A Music Retrieval System based on Query-by-singing for Karaoke Jukebox

Hung-Ming Yu[1], Wei-Ho Tsai[2], and Hsin-Min Wang[1]

[1]Institute of Information Science, Academia Sinica, Taipei, Taiwan
[2] Dept. of Electronic Engineering, National Taipei Univ. of Technology, Taipei, Taiwan
donny@iis.sinica.edu.tw, whtsai@en.ntut.edu.tw, whm@iis.sinica.edu.tw

Abstract. This paper investigates the problem of retrieving Karaoke music by singing. The Karaoke music encompasses two audio channels in each track: one is a mix of vocal and background accompaniment, and the other is composed of accompaniment only. The accompaniments in the two channels often resemble each other, but are not identical. This characteristic is exploited to infer the vocal's background music from the accompaniment-only channel, so that the main melody underlying the vocal signals can be extracted more effectively. To enable an efficient and accurate search for a large music database, we propose a phrase onset detection method based on Bayesian Information Criterion (BIC) for predicting the most likely beginning of a sung query, and adopt a multiple-level multiple-pass Dynamic Time Warping (DTW) for melody similarity comparison. The experiments conducted on a Karaoke database consisting of 1,071 popular songs show the promising results of query-by-singing retrieval for Karaoke music.

Keywords: music information retrieval, Karaoke, query-by-singing

1 Introduction

In recent years, out of the burgeoning amount of digital music circulating on the Internet, there has been an increasing interest in the research for music information retrieval (MIR). Instead of retrieving music with metadata, such as title, performer, and composer, it is desirable to locate music by simply humming or singing a piece of tune to the system. This concept has been extensively studied in various content-based music retrieval research [1-7], collectively called *query-by-humming* or *query-by-singing*.

Depending on the applications, the design of a music retrieval system varies with the type of music data. In general, digital music can be divided into two categories. One is the symbolic music represented by musical scores, e.g., MIDI and Humdrum. The second category relates to those containing acoustic signals recorded from real performances, e.g., CD music and MP3. This type of music is often polyphonic, in which many notes may be played simultaneously, in contrast to monophonic music, in which at most one note is played at any give time. Consequently, extracting the main melody directly from a polyphonic music proves to be a very challenging task [6-10], compared to dealing with the MIDI music, which is easy to acquire the main melody by selecting one of the symbolic tracks.

On the other hand, the development of a query-by-singing MIR system relies on an effective melody similarity comparison. Since most users are not professional singers, a sung query may contain inevitable tempo errors, note dropout errors, note insertion errors, etc. To handle these errors, various approximate matching methods, such as dynamic time warping (DTW) [5][11-12], hidden Markov model [13], and N-gram model [8][10], have been studied, with DTW being the most popular. However, due to the considerable time consumption for DTW, another key issue on designing a query-by-singing MIR system is how to speed up the similarity comparison, so that a large scale music database can be searched efficiently [5][14-15].

In this study, we focus on a sort of music data called *Karaoke*. It stems from Japanese popular entertainment, which provides prerecorded accompaniments to popular songs so that any users can sing live as a professional singer. Karaoke is gaining popularity in East Asia. Nowadays, there is a plenty of Karaoke music in either VCD or DVD format. Each piece of Karaoke track comes with two audio channels: one is a mix of vocal and accompaniment, and the other is composed of accompaniment only. The music in the accompaniment-only channel is usually very similar but not identical to that in the accompanied vocal channel. In this work, methods are proposed to extract vocal’s melody from accompanied Karaoke tracks by reducing the interference from the background accompaniments. In parallel, we apply Bayesian Information Criterion (BIC) [16] to detect the onset time of each phrase in the accompanied vocal channel, which enables the subsequent DTW-based similarity comparison to be performed more efficiently. The proposed system further uses multiple-level multiple-pass DTW to improve the retrieval efficiency and accuracy. We evaluate our approaches on a Karaoke database consisting of 1,071 songs. The experimental results indicate the feasibility of retrieving Karaoke music by singing.

The remainder of this paper is organized as follows. Section 2 introduces the configuration of our Karaoke music retrieval system. Section 3 presents the methods for background music reduction and main melody extraction. Section 4 describes the phrase onset detection. In Section 5, we discuss the similarity comparison module and the schemes to improve the retrieval accuracy and efficiency. Finally, Section 6 presents our experimental results, and Section 7 concludes this study.

2 Overview

Our Karaoke music retrieval system is designed to take as input an audio query sung by a user, and to produce as output the song containing the most similar melody to the sung query. As shown in Fig. 1, it operates in two phases: indexing and searching.

2.1 Indexing

The indexing phase consists of two components: main melody extraction and phrase onset detection. The main melody extraction is concerned with the symbolic description of the melody related to the vocal sung in each song in the collection. Since the channel containing vocal signals also encompasses accompaniments, the melody extracted from raw audio data may not be the tune performed by a singer, but the instruments instead. To reduce the interference from the background accompaniments to main melody extraction, we propose exploiting the signal of accompaniment-only channel to approximate the vocal's background music. The desired vocal signal can thus be distilled by subtracting its background music. Then, the fundamental frequencies of the vocal signals are estimated, whereby converting the waveform representation into a sequence of musical note symbols.

The phrase onset detection aims to locate the expected beginning of a query that users would like to sing to the system. In view of the fact that the length of a popular song is normally several minutes, it is virtually impossible that a user sings a whole song as a query to the system. Further, a user's singing tends to begin with the initial of a sentence of lyrics. For instance, a user may query the system by singing a piece of *The Beatle's* “Yesterday” like this, “Suddenly, I'm not half the man I used to be. There's a shadow hanging over me.” In contrast, a sung query like “I used to be. There's a shadow” or “half the man I used to be.” is believed almost impossible. Therefore, pre-locating the phrase onsets could not only match users' queries better, but also improve the efficiency of the system in the searching phase.

After indexing, the database is composed of the note-based sequences and the labels of phrase onset times for each individual song.

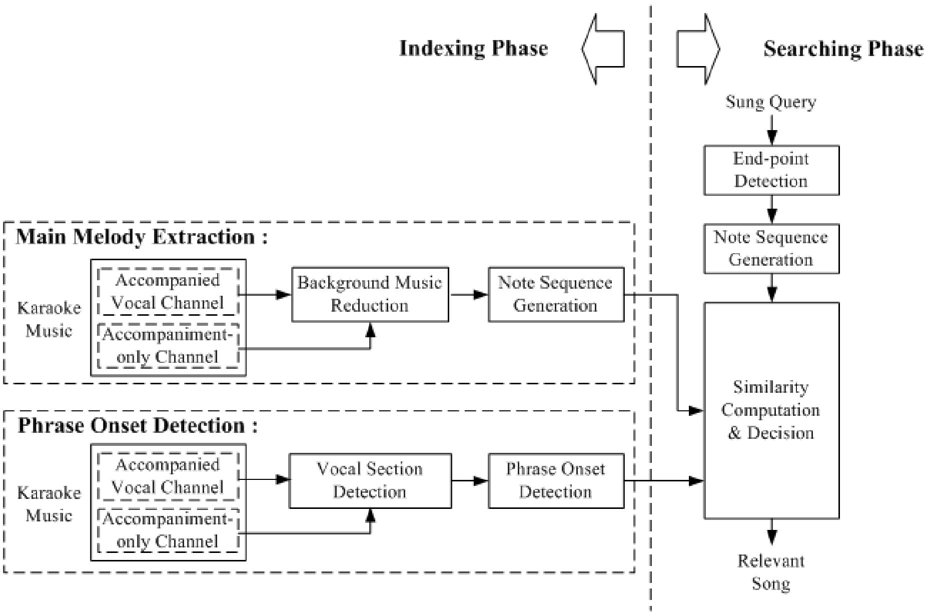

Fig. 1. The proposed Karaoke music retrieval system.

2.2 Searching

In the searching phase, the system determines the song that a user looks for based on what he/she is singing. It is assumed that a user's sung query can be either a complete phrase or a partial phrase but starting from the beginning of a phrase. The system commences with the end-point detection that records the singing voice and marks the salient pauses within the singing waveform. Next, the singing waveform is converted into a sequence of note symbols via the note sequence generation module as used in the indexing phase. Accordingly, the retrieval task is narrowed down to a problem of comparing the similarity between the query's note sequence and each of the documents' note sequences. The song associated with the note sequence most similar to the query's note sequence is regarded as relevant and presented to the user.

3 Main Melody Extraction

3.1 Background Music Reduction

Main melody extraction plays a crucial role in music information retrieval. In contrast to the retrieval of MIDI music, which is easy to acquire the main melody by selecting one of the symbolic tracks, retrieving a polyphonic object in CD or Karaoke format requires to extract the main melody directly from the accompanied singing signals, which proves difficult to handle well simply using the conventional pitch estimation. For this reason, a special effort in this module is put on reducing the background music from the accompanied singing signals.

The indexing phase begins with the extraction of audio data from Karaoke VCD's or DVD's. The data consists of two-channel signals: one is a mix of vocal and accompaniment, and the other is composed of accompaniment only. Fig. 2 shows an example waveform for a Karaoke music piece. It is observed that the accompaniments in the two channels often resemble each other, but are not identical. This motivates us to estimate the pristine vocal signal by inferring its background music from the accompaniment-only channel. To do this, the signals in both channels are first divided into frames by using a non-overlapping sliding window with length of W waveform samples. Let $\mathbf{c}_t=\{c_{t,1}, c_{t,2},\ldots, c_{t,W}\}$ denote the t-th frame of samples in the accompanied vocal channel and $\mathbf{m}_t=\{m_{t,1}, m_{t,2},\ldots, m_{t,W}\}$ denote the t-th frame of the accompaniment-only channel. It can be assumed that $\mathbf{c}_t = \mathbf{s}_t + \mathbf{m}'_t$, where $\mathbf{s}_t =\{s_{t,1}, s_{t,2},\ldots, s_{t,W}\}$ is the pristine vocal signal and $\mathbf{m}'_t =\{m'_{t,1}, m'_{t,2},\ldots, m'_{t,W}\}$ is the underlying background music. Usually, $\mathbf{m}'_t \neq \mathbf{m}_t$, since the accompaniment signals in one channel may be different from another in terms of amplitude, phase, etc., where the phase difference reflects the asynchronism between two channels' accompaniments. As a result, direct subtraction of one channel's signal from another's is of little use for distilling the desired vocal. To handle this problem better, we assume that the accompanied vocal is of the form $\mathbf{c}_t = \mathbf{s}_t + a\mathbf{m}_{t+b}$, where $\mathbf{m}_{t+b}$ is the b-th frame next to $\mathbf{m}_t$, which is most likely corresponding to $\mathbf{m}'_t$, and a is a scaling factor reflecting the amplitude difference between $\mathbf{m}_t$ and $\mathbf{m}'_t$. The optimal b can be found by choosing one of the possible values within a pre-set range, B, that results in the smallest estimation error, i.e.,

$$b^* = \arg\min_{-B \le b \le B} |\,\mathbf{c}_t - a_b^* \mathbf{m}_{t+b}\,|, \tag{1}$$

where a_b^* is the optimal amplitude scaling factor given $\mathbf{m}_{t+b}$. Letting $\partial|\mathbf{c}_t - a_b\mathbf{m}_{t+b}|^2/\partial a_b = 0$, we have a minimum mean-square-error solution of a_b as

$$a_b^* = \frac{\mathbf{c}_t' \mathbf{m}_{t+b}}{\|\,\mathbf{m}_{t+b}\,\|^2}. \tag{2}$$

Accordingly, the underlying vocal signal in frame t can be estimated by $\mathbf{s}_t = \mathbf{c}_t - a_b^* \mathbf{m}_{t+b^*}$. Fig. 2(c) shows the resulting waveform of the accompanied vocal channel after background music reduction. We can see that the accompaniment in the accompanied vocal channel is largely reduced.

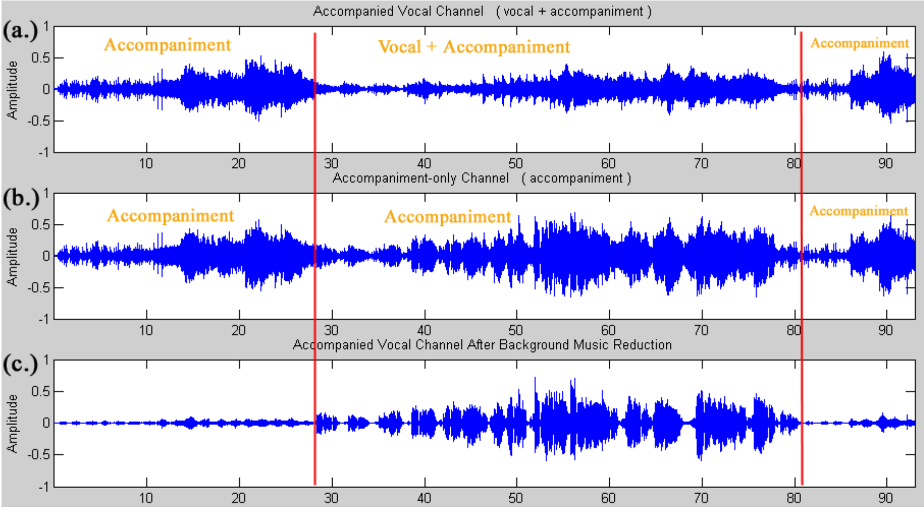

Fig. 2. (a) An accompanied vocal channel. (b) An accompaniment-only channel. (c) The accompanied vocal channel after background music reduction.

3.2 Note Sequence Generation

After reducing the undesired background accompaniments, the next step is to convert each recording from its waveform representation into a sequence of musical notes. Following [7], the converting method begins by computing the short-term Fast Fourier Transform (FFT) of a signal. Let $e_1, e_2, \ldots, e_N$ be the inventory of possible notes performed by a singer, and $x_{t,j}$ denote the signal's energy with respect to FFT index j in frame t, where $1 \le j \le J$. The sung note of a recording in frame t is determined by

$$o_t = \arg\max_{1 \le n \le N} \left(\sum_{c=0}^{C} h^c \, y_{t,n+12c} \right), \tag{3}$$

where C is a pre-set number of harmonics concerned, h is a positive value less than 1 for discounting higher harmonics, and $y_{t,n}$ is the signal's energy on note e_n in frame t, estimated by

$$y_{t,n} = \max_{\forall j, U(j)=e_n} x_{t,j} \,, \tag{4}$$

and

$$U(j) = \left\lfloor 12 \cdot \log_2 \left(\frac{F(j)}{440} \right) + 69.5 \right\rfloor, \tag{5}$$

where $\lfloor \; \rfloor$ is a floor operator, $F(j)$ is the corresponding frequency of FFT index j, and $U(\cdot)$ represents a conversion between the FFT indices and the MIDI note numbers.

3.3 Note Sequence Smoothing

The resulting note sequence may be refined by identifying and correcting the abnormal notes arising from the residual background music. The abnormality in a note sequence can be divided into two types of errors: short-term error and long-term error. The short-term error is concerned with the rapid changes, e.g., jitters between adjacent frames. This type of error could be amended by using the median filtering, which replaces each note with the local median of notes of its neighboring frames. On the other hand, the long-term error is concerned with a succession of the estimated notes not produced by a singer. These successive wrong notes are very likely several octaves above or below the true sung notes, which could result in the range of the estimated notes within a sequence being wider than that of the true sung note sequence. As reported in [7], the sung notes within a verse or chorus section usually vary no more than 22 semitones. Therefore, we may adjust the suspect notes by shifting them several octaves up or down, so that the range of the notes within an adjusted sequence can conform to the normal range. Specifically, let **o** = $\{o_1, o_2, \ldots, o_T\}$ denote a note sequence estimated using Eq. (3). An adjusted note sequence **o′** = $\{o'_1, o'_2, \ldots, o'_T\}$ is obtained by

$$o'_t = \begin{cases} o_t \,, & \text{if } |o_t - \bar{o}| \le (R/2) \\ o_t - 12 \times \left\lfloor \dfrac{o_t - \bar{o} + R/2}{12} \right\rfloor, & \text{if } o_t - \bar{o} > (R/2) \\ o_t - 12 \times \left\lfloor \dfrac{o_t - \bar{o} - R/2}{12} \right\rfloor, & \text{if } o_t - \bar{o} < (-R/2) \end{cases} \,, \tag{6}$$

where R is the normal varying range of the sung notes in a sequence, say 22, and $\bar{o}$ is the mean note computed by averaging all the notes in **o**. In Eq. (6), a note o_t is considered as a wrong note and needs to be adjusted if it is too far away from $\bar{o}$, i.e., $|o_t - \bar{o}| > R/2$. The adjustment is done by shifting the wrong note $\lfloor (o_t - \bar{o} + R/2)/12 \rfloor$ or $\lfloor (o_t - \bar{o} - R/2)/12 \rfloor$ octaves.

4 Phrase Onset Detection

In general, the structure of a popular song involves five sections: *intro*, *verse*, *chorus*, *bridge*, and *outro*. The verse and chorus contain the vocals sung by the lead singer, while the intro, bridge, and outro are largely accompaniments. This makes it natural that a verse or chorus is the favorite that people go away humming when they hear a good song, and hence is often the query that a user may hum or sing to a music retrieval system.

Since a user's singing query tends to begin with the initial of a sentence in lyrics, we can consider a song's lyrics as a collection of phrases. The beginning of each phrase is likely the starting point of a user's query. Therefore, if the onset time of each phrase can be detected before the DTW comparison is performed, it is expected that both the search efficiency and the retrieval accuracy can be improved.

Our strategy for detecting the phrase onsets is to locate the boundaries that the signal in the accompanied vocal channel is changed from accompaniment-only to a mix of vocal and accompaniment. As mentioned earlier, the accompaniment of one channel in a Karaoke track often resembles that of the other channel. Thus, if no vocal is performed in a certain passage, the difference of signal spectrum between the two channels is tiny. In contrast, if a certain passage contains vocal signals, there must be a significant difference between the two channels during this passage. We therefore could examine the difference of signal spectrum between the two channels, thereby locating the phrase onsets. In our system, the Bayesian Information Criterion (BIC) [16] is applied to characterize the level of spectrum difference.

4.1 The Bayesian Information Criterion (BIC)

The BIC is a model selection criterion which assigns a value to a stochastic model based on how well the model fits a data set, and how simple the model is. Given a data set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_N\} \subset R^d$ and a model set $\mathbf{H} = \{H_1, H_2,\ldots, H_K\}$, the BIC value for model H_k is defined as:

$$BIC(H_k) = \log p(\mathbf{X}|H_k) - 0.5\,\lambda\,\#(H_k)\log N, \tag{7}$$

where λ is a penalty factor, $p(\mathbf{X}|H_k)$ is the likelihood that H_k fits $\mathbf{X}$, and $\#(H_k)$ is the number of free parameters in H_k. The selection criterion favors the model having the largest value of BIC.

Assume that we have two audio segments represented by feature vectors, $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_N\}$ and $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2,\ldots, \mathbf{y}_N\}$, respectively. If it is desired to determine whether $\mathbf{X}$ and $\mathbf{Y}$ belong to the same acoustic class, then we have two hypotheses to consider: one is "yes", and the other is "no". Provided that hypotheses "yes" and "no" are characterized by certain stochastic models H_1 and H_2, respectively, our aim will be to judge which among H_1 and H_2 is better. For this purpose, we represent H_1 by a single Gaussian distribution $\mathcal{N}(\mu,\Sigma)$, where μ and Σ are the sample mean and covariance estimated using vectors $\{\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_N, \mathbf{y}_1, \mathbf{y}_2,\ldots, \mathbf{y}_N\}$, and represent H_2 by two Gaussian distributions $\mathcal{N}(\mu_x,\Sigma_x)$ and $\mathcal{N}(\mu_y,\Sigma_y)$, where the sample mean μ_x and covariance Σ_x are estimated using vectors {$\mathbf{x}_1$, $\mathbf{x}_2$,…, $\mathbf{x}_N$}, and the sample mean μ_y and covariance Σ_y are estimated using vectors {$\mathbf{y}_1$, $\mathbf{y}_2$,…, $\mathbf{y}_N$}. Then, the problem of judging which model is better can be solved by computing a difference value of BIC between $BIC(H_1)$ and $BIC(H_2)$, i.e.,

$$\Delta BIC = BIC\,(H_2) - BIC(H_1). \tag{8}$$

Obviously, the larger the value of ΔBIC, the more likely segments **X** and **Y** are from different acoustic classes, and vice versa. We can therefore set a threshold of ΔBIC to determine if two audio segments belong to the same acoustic class.

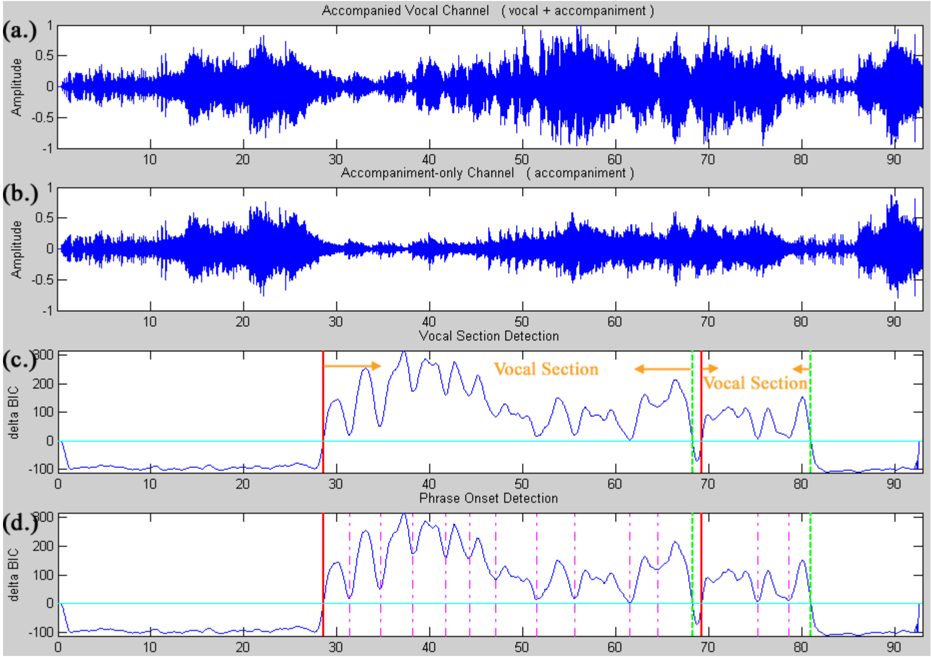

Fig. 3. An example of phrase onset detection. (a) The waveform of an accompanied vocal channel. (b) The waveform of an accompaniment-only channel. (c) The ΔBIC curve and the detected vocal sections. (d) The results of phrase onset detection.

4.2 Phrase Onset Detection via BIC

In applying the concept of BIC to the phrase onset detection problem, our goal is to judge whether the signals in the two channels belong to the same acoustic class during a certain time interval, where one class represents accompaniment only, and the other represents vocal over accompaniment. Thus, by considering **X** and **Y** as two concurrent channels' signals, ΔBIC can be computed along the entire recording, and then plotted as a curve. Fig. 3 shows an example Karaoke music clip underwent our phrase onset detection. Figs. 3(a) and 3(b) are the waveforms in the accompanied vocal channel and the accompaniment-only channel, respectively. The phrase onset detection begins by chopping the waveform in each of the channels into non-overlapping frames of 20ms. Each frame is represented as 12 Mel-scale Frequency Cepstral Coefficients (MFCCs). Then, the ΔBIC value between each pair of one-second segments in the two channels is computed frame by frame, thereby forming a ΔBIC curve over time, as shown in Fig. 3(c). The positive value of ΔBIC indicates that the frame of the accompanied vocal channel contains vocals but the concurrent frame in the accompaniment-only channel does not. In contrast, the negative value of ΔBIC indicates that both frames contain accompaniments only. Therefore, the intervals where positive values of ΔBIC appear are identified as vocal sections.

In the example shown in Fig. 3(c), two vocal sections are identified, each surrounded by a solid line and a dashed line. Within each vocal section, the local minimums on the ΔBIC curve can be further located as phrase onsets because the frame corresponds to rest or breathing between phrases usually yields a small ΔBIC value. Fig. 3 (d) depicts the detected phrase onsets in this way. Note that the trade-off between the retrieval accuracy and retrieval efficiency are highly dependent on the number of detected phrase onsets. In general, the larger the number of the likely phrase onsets is detected, the higher the retrieval accuracy can be achieved. However, increasing the number of the candidate phrase onsets often decreases the retrieval efficiency drastically.

5 Melody Similarity Comparison

Given a user's sung query and a set of music documents, each of which is represented by a note sequence, the task here is to find a music document whose partial note sequence is most similar to the query's note sequence.

5.1 Dynamic Time Warping Framework

Let $\mathbf{q} = \{q_1, q_2, \ldots, q_T\}$ and $\mathbf{u} = \{u_1, u_2, \ldots, u_L\}$ be the note sequences extracted from a user's query and a particular music segment to be compared, respectively. As the lengths of **q** and **u** are usually different, computing the distance between **q** and **u** directly is infeasible. To handle this problem, the most prevalent way is to find the temporal mapping between **q** and **u** by Dynamic Time Warping (DTW). Mathematically, DTW constructs a $T \times L$ distance matrix $\mathbf{D} = [D(t, \ell)]_{T \times L}$, where $D(t, \ell)$ is the distance between note sequences $\{q_1, q_2, \ldots, q_t\}$ and $\{u_1, u_2, \ldots, u_\ell\}$, computed using:

$$D(t,\ell) = \min \begin{cases} D(t-2,\ell-1) + 2 \times d(t,\ell) \\ D(t-1,\ell-1) + d(t,\ell) - \varepsilon \\ D(t-1,\ell-2) + d(t,\ell) \end{cases} , \tag{9}$$

and

$$d(t, \ell) = | q_t - u_\ell | , \tag{10}$$

where ε is a small constant that favors the mapping between notes q_t and u_ℓ, given the distance between note sequences $\{q_1, q_2,\ldots,q_{t-1}\}$ and $\{u_1, u_2,\ldots, u_{\ell-1}\}$.

To compensate for the inevitable errors arising from the phrase onset detection, we assume that the true phrase onset time associated with the automatically detected phrase onset time t_{os} is within $[t_{os}$-$r/2$, $t_{os}+r/2]$, where r is a predefined tolerance. Though the DTW recursion in Eq. (9) indicates that the best path exists only when the length of the music note sequence is within half to twice length of the query note sequence (i.e., between $T/2$ and $2T$), we prefer to limit the best mapping length of **u** to **q** to be between $T/2$ and kT, where k is a value between 1 and 2, so that the mapping can be more precisely. In other words, the tempo of the query is allowed to be $1/k$ times to twice the tempo of the target music document. Therefore, in the implementation, we set the new phrase onset time t'_{os} as t_{os}-$r/2$, clone the subsequence of notes of a music document starting from t'_{os} with a length L of $kT+r$ to **u**, and define the boundary conditions for the DTW recursion as,

$$\begin{cases} D(1,1) = d(1,1) \\ D(t,1) = \infty,\ 2 \le t \le T \\ D(t,2) = \infty,\ 4 \le t \le T \\ D(1,\ell) = \begin{cases} d(1,\ell), & 1 \le \ell \le r \\ \infty, & r < l \le L \end{cases} \\ D(2,\ell) = \begin{cases} d(1,\ell-1) + d(2,\ell), & 2 \le \ell \le r+1 \\ \infty, & r+1 < l \le L \end{cases} \\ D(3,2) = d(1,1) + 2 \times d(3,2) \end{cases} . \tag{11}$$

After the distance matrix **D** is constructed, the similarity between **q** and **u** can be evaluated by

$$S(\mathbf{q},\mathbf{u}) = \min_{T/2 \le \ell \le L} D(T,\ell). \tag{12}$$

5.2 Multiple-Pass DTW to Improve Retrieval Accuracy

Since a query may be sung in a different key or register than the target music document, i.e., the so-called *transposition,* the resulting note sequences of the query and the document could be rather different. This problem can be alleviated by shifting the query's note sequence upward or downward several semitones, so that the mean of the shifted query's note sequence can equal that of the document to be compared. In addition, considering that a user's transposition or key change may occur in a partial sung query, we further perform multiple DTW similarity comparisons by shifting a query sequence upward or downward v semitones. The distance $S(\mathbf{q},\mathbf{u})$ is then defined as,

$$S(\mathbf{q},\mathbf{u}) = \min_{-V \le v \le V} S(\mathbf{q}^{(v)},\mathbf{u}), \tag{13}$$

where $\mathbf{q}^{(v)}$ denotes the query sequence obtained by shifting **q** upward or downward v semitones. As reported in [7], the retrieval performance improves as the value of V increases. However, increasing the value of V substantially increases computational costs, because the similarity comparison requires two extra DTW operations whenever the value of V is increased by one. Thus, an economic value of V = 1, i.e., three-pass DTW, is adopted in this work.

In addition to the difference of key and tempo existing between queries and documents, another problem to be addressed is the existence of voiceless regions in a sung query. The voiceless regions, which may arise from the rest, pause, etc., result in some notes being tagged with “0” in the query note sequence. However, the corresponding non-vocal regions in the document are usually not tagged with “0”, because there are accompaniments in those regions. Although the voiceless regions in a sung query can be detected by simply using the energy information, the accurate detection of non-vocal regions in a music document remains a very difficult problem. To sidestep this problem, we modify the computation of $d(t,\ell)$ in Eq. (10) to

$$d(t,\ell)=\begin{cases}|q_t-u_\ell|, & q_t\neq 0\\ \varphi, & q_t=0\end{cases}, \tag{14}$$

where φ is a small constant. Implicit in Eq. (14) is equivalent to bypassing the voiceless regions of a query.

5.3 Multiple-Level DTW to Improve Retrieval Efficiency

As shown in Fig. 4(a), the computational complexity in terms of the number of the necessary distance computation $D(\cdot)$ for constructing a $T\times L$ table is

$$\begin{aligned} Complexity &= T\times L-\frac{T\times T/2}{2}-\frac{L\times L/2}{2}\\ &= TL-\frac{(T^2+L^2)}{4}. \end{aligned} \tag{15}$$

As mentioned earlier, since L is usually set to be kT, where $1/2\le k\le 2$, the computational complexity can be rewritten as

$$Complexity=\frac{(4k-k^2-1)T^2}{4}. \tag{16}$$

Although the DTW recursion allows a document sequence within half to twice the length of the query sequence, empirical evidence shows that the document length can be simply limited to 1.2 times the length of a query, i.e., k = 1.2, without significantly degrading the retrieval performance. Hence, by substituting k = 1.2 into Eq. (16), the computational complexity is $0.59T^2$. As shown in Fig. 4(b), the introduction of a phrase onset tolerance in the DTW will increase the computational complexity by rT. If r is small compared to T, the increase in complexity is negligible. If r = 0.59T, the computational complexity is twice that of the typical DTW.

Since the complexity is $O(T^2)$, the most promising way to speed up the searching process is to reduce the value of T. Motivated by Keogh and Pazzani’s Piecewise Aggregate Approximation (PAA) [14], we propose a dimensionality reduction technique, called Multi-Level Data Abstraction (MLDA). Unlike PAA, which divides a time series into equal-length frames, and then calculates the mean value of the data falling within a frame, MLDA aims to prune the less likely music clips in a step-by-step manner. In MLDA, the compression rate of data is power of two at each level. For example, if we go through the note sequence and pick one note every two notes, the note sequence is reduced to half length, while the computation is reduced to quarter complexity. We can first use the reduced note sequences to prune less likely music documents. If the original computational complexity is *CC* and the pruning rate is R_{pr}, then the total computational complexity of the two-level DTW is $[1/4+(1-R_{pr})]CC$. If a three-level DTW is applied, the complexity is reduced to $[(1/4)^2+(1/4)(1-R_{pr})+(1-R_{pr})^2]CC$. In this way, when the pruning rate R_{pr} is set at 0.75, the complexity of the two-level DTW and three-level DTW is 1/2 and 3/16 that of the single-level DTW, respectively.

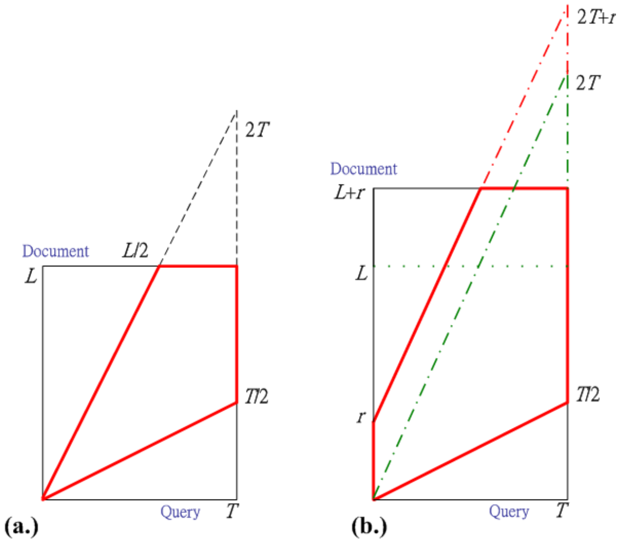

Fig. 4. (a) The search space for the typical DTW, in which the starting frame of a query and that of the target phrase are aligned; (b) The search space for the DTW we apply, in which the starting frame of a query can be mapped to any one of the first *r* frames of the target phrase.

6 Experiments

6.1 Music database

The music database used in this study consisted of 1,071 songs extracted from Karaoke VCDs. The extracted waveform signals were down-sampled from the sampling rate of 44.1 kHz to 22.05 kHz. The database was divided into two subsets. The first subset consisted of 95 songs, denoted as DB-1. The second subset consisted of 976 songs, denoted as DB-2. To compare the system performances achieved with automatic and manual detection of phrase onset, we manually labeled the phrase onsets of the songs in DB-1. There were 775 phrase onsets marked.

We collected 90 queries from 9 male and 4 female users. The duration of each query ranged from 15 seconds to 45 seconds, but only the first 8-second portion was input to the system. The performance was measured on the basis of *song accuracy* defined as,

$$Song\ accuracy(\%) = \frac{\#\, queries\ receiving\ the\ correct\ songs}{\#\, queries} \times 100\%. \qquad (17)$$

In addition, considering a more user-friendly scenario where a list of Top-N ranked documents can be provided for user’s choices, we also computed the Top-N accuracy defined as the percentage of the queries whose target songs are among Top-N. The overall system performance was evaluated on both DB-1 and DB-2.

6.2 Experimental results

The first experiment was conducted to evaluate the effectiveness of the background music reduction using DB-1. Here, the phrase onsets were labeled manually. The retrieval performance is given in Table 1. It is clear that the background music reduction improves the retrieval performance.

Table 1. Retrieval performance obtained with and without background music reduction.

	Song accuracy (%)		
	Top 1	Top 3	Top 10
without background music reduction	56.67	67.78	76.67
with background music reduction	72.22	76.67	83.33

The second experiment was conducted to compare the retrieval performance obtained with the manual phrase onset labeling and the automatic phrase onset detection. The automatic phrase onset detection approach marked 2,719 phrase onsets in the 95 songs in DB-1, which is about 3.5 times that of hand-labeled phrase onsets. The retrieval performance is given in Table 2. We observe that the retrieval accuracy only decreases slightly when the way to mark the phrase onsets was changed from manual to automatic.

Table 2. Retrieval performance based on manual and automatic phrase onset detections.

	Song accuracy (%)		
	Top 1	Top 3	Top 10
Manual phrase onset labeling	72.22	76.67	83.33
Automatic phrase onset detection	70.00	74.44	77.78

Lastly, we evaluated the performance of the Karaoke retrieval system using both DB-1 and DB-2, with the same 90 queries. The system automatically marked 27,397 phrase onsets in the 1,071 songs in DB-1 and DB-2. The retrieval performance is given in Table 3. We observe that the Top-1 accuracy drops from 70.00% to 51.11% as the database expands from 95 songs to 1,071 songs, while the Top-10 accuracy only slightly drops from 77.78% to 70.00%. To speed up the searching, the four-level DTW, with various pruning rates, R_{pr}, was implemented. We observe from Table 3 that the searching time for the multiple-level DTW can be greatly reduced at a small cost of retrieval accuracy degradation.

Table 3. Retrieval performance evaluated using both DB-1 and DB-2.

	Song accuracy (%)		
	Top 1	Top 3	Top 10
Single-level DTW, complexity *CC*	51.11	57.78	70.00
Four-level DTW, R_{pr} = 0.6 (complexity reduced to 0.145 *CC*)	51.11	57.78	67.78
Four-level DTW, R_{pr} = 0.75 (complexity reduced to 0.063 *CC*)	50.00	55.56	65.56
Four-level DTW, R_{pr} = 0.9 (complexity reduced to 0.025 *CC*)	46.67	54.44	63.33

7 Conclusions

We have presented a Karaoke music retrieval system that allows users to locate their desired music by singing to the system. Since the vocals and various concurrent accompaniments are mixed together in an accompanied vocal channel, we proposed a method to reduce the accompaniments in the accompanied vocal channel so that the accuracy of main melody extraction could be improved. In addition, we applied Bayesian Information Criterion (BIC) to detect the onset time of a musical phrase which reflects the most likely beginning of a sung query. The phrase onset detection, in conjunction with multiple-level multiple-pass DTW matching for similarity comparison, enables an efficient and effective search for a large music database. The experiments conducted on a music database consisting of 1,071 songs confirmed the feasibility of our retrieval system.

Acknowledgments. This work was supported in part by the National Science Council, Taiwan, under Grants: NSC94-2422-H-001-007 and NSC95-2422-H-001-008.

References

1. Ghias, A., H. Logan, D. Chamberlin, and B. C. Smith, “Query by Humming: Musical Information Retrieval in an Audio Database,” *Proc. ACM International Conference on Multimedia*, 1995.
2. Kosugi, N., Y. Nishihara, T. Sakata, M. Yamamuro, and K. Kushima, “Music Retrieval by Humming,” *Proc. IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, 1999.
3. Kosugi, N., Y. Nishihara, T. Sakata, M. Yamamuro, and K. Kushima, “A Practical Query-By-Humming System for a Large Music Database,” *Proc. ACM International Conference on Multimedia*, 2000.
4. Nishimura, T., H. Hashiguchi, J. Takita, J. X. Zhang, M. Goto, and R. Oka, “Music Signal Spotting Retrieval by a Humming Query Using Start Frame Feature Dependent Continuous Dynamic Programming,” *Proc. International Symposium on Music Information Retrieval,* 2001.
5. Jang, J. S. Roger, and H. R. Lee, “Hierarchical Filtering Method for Content-based Music Retrieval via Acoustic Input,” *Proc. ACM International Conference on Multimedia,* 2001.
6. Song, J., S. Y. Bae, and K. Yoon, “Mid-Level Music Melody Representation of Polyphonic Audio for Query-by-Humming System,” *Proc. International Conference on Music Information Retrieval*, 2002.
7. Yu, H. M., W. H. Tsai, and H. M. Wang, ”A Query-by-singing Technique for Retrieving Polyphonic Objects of Popular Music,” *Proc. Asian Information Retrieval Symposium*, 2005
8. Doraisamy, S. and S. M. Ruger, “An Approach Towards a Polyphonic Music Retrieval System,” *Proc. International Symposium on Music Information Retrieval,* 2001.
9. Goto, M., “A Predominant-F0 Estimation Method for CD Recordings: MAP Estimation Using EM Algorithm for Adaptive Tone Models,” *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing,* 2001.
10. Doraisamy, S. and S. Rüger, “Robust Polyphonic Music Retrieval with *N*-grams,” *Journal of Intelligent Information Systems*, 21(1), pp. 53–70, 2003.
11. Liu, C. C., A. J. L. Hsu, and A. L. P. Chen, “An Approximate String Matching Algorithm for Content-Based Music Data Retrieval,” *Proc. IEEE International Conference on Multimedia Computing and Systems,* 1999.
12. Mo, J. S., C. H. Han, and Y. S. Kim, “A Melody-Based Similarity Computation Algorithm for Musical Information,” *Proc. Workshop on Knowledge and Data Engineering Exchange*, 1999.
13. Shifrin, J. and W. Burmingham, “Effectiveness of HMM-based Retrieval on Large Databases,” *Proc. International Conference on Music Information Retrieval*, 2003.
14. Keogh, E. and M. Pazzani, “Scaling up Dynamic Time Warping for Datamining Applications,” *Proc. ACM SIGKDD*, 2000.
15. Salvador, S. and P. Chan, “FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space,” *Proc. KDD Workshop on Mining Temporal and Sequential Data*, 2004
16. Schwarz, G., “Estimation the Dimension of a Model,” *The Annals of Statistics*, 6, pp. 461-364, 1978.