Real-time Rhythm Tracking for Drumless Audio Signals — Chord Change Detection for Musical Decisions —

Masataka Goto and **Yoichi Muraoka**
School of Science and Engineering, Waseda University
3-4-1 Ohkubo Shinjuku-ku, Tokyo 169, JAPAN.
{goto, muraoka}@muraoka.info.waseda.ac.jp

Abstract

This paper presents a real-time rhythm tracking system that recognizes a rhythmic structure in musical audio signals without drum-sounds. Most previous systems dealt with MIDI signals and had difficulty in processing, in real time, audio signals containing sounds of various instruments and in tracking rhythm above the quarter-note level. Our system not only tracks beats at the quarter-note level but also recognizes rhythmic structure at the half-note and measure levels. To make musical decisions about the audio signals, we propose a method of detecting chord changes that does not rely on chord name identification. The method enables the system to understand music at different rhythmic levels — for example, to find the beginnings of half notes and measures — and to select the best of various hypotheses about beat positions. Experimental results show that our system is robust enough to handle audio signals sampled from compact discs of popular music.

1 Introduction

Although a great deal of music-related research has been undertaken, it is still difficult to build a computer system that can understand music. In terms of computational auditory scene analysis, one of the goals is to implement a computational model that can understand musical audio signals in a human-like fashion. A popular approach to this goal is to build an automatic music transcription system or a sound source separation system, which typically transforms audio signals into a symbolic representation such as a musical score or MIDI data. Although such detailed-transcription technologies are important, they have difficulty in dealing with compact disc audio signals in general. Because only a trained listener can identify musical notes and chord names, we can infer that musical transcription is an advanced skill difficult even for human beings to acquire.

On the other hand, an untrained listener understands music to some extent without mentally representing audio signals as musical scores. For example, even a listener who cannot identify chord names can sense harmony and chord changes. A listener who cannot completely segregate and identify every musical note can nevertheless track musical beats and keep time to music by hand-clapping or foot-tapping. We therefore think that it is important to first build a computer system that can understand music the way untrained human listeners do, without relying on transcription, and then extend the system so that it can understand music the way musicians do.

Our approach is to build a real-time rhythm-tracking system that recognizes a hierarchical musical structure of three rhythmic levels in real-world audio signals, such as those sampled from popular compact discs. Rhythm tracking is an important part of the computational modeling of music understanding because rhythm is fundamental, for both trained and untrained listeners, to the perception of Western music. Our system can understand music at the three rhythmic levels: the quarter-note level, the half-note level, and the measure (bar) level.[1] It not only finds the pulse sequence corresponding to the beats at the quarter-note level but also finds the beginnings of half notes and measures under the assumption that the time-signature is 4/4.

Most previous rhythm-tracking related systems [Dannenberg and Mont-Reynaud, 1987; Desain and Honing, 1989; 1994; 1995; Allen and Dannenberg, 1990; Driesse, 1991; Rosenthal, 1992a; 1992b; Rowe, 1993; Large, 1995; Smith, 1996] have dealt with MIDI signals or used onset times as their input. Since it is quite difficult to obtain complete MIDI representations from audio data, MIDI-based systems are limited in

[1] Although our system does not rely on score representation, for convenience here we use score-representing terminology like [Rosenthal, 1992a; 1992b]. The quarter-note level indicates the temporal basic unit that a human feels in music and that usually corresponds to a quarter note in scores.

their application. Although some systems [Schloss, 1985; Katayose *et al.*, 1989; Vercoe, 1994; Todd, 1994; Todd and Lee, 1994; Scheirer, 1996] dealt with audio signals, they either did not consider higher-level musical structure such as the half-note and measure levels or did not process, in real time, popular music sampled from compact discs. We developed a real-time beat-tracking system for audio signals [Goto and Muraoka, 1994; 1995a; 1995b], but it assumed that the input contained drum-sounds (a bass drum and a snare drum) and was not generally able to track beats in audio signals without drum-sounds. Moreover, it did not recognize rhythmic structure at the measure level.

In the following sections we describe how we extended our previous system so that it can deal with drumless audio signals and recognize the higher-level rhythmic structure. To make musical decisions about the audio signals, we propose a method of detecting chord changes. Because our method does not rely on identifying chord names, it can detect chord changes in audio signals sampled from compact discs, where chord identification is generally difficult.

2 Rhythm Tracking Problem

In this section we specify the rhythm tracking problem that we are dealing with and present the main difficulties of tracking rhythm.

2.1 Problem Specification

In our formulation, rhythm tracking is defined as a process that organizes music into a hierarchical musical structure with three levels of rhythm: *the quarter-note level*, *the half-note level*, and *the measure level* (Figure 1). The first step in solving our rhythm tracking problem is thus obtaining an appropriate sequence of *beat times* in an input musical audio signal. Beat times are temporal positions of almost regularly spaced beats corresponding to quarter notes and the sequence of beat times is called the quarter-note level. We then address the problem of finding the beginnings of half notes and measures. The sequence of *half-note times* (temporal positions of strong beats[2]) is obtained by determining whether a beat is *strong* or *weak* (*half-note-level type*). The sequence of *measure times* (temporal positions of the beginnings of measures) is obtained by determining whether a half note is the *beginning* or the *middle* of a measure (*measure-level type*). The sequence of half-note times is called the half-note level and the sequence of measure times is called the measure level. Both half-note-level and measure-level types are called the *beat types*.

[2]Under the assumption that the time-signature of an input song is 4/4, in this paper a *strong beat* is either the first or third quarter note in a measure; a *weak beat* is the second or fourth.

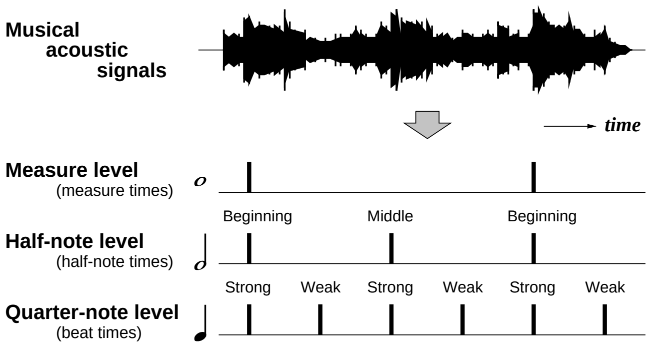


Figure 1: Rhythm tracking problem.

2.2 Acoustic Rhythm-Tracking Issues

The difficulties of tracking the rhythm in real-world musical acoustic signals can be summarized as (1) detecting rhythm-tracking cues in audio signals, (2) examining multiple hypotheses about beat positions, and (3) making musical decisions.

In detecting tracking cues, it is necessary to detect several cues for different purposes: finding beat times and recognizing the higher-level rhythmic structure. Our previous system [Goto and Muraoka, 1995b] addressed the first purpose by detecting onset times using frequency analysis and then obtaining beat times using autocorrelation and cross-correlation of the onset times. The tracking cues for the higher-level rhythmic structure of drumless audio signals, however, were not dealt with.

The multiple-hypothesis issue was addressed in our previous system [Goto and Muraoka, 1994; 1995a; 1996] by managing multiple agents that examined parallel hypotheses according to different strategies.

With regard to the musical-decision issue, our previous system [Goto and Muraoka, 1995a; 1995b] made use of prestored drum patterns, which were matched with the currently detected drum pattern of the input. Although this method was effective for audio signals with drum-sounds, it cannot be applied to the drumless audio signals we are considering here.

In this paper we address the main issue in extending the previous system to drumless audio signals and higher-level rhythm understanding. The issue is that higher-level processing using musical knowledge in addition to lower-level signal processing is indispensable for recognizing the higher-level rhythmic structure and evaluating which is the best interpretation of beat positions in an ambiguous situation. Musical knowledge that is useful for analyzing musical scores or MIDI signals, however, cannot immediately be applied to raw audio signals because of the difficulty of obtaining MIDI-like representations of those signals.

3 Chord Change Detection for Musical Decisions

To address the above-mentioned higher-level processing issue, we propose a method for making musical decisions based on chord changes. In the following sections, we first describe a method of detecting chord changes for obtaining rhythm-tracking cues for the higher-level rhythmic structure (Section 3.1), and then explain a way of making semantic decisions with heuristic musical knowledge based on the detected chord changes (Section 3.2).

3.1 Chord Change Detection

By making use of provisional beat times obtained on the basis of onset times (i.e., making use of top-down information from a beat-position hypothesis), this detection method examines possibilities of chord changes in a frequency spectrum without identifying any musical notes or chord names. The idea for this method came from the observation that a listener who cannot identify chord names can nevertheless sense chord changes.

When all frequency components which are included in chord tones and their harmonic overtones[3] are considered, they are found to tend to change significantly when a chord is changed and to be relatively stable when a chord is not changed. Although it is generally difficult to extract all frequency components from audio signals correctly and completely, dominant frequency components during a certain period of time can be estimated roughly by using a histogram of frequency components.

This method therefore calculates two kinds of possibilities of chord changes, at the quarter-note level and at the eighth-note level, by slicing the frequency spectrum into strips at the provisional beat times (top-down information). We call the former the *quarter-note chord change possibility* and the latter the *eighth-note chord change possibility*. The quarter-note and eighth-note chord change possibilities respectively represent how a chord is likely to change on each quarter-note position and on each eighth-note position under the current beat-position hypothesis. As described in Section 3.2, these possibilities are used for different purposes.

These possibilities are calculated as follows:

1. *Slicing the frequency spectrum into spectrum strips*
   The frequency spectrum (power spectrum) is calculated using the Fast Fourier Transform of the digitized audio signal (Section 4.1). In preparation for evaluating the quarter-note chord change possibility Cq_n, the frequency spectrum is sliced into spectrum strips Sq_n at the quarter-note times (beat times):

$$Sq_n = \{p(t,f)\},\ Tq_n \le t < Tq_{n+1} \qquad (1)$$

where Tq_n is the n-th beat time and $p(t,f)$ is the power of the spectrum of frequency f at time t.[4] On the other hand, in preparation for evaluating the eighth-note chord change possibility Ce_n, the spectrum is sliced into spectrum strips Se_n at the eighth-note times Te_n interpolated from Tq_n:

$$Se_n = \{p(t,f)\},\ Te_n \le t < Te_{n+1} \qquad (2)$$

$$Te_n = \begin{cases} Tq_{n/2} & (n \bmod 2 = 0) \\ (Tq_{(n-1)/2} + Tq_{(n+1)/2})/2 & \\ & (n \bmod 2 = 1) \end{cases} \qquad (3)$$

2. *Forming histograms*
   The system forms histograms $Hq_n(f)$ and $He_n(f)$ (after this, we will use abbreviations such as $Hi_n(f)$ $(i = q, e)$) summed up along the time axis in the corresponding strip Sq_n and Se_n:

$$Hi_n(f) = \sum_{t=Ti_n+Gi_n}^{Ti_{n+1}-Gi_n} p(t,f) \qquad (4)$$

where Gi_n (equal to $(Ti_{n+1} - Ti_n)/5$ in our current implementation) is a margin to avoid influences of noises and unstable frequency components around the note onset.

3. *Detecting dominant frequencies*
   First, peaks $Ki_n(f)$ along the frequency axis in $Hi_n(f)$ are given by

$$Ki_n(f) = \begin{cases} Hi_n(f) & \text{if } Hi_n(f) \ge \\ & \quad Hi_n(f \pm 1) \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$

Our current implementation considers only peaks whose frequency is between 10 Hz and 1 kHz. These peaks can be considered the dominant tones' frequencies in each strip and tend to correspond to frequency components of a chord or a melody.

These peaks are then transformed in order to avoid detecting unnecessary noise peaks during a silent period such as a rest and to consider that the previous chord continues during its period. Considering temporal transition of the maximum mi_n of $Ki_n(f)$, we can express the transformed peaks $Qi_n(f)$ as

$$Qi_n(f) = \mathrm{clip}(\Psi Ki_n(f) \,/\, Mi_n) \qquad (6)$$

$$Mi_n = \max(mi_n, \Phi Mi_{n-1}) \qquad (7)$$

$$\mathrm{clip}(x) = \begin{cases} 0 & (x < 0) \\ x & (0 \le x \le 1) \\ 1 & (1 < x) \end{cases} \qquad (8)$$

where Ψ $(= 5)$ and Φ $(= 0.99)$ are constant values. Finally the desired peaks $Pi_n(f)$ in each strip are

[3] In the case of actual songs, frequency components of a melody and other parts are also considered. These components tend to be in harmony with chord tones.

[4] f and t are integers, and $1f$ and $1t$ are respectively equal to the frequency resolution (10.77 Hz) and the discrete time step (11.61 ms).

calculated so that the previous peaks $Pi_{n-1}(f)$ can be regarded as continuing during a relatively silent period in which the sum of $Qi_n(f)$ is low:

$$Pi_n(f) = \begin{cases} Qi_n(f) & \text{if } \sum_f Qi_n(f) \geq \\ & \quad \Theta \sum_f Pi_{n-1}(f) \\ Pi_{n-1}(f) & \text{otherwise} \end{cases} \quad (9)$$

where Θ (= 0.1) is a constant value. This makes it possible to prevent the chord change possibilities from increasing rapidly after every silent period.

4. *Comparing frequencies between adjacent strips*

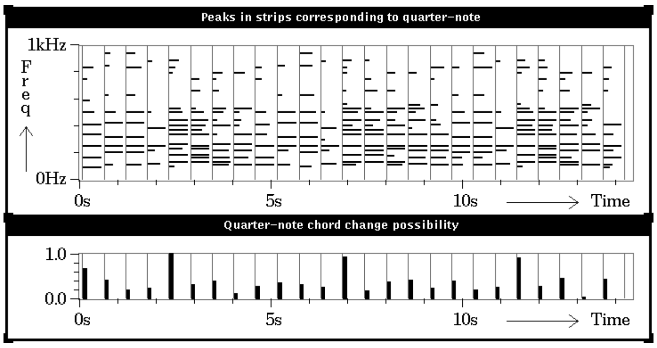
   The chord change possibilities are calculated by comparing peaks between adjacent strips: $Pi_{n-1}(f)$ and $Pi_n(f)$. When a chord is changed at the boundary time Ti_n between those strips, different peaks tend to occur in $Pi_n(f)$ compared with $Pi_{n-1}(f)$. Considering temporal transition of positive differences ci_n, we can express the quarter-note chord change possibility Cq_n and the eighth-note chord change possibility Ce_n as

$$Ci_n = ci_n / di_n \quad (10)$$

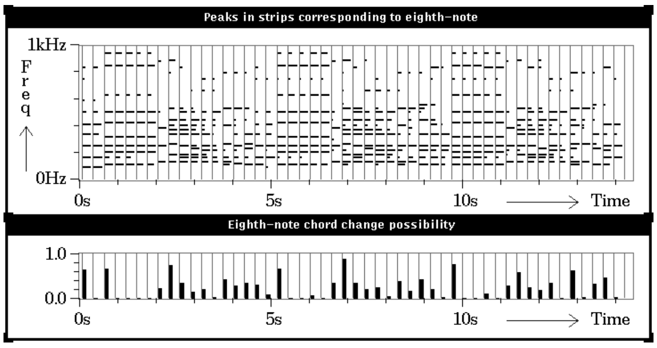
$$ci_n = \sum_f \text{clip}(Pi_n(f) - Pi_{n-1}(f)) \quad (11)$$

$$di_n = \max(ci_n, \Phi\, di_{n-1}) \quad (12)$$

Figure 2 shows examples of two kinds of chord change possibilities. The thin vertical lines represent the quarter-note times Tq_n in (a) and the eighth-note times Te_n in (b). The beginning of measure occurs at every four quarter-note times from the extreme left in (a), and the beat occurs at every two eighth-note times from the extreme left in (b). In each of (a) and (b), the horizontal lines above represent the peaks $Pi_n(f)$ in each strip and the thick vertical lines below represent the chord change possibility Ci_n.



(a) Examining quarter-note chord change possibility



(b) Examining eighth-note chord change possibility

Figure 2: Examples of peaks in sliced frequency spectrum and of chord change possibilities.

3.2 Musical Decisions

By utilizing the two kinds of chord change possibilities, the system recognizes the higher-level rhythmic structure (i.e., determines the half-note times and the measure times) and selects the best hypothesis from various agent-generated hypotheses about beat positions. For these purposes, we introduce the following two kinds of heuristic musical knowledge.

(1) Quarter-note-level knowledge

   Chords are more likely to change at the beginnings of measures than at other positions. In other words, the quarter-note chord change possibility tends to be higher on a strong beat than on a weak beat and higher on the strong beat at the beginning of a measure than on the other strong beat in the measure.

(2) Eighth-note-level knowledge

   Chords are more likely to change on beats (quarter notes) than on other positions between two successive correct beats. In other words, the eighth-note chord change possibility tends to be higher on beats than on eighth-note displacement positions.

To recognize the higher-level rhythmic structure, the system utilizes the quarter-note-level knowledge (1). It first calculates Uq_n which represents a past tendency of every other quarter-note chord change possibility and Uh_n which represents a past tendency of every four quarter-note chord change possibility:

$$Uq_n = \lambda_1\, Uq_{n-2} + \lambda_2\, Cq_n \quad (13)$$

$$Uh_n = \lambda_1\, Uh_{n-4} + \lambda_2\, Cq_n \quad (14)$$

where λ_1 (= 0.99) and λ_2 (= 0.2) are constant values. If $Uq_n - Uq_{n-1} > \mu_q$, it then judges that the position of a half-note time is Tq_n, where μ_q (= 0.3) is a constant threshold. If Tq_n is a half-note time and $Uh_n - Uh_{n-2} > \mu_h$, it judges that the position of a measure time is Tq_n, where μ_h (= 0.2) is a constant threshold. The reliabilities of these judgements are defined as

$$Lq_n = \text{clip}(Uq_n - Uq_{n-1}) \quad (15)$$

$$Lh_n = \text{clip}(Uh_n - Uh_{n-2}) \quad (16)$$

Based on the previous position of a half-note time and a measure time, the following beat types (half-note-level type and measure-level type) are determined under the assumptions that *strong* and *weak* alternate on beat times and that *beginning* and *middle* alternate on half-note times.

To select the best hypothesis, the system utilizes the eighth-note-level knowledge (2). As described in Section 4.2, the final output is determined on the basis of the appropriate hypothesis that has the highest reliability. To evaluate the reliability of a hypothesis, the system calculates Le_n, which is the reliability of the judgement that Tq_n $(= Te_{2n})$ is the position of a beat:

$$Le_n = \lambda_1 \, Le_{n-1} + \lambda_2 \, (Ce_{2n} - Ce_{2n+1}) \qquad (17)$$

If Le_n becomes higher (i.e., the eighth-note chord change possibility keeps on being higher on beats than on other positions), the reliability is increased so that the system can select the hypothesis under which the appropriate Ce_n is obtained. The reliability is also evaluated by different viewpoints as described in Section 4.2.

4 System Description

The system for musical audio signals without drum-sounds[5] assumes that the time-signature of an input song is 4/4 and that its tempo is constrained to be between 61 M.M. (Mälzel's Metronome: the number of quarter notes per minute) and 120 M.M., and is roughly constant. The system maintains, as the real-time output, a description called *beat information* (*BI*) that consists of the beat time, its beat types, and the current tempo.

[5]A detailed description of our beat-tracking system for audio signals that include drum-sounds is presented in [Goto and Muraoka, 1995a; 1995b].

Figure 3 shows an overview of our rhythm tracking system. The system first digitizes an input audio signal in the *A/D Conversion* stage. In the *Frequency Analysis* stage, multiple onset-time finders detect onset times in different ranges of the frequency spectrum, and those results are transformed into vectorial representation (called *onset-time vectors*) by *onset-time vectorizers*. In the *Beat Prediction* stage, the system manages multiple agents that, according to different strategies, make parallel hypotheses based on those onset-time vectors. Each agent first calculates the *inter-beat interval* (the temporal difference between two successive beats) and predicts the next beat time. By communicating with a *chord change checker*, it then determines its beat types and evaluates the reliability of its own hypothesis. A *hypotheses manager* gathers all hypotheses and then determines the final output on the basis of the most reliable one. Finally, in the *BI Transmission* stage, the system transmits BI to other application programs via a computer network.

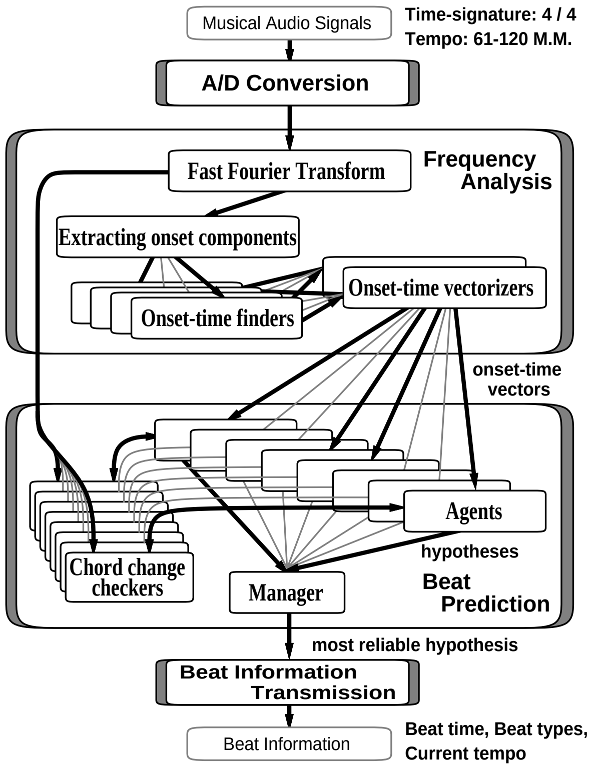


Figure 3: Overview of our rhythm tracking system.

4.1 Frequency Analysis

In the *Frequency Analysis* stage, the frequency spectrum and several sequences of N-dimensional onset-time vectors are obtained for later processing (Figure 4). The full frequency band is split into several frequency ranges, and each dimension of the onset-time vectors corresponds to

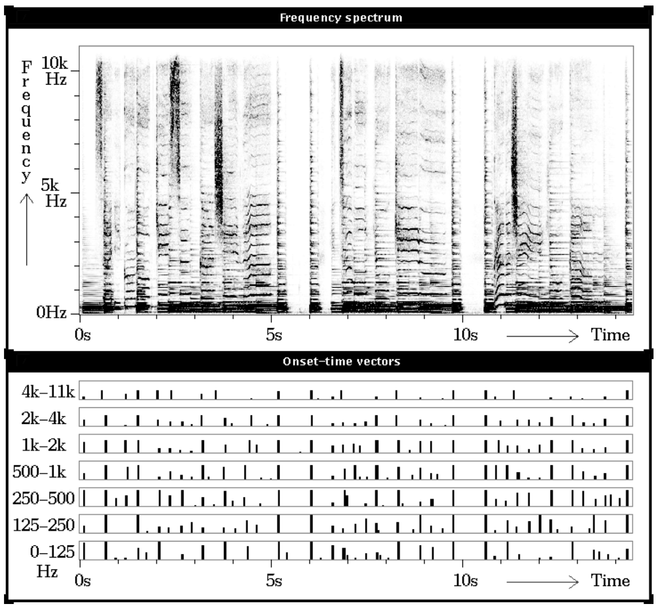


Figure 4: Example of a frequency spectrum and an onset-time vector sequence.

a different frequency range. This representation makes it possible to consider onset times of all the frequency ranges at the same time.

Fast Fourier Transform (FFT)

The frequency spectrum is calculated with the FFT using the Hanning window. Each time the FFT is applied to the input signal, the window is shifted to the next frame.

In our current implementation, the input signal is digitized at 16 bit / 22.05 kHz, and two kinds of FFT are calculated. One FFT, for extracting onset components in the Frequency Analysis stage, is calculated with a window size of 1024 samples, and the window is shifted by 256 samples. The frequency resolution is consequently 21.53 Hz and the discrete time step (1 *frame-time*[6]) is 11.61 ms. The other FFT, for examining chord changes in the Beat Prediction stage, is simultaneously calculated in audio down-sampled at 16 bit / 11.025 kHz with a window size of 1024 samples, and the window is shifted by 128 samples. The frequency resolution and the time step are consequently 10.77 Hz and 1 frame-time.

Extracting Onset Components

The frequency component $p(t, f)$ that fulfills Condition (18) is extracted as an onset component.

$$\min(p(t, f), p(t+1, f)) > pp \tag{18}$$

$$pp = \max(p(t-1, f), p(t-1, f \pm 1)) \tag{19}$$

Its degree of onset $d(t, f)$ (rapidity of increase in power) is given by

$$d(t,f) = \begin{cases} \max(p(t,f), p(t+1,f)) - pp \\ \qquad \text{if Condition (18) is fulfilled} \\ 0 \quad \text{otherwise} \end{cases} \tag{20}$$

Onset-time Finders

Multiple onset-time finders (seven in our current implementation) detect onset times in several different frequency ranges (0-125 Hz, 125-250 Hz, 250-500 Hz, 500 Hz-1 kHz, 1-2 kHz, 2-4 kHz, and 4-11 kHz). Each onset time is given by the peak time found by peak-picking in the sum $D(t)$ along the time axis, where $D(t) = \sum_f d(t, f)$. $D(t)$ is linearly smoothed with a convolution kernel before its peak time is calculated. Limiting the frequency range of $\sum_f$ makes it possible to find onset times in the different frequency ranges.

Onset-time Vectorizers

Each onset-time vectorizer transforms the results of all onset-time finders into a sequence of onset-time vectors: the same onset times in all the frequency ranges are put together into one vector. In the current system, three vectorizers transform onset times from seven finders into three sequences of seven-dimensional onset-time vectors with the different sets of frequency weights (focusing on all/low/middle frequency ranges). These results are sent to agents in the *Beat Prediction* stage.

[6]The frame-time is the unit of time used in our system, and the term *time* in this paper is the time measured in units of the frame-time.

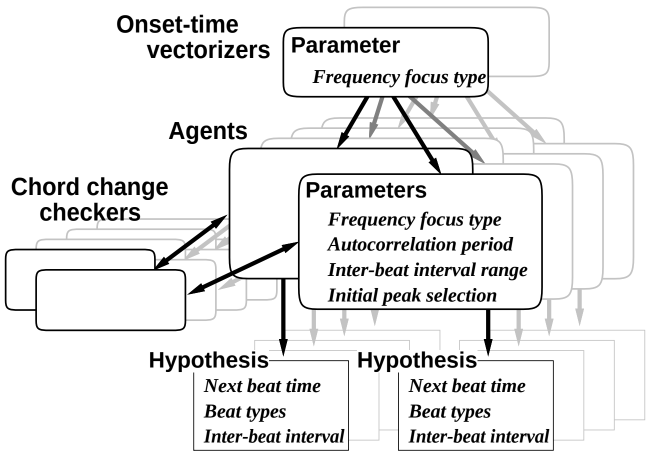


Figure 5: Relations between onset-time vectorizers, agents, and chord change checkers.

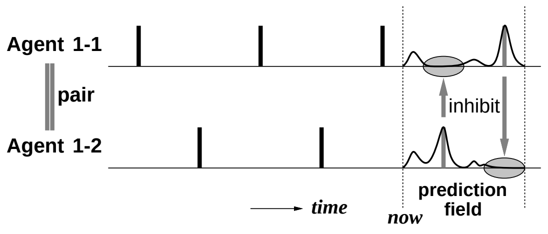


Figure 6: Agent interaction through a prediction field.

4.2 Beat Prediction

Multiple agents interpret the sequences of onset-time vectors according to different strategies and maintain their own hypotheses. Each hypothesis consists of a predicted next-beat time, its beat types (half-note-level type and measure-level type), and the current inter-beat interval (Figure 5). These hypotheses are gathered by the manager and the most reliable one is considered the final output.

All agents are grouped into pairs.[7] Two agents in the same pair examine the same inter-beat interval and cooperatively predict the next beat times; their two predictions will always differ by half the inter-beat interval. For this purpose, one agent interacts with the other agent through a *prediction field*, which is an expectancy curve[8] that represents the time that the next beat is ex-

[7]In our current implementation there are twelve agents grouped into six agent-pairs.

[8]Other systems [Desain, 1992; Desain and Honing, 1994; Vercoe, 1994] have used a similar concept of expectancy curve for predicting future events, but not for managing interactions between agents.

pected to occur (Figure 6). The height of each local peak in the prediction field can be interpreted as the probability that the next beat is at that position. The two agents interact with each other by inhibiting the prediction field in the other agent. The beat time of each hypothesis reduces the probability of a beat in the temporally corresponding neighborhood in the other's field.

Each agent has the following four parameters that determine its strategy for making the hypothesis (Figure 5). Initial settings of the parameters are listed in Table 1.

1. *frequency focus type*

   This determines which vectorizer an agent receives onset-time vectors from. This value is chosen from among *type-all*, *type-low*, and *type-mid*, respectively corresponding to vectorizers focusing on all frequency ranges, low frequency ranges, and middle frequency ranges.

2. *autocorrelation period*

   This determines the window size for calculating the vectorial autocorrelation (described later) of the onset-time vector sequence. The greater this value, the older the onset-time information considered.

3. *inter-beat interval range*

   This controls the range of possible inter-beat intervals. As described later, this limits the range of selecting a peak in the result of the vectorial autocorrelation.

4. *initial peak selection*

   This takes a value of either *primary* or *secondary*. When the value is *primary*, the largest peak in the prediction field is initially selected, and the peak is considered the next beat time; when the value is *secondary*, the second largest peak is initially selected. This helps generate a variety of hypotheses.

Table 1: Initial settings of the strategy parameters.

pair-agent	frequency focus type	auto-correlation period	inter-beat interval range	initial peak selection
1-1	type-all	500 f.t.	43-85 f.t.	primary
1-2	type-all	500 f.t.	43-85 f.t.	secondary
2-1	type-all	1000 f.t.	43-85 f.t.	primary
2-2	type-all	1000 f.t.	43-85 f.t.	secondary
3-1	type-low	500 f.t.	43-85 f.t.	primary
3-2	type-low	500 f.t.	43-85 f.t.	secondary
4-1	type-low	1000 f.t.	43-85 f.t.	primary
4-2	type-low	1000 f.t.	43-85 f.t.	secondary
5-1	type-mid	500 f.t.	43-85 f.t.	primary
5-2	type-mid	500 f.t.	43-85 f.t.	secondary
6-1	type-mid	1000 f.t.	43-85 f.t.	primary
6-2	type-mid	1000 f.t.	43-85 f.t.	secondary

"f.t." is the abbreviation of frame-time (11.61 ms).

The following describe the formation of hypotheses, the chord change checkers, and the management of hypotheses.

Beat-predicting Agents

Each agent makes a hypothesis as follows and sends it to the one-to-one corresponding chord change checker and the manager.

(1) *Determining the inter-beat interval*

To determine the inter-beat interval, each agent receives the sequence of onset-time vectors and calculates its vectorial autocorrelation.[9] The windowed and normalized vectorial autocorrelation function $Ac(\tau)$ is defined as

$$Ac(\tau) = \frac{\sum_{t=c-W}^{c} \mathrm{win}(c-t, W)\,(\vec{o}(t) \cdot \vec{o}(t-\tau))}{\sum_{t=c-W}^{c} \mathrm{win}(c-t, W)\,(\vec{o}(t) \cdot \vec{o}(t))} \quad (21)$$

where $\vec{o}(t)$ is the N-dimensional onset-time vector at time t, c is the current time, and W is the strategy parameter *autocorrelation period*. The window function $\mathrm{win}(t, s)$ whose window size is s is given by

$$\mathrm{win}(t, s) = 1.0 - 0.5\,\frac{t}{s} \quad (22)$$

The inter-beat interval is given by the τ with the maximum height in $Ac(\tau)$ within the range limited by the parameter *inter-beat interval range*. If the reliability of a hypothesis becomes high enough, its agent tunes this parameter to narrow the range of possible inter-beat intervals so that it examines only a neighborhood of the current appropriate one.

(2) *Predicting the next beat time*

To predict the next beat time, each agent forms a prediction field (Figure 7). The prediction field is the result of calculating the windowed cross-correlation function $Cc(\tau)$ between the sum $O(t)$ of all dimensions of $\vec{o}(t)$ and the provisional beat time sequence $B_m(t)$ whose interval is the inter-beat interval obtained using Equation (21):

$$Cc(\tau) = \sum_{t=c-V}^{c} \Big(\mathrm{win}(c-t, V)\, O(t) \sum_{m=1}^{\Omega} \delta(t - B_m(c+\tau))\Big) \quad (23)$$

$$B_m(t) = \begin{cases} t - I(t) & (m = 1) \\ B_{m-1}(t) - I(B_{m-1}(t)) & (m > 1) \end{cases} \quad (24)$$

$$\delta(x) = \begin{cases} 1 & (x = 0) \\ 0 & (x \neq 0) \end{cases} \quad (25)$$

where $I(t)$ is the inter-beat interval at time t, V ($= \Omega I(c)$) is the window size for calculating cross-correlation, and Ω ($= 12$) is a constant value. The

[9]The paper [Vercoe, 1994] also proposed the use of a variant of autocorrelation for rhythmic analysis.

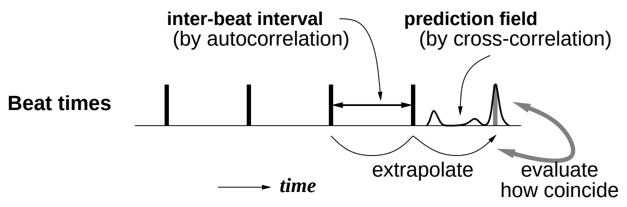


Figure 7: Predicting the next beat.

prediction field is thus given by $Cc(\tau)$ where $0 \leq \tau \leq I(c) - 1$.

Each agent then selects the next beat time from local peaks in the prediction field after the field is inhibited by its paired agent. When the reliability of a hypothesis is low, the agent initially selects the peak in the prediction field according to the parameter *initial peak selection*, and then tries to pursue the peak close to the sum of the previously selected one and the inter-beat interval.

(3) *Judging the beat types*

Each agent determines the beat types of the predicted beat time according to the half-note time and the measure time. As described in Section 3.2, these times are obtained from the quarter-note chord change possibility received from the corresponding chord change checker.

(4) *Evaluating the reliability of its own hypothesis*

Each agent finally evaluates the reliability of its own hypothesis in the following three steps. First, the reliability is evaluated according to how the next beat time predicted on the basis of the onset times coincides with the time extrapolated from the past beat times (Figure 7). If they coincide, the reliability is increased; otherwise, the reliability is decreased. Second, the reliability is evaluated according to how appropriate the eighth-note chord change possibility is. If Le_n (defined in Section 3.2) is higher, the reliability is increased; otherwise, the reliability is decreased. Third, the reliability is evaluated according to how appropriate the quarter-note chord change possibility is. If Lq_n is higher, the reliability is increased a little.

Chord Change Checkers

Each chord change checker examines the two kinds of chord change possibilities as described in Section 3.1. It analyzes the frequency spectrum on the basis of beat times (top-down information) received from the corresponding agent, and it sends the possibilities back to the agent (Figure 5).

Hypotheses Manager

The manager classifies all agent-generated hypotheses into groups according to beat time and inter-beat interval. Each group has an overall reliability given by the sum of the reliabilities of the group's hypotheses. The manager then selects the dominant group that has the highest reliability. Since an incorrect group could be selected if temporarily unstable beat times split the appropriate dominant group, the manager repeats grouping and selecting three times while narrowing the margin of beat times allowable for being classified into the same group. The reliable hypothesis in the most dominant group is thus selected as the output and sent to the BI Transmission stage.

The manager updates the beat types in the output using only the beat types that were labeled when Lq_n and Lh_n were high compared with the recent maximum, since the beat types labeled by each agent might be incorrect because of a local irregularity of chord changes or a detection error.

5 Experiments and Results

We tested the system implemented on a distributed-memory parallel computer, the Fujitsu AP1000, consisting of 64 processing elements. In the following, we describe an experimental result of testing the proposed method of detecting chord changes (Section 5.1) and describe the overall recognition rates of the system (Section 5.2).

5.1 Performance Test of Chord Change Detection

We tested the basic performance of the proposed method of chord change detection by using a random chord progression. This chord progression consisted of one hundred chord transitions of 101 chords that were randomly selected from sixty kinds of chords: the twelve kinds of root (A, A♯, B, C, C♯, D, D♯, E, F, F♯, G, G♯) with the five chord types (major triad, minor triad (m), dominant 7th chord (7), minor 7th chord (m7), major 7th chord (M7)). These chords were so selected that the adjacent chords were different. Using a synthesizer's piano tone, we played them in the basic root position (close position voicing). The fundamental frequency of the chord root note was between 110 Hz and 208 Hz. To examine the case in which the chord did not change, we played each chord twice with the duration of a quarter note (600 ms) under the tempo 100 M.M.

The mean, standard deviation (SD), maximum, and minimum of the quarter-note chord change possibility Cq_n and the eighth-note chord change possibility Ce_n

Table 2: Results of testing chord change detection.

	Cq_n		Ce_n	
	CH	NC	on Tq_n (CH, NC)	off Tq_n
mean	0.73	0.01	0.56 (0.81, 0.30)	0.03
SD	0.22	0.02	0.29 (0.18, 0.08)	0.05
max.	1.00	0.10	1.00 (1.00, 0.48)	0.21
min.	0.28	0.00	0.12 (0.37, 0.12)	0.00

CH: chord change. NC: no chord change.

when the appropriate beat times were provided for slicing the frequency spectrum are listed in Table 2. The "CH" and "NC" of the Cq_n in Table 2 respectively mean the Cq_n when a chord was changed at Tq_n and the Cq_n when a chord was not changed at Tq_n. The values listed in these columns indicate that the Cq_n at chord changes (CH) were appropriately higher than at the others (NC).

On the other hand, the "on Tq_n" and "off Tq_n" of the Ce_n respectively mean the Ce_n on beats ($n \bmod 2 = 0$) and the Ce_n on eighth-note displacement positions ($n \bmod 2 = 1$). In the case of the "on Tq_n," because the chord-change case (CH) alternated with the no-chord-change case (NC), these cases were also analyzed separately. The values listed in these columns indicate that chord changes were appropriately detected using Ce_n. The Ce_n of NC of "on Tq_n" tended to be higher than the Ce_n of "off Tq_n" because the chord notes were always played at a beat time, whereas all frequency components on an eighth-note displacement position had lasted from the previous beat time.

5.2 Overall Recognition Rates

We tested the system on monaural audio signals that were sampled from commercial compact discs of popular music and contained the sounds of various instruments (not including drums). The initial one or two minutes of 40 songs performed by 28 artists were used as the inputs. The time-signature was 4/4 and the tempi ranged from 62 M.M. to 116 M.M. and were roughly constant.

In our experiment the system obtained the correct beat time (tracked beats at the quarter-note level) in 35 of the 40 songs (87.5 %)[10] and determined the correct half-note-level type (tracked beats at the half-note level) in 34 of the 35 songs (97.1 %) in which the correct beat times were obtained. Moreover, it determined the correct measure-level type (tracked beats at the measure level) in 32 of the 34 songs (94.1 %) in which the half-note-level type was correct.

We evaluated how quickly the system started to track the correct rhythm stably at each rhythmic level. The mean, standard deviation, maximum, and minimum of the tracking start time of all the correctly tracked songs are listed in Table 3. In each song where the rhythmic structure was eventually determined correctly, the system initially had trouble determining the beat types even though the beat time was correct.

[10]In evaluating the recognition accuracy of our system, we did not count unstably tracked songs in which correct beats were obtained just temporarily.

Table 3: Start time of tracking the correct rhythm at the quarter-note, half-note, and measure levels.

rhythmic level	Quarter-note	Half-note	Measure
mean	10.71 sec	14.70 sec	20.70 sec
SD	9.05 sec	9.21 sec	9.95 sec
max.	35.77 sec	42.56 sec	42.56 sec
min.	0.79 sec	3.42 sec	3.42 sec

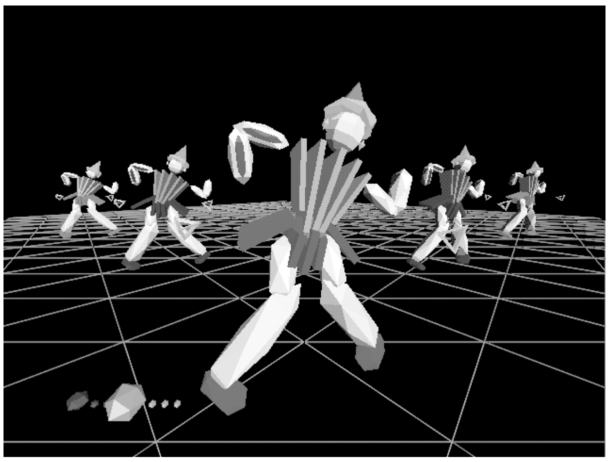


Figure 8: Virtual dancers synchronized with musical beats.

The beat times were not obtained correctly in the five songs because onset times were very few and irregular or the tempo fluctuated temporarily. Consequently, the chord change possibilities in those songs could not be obtained appropriately. The main reason why the half-note-level or measure-level type was incorrect in the other mistaken songs was irregularity of chord changes, such as chords changing at every quarter-note or every other quarter-note.

These results show that the system is robust enough to deal with real-world musical signals and that it recognizes the musical structure of three rhythmic levels.[11] We have also developed an application of our rhythm-tracking system that displays real-time computer graphics dancers whose motions change in time to musical beats (Figure 8) and confirmed that the system is useful in the real application.

6 Conclusion

We have described the main rhythm-tracking problem and solution in dealing with drumless audio signals and have described the configuration and implementation of our real-time rhythm-tracking system. The experimental results show that the system can track, in real time, beats at the quarter-note level, the half-note level, and the measure level in audio signals sampled from compact discs of popular music.

We proposed a method for detecting chord changes by analyzing the frequency spectrum sliced at provisional beat times (top-down information). We think that such

[11]The rhythm-tracking results are further evaluated in [Goto and Muraoka, 1997].

an approach, without chord name identification, is meaningful because a person generally does not perceive music as musical symbols. This method enabled our system to determine the beat types in audio signals without drum-sounds and to select the appropriate hypothesis from multiple agent-generated hypotheses.

We plan to upgrade the system to generalize to other musical genres, to follow tempo changes, and to make use of other higher-level musical structure.

Acknowledgments

We thank David Rosenthal for his helpful comments on earlier drafts of this paper. We also thank Fujitsu Laboratories Ltd. for use of the AP1000.

References

[Allen and Dannenberg, 1990] Paul E. Allen and Roger B. Dannenberg. Tracking musical beats in real time. In *Proc. of the 1990 Intl. Computer Music Conf.*, pages 140–143, 1990.

[Dannenberg and Mont-Reynaud, 1987] Roger B. Dannenberg and Bernard Mont-Reynaud. Following an improvisation in real time. In *Proc. of the 1987 Intl. Computer Music Conf.*, pages 241–248, 1987.

[Desain and Honing, 1989] Peter Desain and Henkjan Honing. The quantization of musical time: A connectionist approach. *Computer Music Journal*, 13(3):56–66, 1989.

[Desain and Honing, 1994] Peter Desain and Henkjan Honing. Advanced issues in beat induction modeling: syncopation, tempo and timing. In *Proc. of the 1994 Intl. Computer Music Conf.*, pages 92–94, 1994.

[Desain and Honing, 1995] Peter Desain and Henkjan Honing. Computational models of beat induction: the rule-based approach. In *Working Notes of the IJCAI-95 Workshop on Artificial Intelligence and Music*, pages 1–10, 1995.

[Desain, 1992] Peter Desain. Can computer music benefit from cognitive models of rhythm perception? In *Proc. of the 1992 Intl. Computer Music Conf.*, pages 42–45, 1992.

[Driesse, 1991] Anthonie Driesse. Real-time tempo tracking using rules to analyze rhythmic qualities. In *Proc. of the 1991 Intl. Computer Music Conf.*, pages 578–581, 1991.

[Goto and Muraoka, 1994] Masataka Goto and Yoichi Muraoka. A beat tracking system for acoustic signals of music. In *Proc. of the Second ACM Intl. Conf. on Multimedia*, pages 365–372, 1994.

[Goto and Muraoka, 1995a] Masataka Goto and Yoichi Muraoka. Music understanding at the beat level — real-time beat tracking for audio signals —. In *Working Notes of the IJCAI-95 Workshop on Computational Auditory Scene Analysis*, pages 68–75, 1995.

[Goto and Muraoka, 1995b] Masataka Goto and Yoichi Muraoka. A real-time beat tracking system for audio signals. In *Proc. of the 1995 Intl. Computer Music Conf.*, pages 171–174, 1995.

[Goto and Muraoka, 1996] Masataka Goto and Yoichi Muraoka. Beat tracking based on multiple-agent architecture — a real-time beat tracking system for audio signals —. In *Proc. of the Second Intl. Conf. on Multiagent Systems*, pages 103–110, 1996.

[Goto and Muraoka, 1997] Masataka Goto and Yoichi Muraoka. Issues in evaluating beat tracking systems. In *Working Notes of the IJCAI-97 Workshop on Issues in AI and Music*, 1997 (in press).

[Katayose *et al.*, 1989] H. Katayose, H. Kato, M. Imai, and S. Inokuchi. An approach to an artificial music expert. In *Proc. of the 1989 Intl. Computer Music Conf.*, pages 139–146, 1989.

[Large, 1995] Edward W. Large. Beat tracking with a nonlinear oscillator. In *Working Notes of the IJCAI-95 Workshop on Artificial Intelligence and Music*, pages 24–31, 1995.

[Rosenthal, 1992a] David Rosenthal. Emulation of human rhythm perception. *Computer Music Journal*, 16(1):64–76, 1992.

[Rosenthal, 1992b] David Rosenthal. *Machine Rhythm: Computer Emulation of Human Rhythm Perception.* PhD thesis, Massachusetts Institute of Technology, 1992.

[Rowe, 1993] Robert Rowe. *Interactive Music Systems.* The MIT Press, 1993.

[Scheirer, 1996] Eric D. Scheirer. Using bandpass and comb filters to beat-track digital audio. (unpublished), 1996.

[Schloss, 1985] W. Andrew Schloss. *On The Automatic Transcription of Percussive Music — From Acoustic Signal to High-Level Analysis.* PhD thesis, CCRMA, Stanford University, 1985.

[Smith, 1996] Leigh M. Smith. Modelling rhythm perception by continuous time-frequency analysis. In *Proc. of the 1996 Intl. Computer Music Conf.*, pages 392–395, 1996.

[Todd and Lee, 1994] Neil Todd and Chris Lee. An auditory-motor model of beat induction. In *Proc. of the 1994 Intl. Computer Music Conf.*, pages 88–89, 1994.

[Todd, 1994] Neil P. McAngus Todd. The auditory "primal sketch": A multiscale model of rhythmic grouping. *Journal of New Music Research*, 23(1):25–70, 1994.

[Vercoe, 1994] Barry Vercoe. Perceptually-based music pattern recognition and response. In *Proc. of the Third Intl. Conf. for the Perception and Cognition of Music*, pages 59–60, 1994.