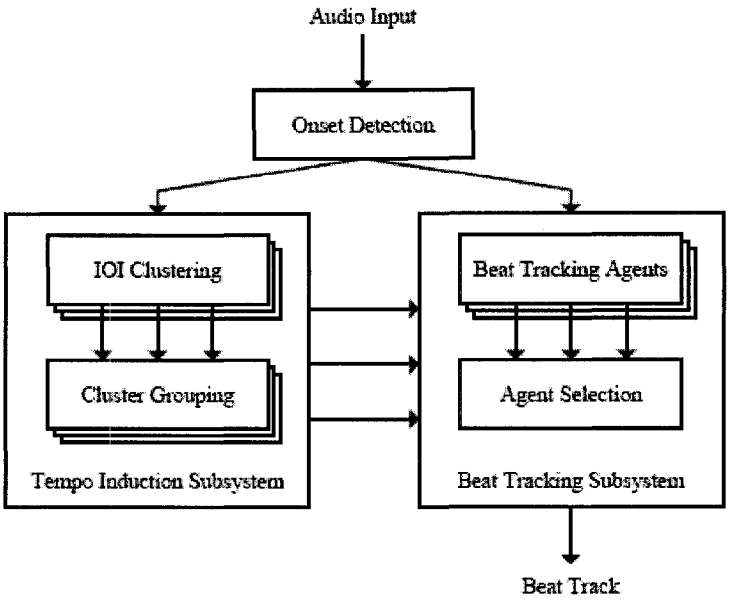

Figure A.1: System architecture of BeatRoot [7]

The beat tracking algorithm uses a multiple-agent architecture to rank various possible beat sequences to the music simultaneously. Each agent representing a tempo hypotheses, and is initialized with a tempo (rate) and an onset time, which defines the phase of beat sequence. Music is processed sequentially with the agents predicting the next beat times and matching them to the music. The scores of those matching are kept by each agent and evaluated based on how well the predicted beats match the musical events. At the end, the beat hypotheses with highest ranked score are selected as the solution.

Bibliography

[1] Adams, B., Dorai, C., and Venkatesh, S., 2002, "Finding the Beat: An Analysis of the Rhythmic Elements of Motion Pictures", *International Journal of Image and Graphics*, Vol. 2, No. 2, pp. 215-245.

[2] Bainbridge, D., Nevill-Manning, C.G., Witten, I.H., Smith, L.A., and McNab, R.J., 1999, "Towards a Digital Library of Popular Music", *Proceedings of the 4th ACM Conference on Digital Libraries*, Berkeley, CA, USA, pp. 161-169.

[3] Cardle, M., Brooks, S., Barthe, L., Hassan, M., and Robinson, P., 2002, " Music-driven Motion Editing", *Proceedings of ACM SIGGAPH conference Abstracts and Applications*, July 2002, San Antonio, Texas, USA, pp. 222-222.

[4] Cardle, M., Brooks, S., Bar-Joseph, Z., and Robinson, P., 2002, "Sound-by-Numbers: Motion-Driven Sound Synthesis", *Proceedings of ACM SIGGRAPH/Eurographics symposium on Computer Animation*, July 2003, San Diego, CA, USA, pp. 349-356.

[5] Cardle, M., Vlachos, M., Brooks, S., and Gunopulos, D., 2003, "Fast Motion Capture Matching with Replicated Motion Editing", *Proceedings of SIGGRAPH*

2003 Technical Sketches & Applications, July 2003, San Diego, CA, USA.

[6] Cemgil, T., Kappen, B., Desain, P., and Honing, H., 2001, "On Tempo Tracking: Tempogram Representation and Kalman Filtering", *Journal of New Music Research*, vol. 29, No. 4, pp. 259-273.

[7] Dixon, S., "Automatic Extraction of Tempo and Beat from Expressive Performances", *Journal of New Music Research*, Vol. 30, No. 1, pp 39-58.

[8] Dixon, S., 2007, "Evaluation of the Audio Beat Tracking System BeatRoot", *Journal of New Music Research*, Vol. 36, No. 1, pp. 39-50.

[9] Doel, K. V. D., Kry, P. G., and Pai, D. K., 2001, "FoleyAutomatic: Physically-based Sound Effects for Interactive Simulation and Animation", *Proceedings of ACM SIGGRAPH 2001 Conference*, pp. 537-544.

[10] Doraisamy, S., and Rüger, S., 2004, "A Polyphonic Music Retrieval System Using N-grams", *Proceedings of the International Conference on Music Information Retrieval*, Barcelona, Spain, pp. 204-209.

[11] Downie, S., and Nelson, M., 2000, "Evaluation of a Simple and Effective Music Information Retrieval Method", *Proceedings of the 23rd ACM International Conference on Research and Development in Information Retrieval*, Athens, Greece, pp. 73-80.

[12] Eck, D., 2006, "Beat Induction Using an Autocorrelation Phase Matrix", *The Proceedings of the 9th International Conference on Music Perception and Cognition*, pp. 931-932.

[13] Foote, J., and Cooper, M., 2002, "Automatic Music Summarization via Similarity Analysis", *Proceedings of 3rd International Symposium on Musical Information Retrieval*, September 2002, Paris, France, pp. 81-85.

[14] Foote, J., Cooper, M., and Girgensohn, A., 2002, "Creating Music Videos Using Automatic Media Analysis", *Proceedings of ACM Multimedia 2002*, December 2002, Juan-les-Pins, France, pp. 553-560.

[15] Foote, J., Cooper, M., and Nam, U., 2002, "Audio Retrieval by Rhythmic Similarity", *Proceedings of 3rd International Symposium on Musical Information Retrieval*, September 2002, Paris, France, pp. 265-266.

[16] Foote, J., Uchihashi, S., 2001, "The Beat Spectrum: A New Approach to Rhythm Analysis", *Proceedings of International Conference on Multimedia and Expo*, Tokyo, Japan, 2001.

[17] Goto, M., 2001, "An Audio-Based Real-Time Beat Tracking System for Music with or without Drum-Sounds", *Journal of New Music Research*, Vol. 30, pp. 159-171.

[18] Gouyon, F., Klapuri, A., Dixon, S., Alonso, M., Tzanetakis, G., and Uhle, C., 2006, " An Experimental Comparison of Audio Tempo Induction Algorithms",

IEEE Transactions on Audio, Speech and Language Processing, Vol. 14, No. 5, pp. 1832-1844.

[19] Hahn, J., Fouad, H., Gritz, L., and Lee, J.W., 1998, " Integrating Sounds and Motions in Virtual Environments", *Presence: Teleoperators and Virtual Environments*, Vol. 7, No. 1, pp. 66-67.

[20] Hasty, C., 1997, Meter As Rhythm, *Oxford University Press.*

[21] Hiraga, R., Mizaki, R., and Fujishiro, I., 2002, "Performance Visualization - a New Challenge to Music Through Visualization", *Proceedings of ACM Multimedia 2002*, December 2002, Juan-les-Pins, France, pp. 239-242.

[22] Hu, N., and Dannenberg, R.B., 2002, "A Comparison of Melodic Database Retrieval Techniques Using Sung Queries", *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital libraries*, Portland, Oregon, USA, pp. 301-307.

[23] Hua, X. S., Lu, L., and Zhang, H. J., 2003, "AVE: Automated Home Video Editing", *Proceedings of the 11th ACM International Conference on Multimedia*, Berkeley, CA, USA, pp. 490-497.

[24] Hua, X. S., Lu, L., and Zhang, H. J., 2004, "Optimization-Based Automated Home Video Editing System", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 14, No. 5, pp. 572-583.

[25] Kapur, A., Benning, M., and Tzanetakis, G., 2004, "Query-by-beat-boxing: Music Retrieval for the DJ", *Proceedings of the International Conference on Music Information Retrieval*, October 2004, Barcelona, Spain, pp. 170-177.

[26] Ke, Y., Hoiem, D., and Sukthankar, R., 2005, "Computer Vision for Music Identification," *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2005, San Diego, USA, Vol. 2, pp. 597-604.

[27] Kim, T., Park, S. I., and Shin, S. Y., 2003, "Rhythmic-Motion Synthesis Based on Motion-Beat Analysis", *ACM Transaction on Graphics*, Vol. 22, No. 3, pp. 392-401.

[28] Kovar, L., Gleicher, M.,and Pighin, F., 2002, "Motion Graphics", *ACM Transactions on Graphics*, June 2002, Vol. 21, No. 3, pp. 473-482.

[29] Lee, E., Enke, U., and Borchers, J., and Jong, L., 2007, "Towards Rhythmic Analysis of Human Motion using Acceleration-Onset Times", *Proceedings of the 7th International Conference on New Interfaces for Musical Expression*, New York, NY, USA, pp. 136-141.

[30] Lee, H. C., and Lee, I. K., 2005, "Automatic Synchronization of Background Music and Motion", *Computer Animation Computer Graphics Forum*, Vol. 24, No. 3, pp. 353-361.

[31] Lin, Y., 2006, "Efficient Human Motion Retrieval in Large Databases", *Proceedings of the 4th International Conference on Computer Graphics and Interactive*

Techniques in Australasia and Southeast Asia, Kuala Lumpur, Malaysia, pp. 31-37.

[32] Logan, B., and Salomon, A., 2001, "A Music Similarity Function Based on Signal Analysis", *Proceedings of IEEE International Conference on Multimedia and Expo*, August 2001, Tokyo, Japan, pp. 745-748.

[33] Lubiw, A., and Tanur, L., 2004, "Pattern Matching in Polyphonic Music As a Weighted Geometric Translation Problem", *Proceedings of the 5th International Conference of Music Information Retrieval*, October 2004, Barcelona, Spain, pp. 289-296.

[34] Luo, F., 2006, "Wavelet-Based Image Registration and Segmentation Framework for the Quantitative Evaluation of Hydrocephalus", *Master of Applied Science Thesis*, Saint Mary's University, Halifax, Canada.

[35] Melucci, M., and Orio, N., 1999, "Musical Information Retrieval Using Melodic Surface", *Proceedings of the 4th ACM Conference on Digital Libraries*, Berkeley, CA, USA, pp. 152-160.

[36] Meudic, B., 2002, "A Causal Algorithm for Beat-Tracking", *Proceedings of the 2nd Conference on Understanding and Creating Music*, Caserta, Italy.

[37] Nakamura, J., Kaku, T., Hyun, K., Noma, T., and Yoshida, S., 1994, "Automatic Background Music Generation Based on Actors' Mood and Motions", *Journal of Visualization and Computer Animation*, Vol. 5, No. 4, pp. 247-264.

[38] Nayak, M., Srunuvasan, S. H., and Kankanhalli, M. S., 2003, "Music Synthesis for Home Videos: An Analogy Based Approach", *Proceedings of IEEE Pacific-Rim Conference on Multimedia*, Vol. 3, pp. 1556-1560.

[39] Pampalk, E., Rauber, A., and Merkl, D., 2002, "Content-based Organization and Visualization of Music Archives", *Proceedings of the 10th ACM International Conference on Multimedia*, Juan-les-Pins, France, pp. 570-579.

[40] Pampalk, E., Dixon, S., and Widmer, G., 2004, "Exploring Music Collection by Browsing Different Views", *Computer Music Journal*, Vol. 28, No. 2, pp. 49-62.

[41] Penasse, V., and Nakamura, Y., 2003, "Dance Motion Synthesis with Music Synchronization", *Robotics Society of Japan*, Vol. 21, pp. 2J28.

[42] Pickens, J., and Crawford, T., 2002, "Harmonic Models for Polyphonic Music Retrieval", *Proceedings of the 11th International Conference on Information and Knowledge Management*, McLean, VA, USA, pp. 430-437.

[43] Pluim, J.P.W., Maintz, J.B. and Viergever, M.A., 2000, "Image Registration by Maximization of Combined Mutual Information and Gradient Information", *IEEE Transactions on Medical Imaging*, Vol. 19, No. 8, pp. 809-814.

[44] Selfridge-Field, E, 1997, Beyond MIDI: the Handbook of Musical Codes, *The MIT Press*, London.

[45] Sethares, W. A., and Staley, T. W., 2001, "Meter and Periodicity in Musical Performance", *Journal of New Music Research*, Vol. 30, pp. 149-158.

[46] Shiratori, T., Nakazawa, A., and Ikeuchi, K., 2003, "Rhythmic Motion Analysis Using Motion Capture and Musical Information", *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, July-August 2003, Tokyo, Japan, pp. 89-94.

[47] Temperley, D., 2004, "An Evaluation System for Metrical Models", *Computer Music Journal*, Vol. 28, No. 3, pp. 28-44.

[48] Tseng, Y., 1999, "Content-based Retrieval for Music Collections", *Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, CA, USA, pp. 176-182.

[49] Toole, C.O, Smeaton, A., Murphy, N., and Marlow, S., 1999, "Evaluation of Automatic Shot Boundary Detection on a Large Video Test Suite", *Proceedings of the 2nd UK Conference on Image Retrieval*, Newcastle, UK, pp. 1-12s.

[50] Whitney, J., 1981, Digital Harmony, *McGraw-Hill*, New York, NY.

[51] Yang, C., 2001, "Music Database Retrieval Based on Spectral Similarity", *Proceedings of the 2nd Annual International Symposium on Music Information Retrieval*, pp. 37-38.

[52] Yang, R., and Brown, M. S., 2004, "Music Database Query with Video by Synesthesia Observation", *2004 IEEE International Conference on Multimedia and Expo*, June 2004, Vol. 1, Issue 27-30, pp. 305-308.

[53] Yoon, J., and Lee, I. 2007, "Synchronized Background Music Generation for Video", *Proceedings of the International Conference on Advances in Computer Entertainment Technology*, Salzburg, Austria, pp. 270-271.

[54] Zaanen, M. V., Bod, R., and Honing, H., 2003, "A Memory-Based Approach to Meter Induction", *Proceedings of ESCOM 2003*, pp. 250-253.