A Query-by-Singing System for Retrieving Karaoke Music

Hung-Ming Yu, Wei-Ho Tsai, *Member, IEEE*, and Hsin-Min Wang, *Senior Member, IEEE*

Abstract—**This paper investigates the problem of retrieving karaoke music using query-by-singing techniques. Unlike regular CD music, where the stereo sound involves two audio channels that usually sound the same, karaoke music encompasses two distinct channels in each track: one is a mixture of the lead vocals and background accompaniment, and the other consists of accompaniment only. Although the two audio channels are distinct, the accompaniments in the two channels often resemble each other. We exploit this characteristic to (i) infer the background accompaniment for the lead vocals from the accompaniment-only channel, so that the main melody underlying the lead vocals can be extracted more effectively; and (ii) detect phrase onsets based on the Bayesian Information Criterion (BIC) to predict the onset points of a song where a user's sung query may begin, so that the similarity between the melodies of the query and the song can be examined more efficiently. To further refine extraction of the main melody, we propose correcting potential errors in the estimated sung notes by exploiting a composition characteristic of popular songs whereby the sung notes within a verse or chorus section usually vary no more than two octaves. In addition, to facilitate an efficient and accurate search of a large music database, we employ multiple-pass Dynamic Time Warping (DTW) combined with multiple-level data abstraction (MLDA) to compare the similarities of melodies. The results of experiments conducted on a karaoke database comprised of 1,071 popular songs demonstrate the feasibility of query-by-singing retrieval for karaoke music.**

Index Terms—**Bayesian Information Criterion, Dynamic Time Warping, karaoke, music information retrieval, query-by-singing**

I. Introduction

In recent years, the rapid advances in Internet connectivity and signal processing technologies have led to a dramatic and unprecedented increase in the availability of music material. On the one hand, the increase has further expanded the human appetite for music, while, on the other, it has provided a major impetus to the development of various new technologies. Ironically, the burgeoning availability of music material has made it increasingly difficult to locate desired items from the innumerable options. Thus, techniques that could help users acquire the music they want quickly are being extensively developed to keep pace with the proliferation of music material. Among such techniques, retrieving music material via audio queries is of particular interest in the domain of accessing popular music. The root concept, query-by-humming/singing or query-by-example, continues to motivate the development of many promising solutions for retrieving music beyond the conventional text-processing paradigm. Examples of such solutions enable users to retrieve a song by humming a catchy tune without having to name the song [1-17,30-36], or help users find songs performed by their favorite singers [18], in their favorite genre [19] or mood [20] by playing an excerpt of the desired music as a query.

Depending on the application, the design of a music information retrieval (MIR) system varies with the type of musical data. In general, digital music can be divided into two categories: symbolic music represented by musical scores, e.g., MIDI and Humdrum; and music containing acoustic signals recorded from real performances, e.g., CD music and MP3. The second type of music is often polyphonic, so many notes may be played simultaneously. By contrast, in monophonic music, at most one note is played at any given time. As a consequence, extracting the main melody directly from a piece of polyphonic music has proved to be a very challenging task [21,22], compared to dealing with MIDI music, where the main melody can be acquired easily by selecting one of the symbolic tracks. Because of this difficulty, the vast majority of current query-by-humming/singing systems [1-7,15,31,35] work within the monophonic domain, which converts a monophonic audio query into a symbolic format to match a monophonic symbolic collection. Some studies [16,17] focus on locating the major themes from a piece of symbolic polyphonic music in which the note information is given a priori. However, very few systems operate in the mode of monophonic audio queries applied to a polyphonic audio collection [8,9,13,30,36], or entirely polyphonic audio queries applied to a polyphonic audio collection [10-12]. Currently, query-by-humming/singing techniques for polyphonic music are in the initial stages of development and can only handle small music databases; thus, they are far from ready for practical use.

In this study, we focus on *karaoke* music, which also belongs to the polyphonic audio category. However, its dual-channel characteristic, as we explain later, makes the process of main melody extraction somewhat easier, compared to dealing with regular CD music. The term karaoke stems from a popular type of Japanese entertainment, which provides pre-recorded

Manuscript received January 9, 2008; revised May 27, 2008. This work was supported in part by National Digital Archives Program (NDAP, Taiwan) sponsored by the National Science Council, Taiwan, under Grants NSC94-2422-H-001-007, NSC95-2422-H-001-008, and NSC95-2422-H-001-031. The associate editor coordinating the review of this paper and approving it for publication was Prof. Alan Hanjalic.

H. M. Yu is with the Institute of Information Science, Academia Sinica, Taipei, Taiwan (e-mail: donny@iis.sinica.edu.tw).

W. H. Tsai is with the Department of Electronic Engineering & Graduate Institute of Computer and Communication Engineering, National Taipei University of Technology, Taipei, Taiwan (e-mail: whtsai@ntut.edu.tw).

H. M. Wang is with the Institute of Information Science, Academia Sinica, Taipei, Taiwan (e-mail: whm@iis.sinica.edu.tw).

accompaniments to popular songs so that any user can sing live like a professional singer. This entertainment medium is now becoming increasingly popular in many places, especially in South East Asia, and is also driving the development of many home devices and online services. To sing karaoke, users have to specify the song they are going to perform, usually by inputting the song title into the karaoke machine or system. As users often cannot recall the title of a particular song, retrieving music based on query-by-humming/singing is in great demand for karaoke applications. However, research on this topic [30] has not yet been explored seriously.

Unlike regular CD music, where the stereo sound involves two audio channels that usually sound the same, each audio track on a karaoke VCD comprises two distinct channels: one is a mixture of the lead vocals and background accompaniment, and the other consists of accompaniment only. Although the two audio channels are distinct, the music in the accompaniment-only channel usually sounds similar to the background accompaniment in the accompanied vocal channel. This characteristic can be exploited to infer the background accompaniment for the lead vocals from the accompaniment-only channel, so that the main melody underlying the lead vocals can be extracted more reliably. On the other hand, as songs usually become popular because their tunes are easy to hum and memorize, the notes sung in such songs usually vary far less than the range of notes that humans are able to sing. It has been observed that, in popular songs, the sung notes in a verse or chorus section usually vary no more than two octaves [13]. We can exploit this composition characteristic to correct potential errors in the sung notes estimated by signal processing approaches, and thereby enhance extraction of the main melody. The feasibility of main melody extraction for karaoke music allows us to scale up the MIR system to work with a large polyphonic database.

In addition to main melody extraction, another key aspect of the development of a query-by-humming/singing MIR system is how to compare the similarity of the melodies in users' queries with those of the actual music in the database effectively. Since most users are not professional singers, a hummed/sung query may contain tempo errors, note dropout errors, and note insertion errors, etc. Even professional singers do not necessarily present error-free queries to MIR systems [31-33,37], because they may not always recall the theme perfectly. To handle such errors, various approximate matching methods, such as dynamic time warping (DTW) [5,6,13,23,24], the hidden Markov model [15,31], and the N-gram model [11,14], have been developed, with DTW being the most popular. However, due to the considerable computational time required for DTW, several speed-up methods have been proposed [5,23,24,31], so that a large-scale music database can be searched more efficiently. In contrast to existing methods, our strategy incorporates the audio characteristics of karaoke music into the system design. Specifically, the proposed method detects the phrase onsets in a karaoke song that correspond to the points of the song where a user's sung query may begin. This is done by using the Bayesian Information Criterion (BIC) [25] to examine the difference between the signal in the accompaniment-only channel and that in the accompanied vocal channel. By locating the phrase onsets, the total search space of DTW can be substantially reduced. We also employ a multiple-level multiple-pass DTW to improve the efficiency and accuracy of melody similarity comparison. The results of experiments conducted on a karaoke database comprised of 1,071 songs demonstrate the feasibility of retrieving karaoke music by humming/singing.

The remainder of this paper is organized as follows. The general characteristics of popular music are discussed in Section II. Section III describes the configuration of our karaoke music retrieval system. Section IV presents the proposed methods for background accompaniment reduction and main melody extraction. Section V describes the phrase onset detection process. In Section VI, we discuss the similarity comparison module and schemes that improve retrieval accuracy and efficiency. Section VII presents our experiment results, and Section VIII concludes the study.

II. GENERAL CHARACTERISTICS OF POPULAR MUSIC

Analysis of the characteristics of the data to be processed is an essential step in designing a reliable information retrieval system. It is recognized that, usually, popular music is popular because the melodies are easy to sing and memorize; however, extracting the melodies automatically is difficult. In this section, we highlight some characteristics of popular music that could be exploited to advance the realization of a query-by-singing retrieval system for karaoke music.

A. The Structure of A Popular Song

In general, the structure of a popular song can be divided into five sections:

- *intro*: usually the first 10-20 seconds of the song, and simply an instrumental statement of the subsequent sections;
- *verse*: typically comprises the main content of the story represented in the song's lyrics;
- *chorus*: often the heart of the song, where the most recognizable melody is present and repeated;
- *bridge*: located roughly two-thirds of the way through the song, where a key change, tempo change, or new lyric is usually introduced to create a sensation of something different coming next; and
- *outro*: often a fading version of the chorus or an instrumental restatement of some earlier sections to bring the song to a conclusion.

Except for the intro and outro, the other sections may be repeated several times with different lyrics and melodies. For example, the song "*Day Tripper*" by The Beatles can be summarized as the structure "intro-verse-chorus-verse-chorus-bridge-verse-chorus-outro" [38]. In essence, the intro, bridge, and outro are usually accompaniments only, while the verse and the chorus contain vocals sung by the lead singer. Because of containing vocals, the verse and chorus are usually the parts that people would like to sing or hum after hearing a good song; hence, they often form the query that a user may hum or sing to a music retrieval system.

B. Range of Notes Sung in a Popular Song

The notes of human singing may vary from F2 (87.3 Hz) to

B5 (987.8Hz), corresponding to a range of 43 *semitones*[1]. However, it is recognized that the sung notes in a music recording usually vary far less than this range, while the range of sung notes in a verse or a chorus section can be even narrower. Fig. 1 shows a sample fragment of a song performed with MIDI[2]. It is clear that the range of notes in the verse can be distinguished from that in the chorus because the sung notes in a section do not cover the whole range of possible notes, but are distributed in a narrower range instead. Our informal survey using 50 pop songs showed that the range of sung notes in a whole song and in a verse or chorus section is approximately 25 and 22 semitones, respectively. Fig. 2 details the statistical results. This information is useful for discarding virtually impossible notes when one tries to transcribe the sung notes in a song.

C. *Harmonicity between Singing and Accompaniment*

In addition to singing, popular music contains background accompaniment during most, if not all, vocal passages. Various signals from different sources are mixed together in a single track. This makes it more difficult to design a system for retrieving real performance music than to design a system for retrieving MIDI music, since the desired information, usually in the solo voice, is inextricably intertwined with the background signals. Furthermore, the background accompaniment often contains notes several octaves above or below those of the singing so that the mixture of the lead vocals and the background accompaniment is harmonic. However, such harmonicity between the singing voice and the accompaniment makes the vocal melody notoriously difficult to extract. Fig. 3 shows an example of a song performed with MIDI. We observe from the notes indicated by arrows that a large proportion of sung notes are accompanied by notes one or two octaves above them. Nevertheless, viewed from another angle, harmonicity can be exploited as a constraint in the determination of sung notes. A method based on this idea to improve main melody extraction is discussed in detail in Section IV.

III. SYSTEM OVERVIEW

Our karaoke music retrieval system is designed to take an audio query sung by a user as input, and produce the song containing the most similar melody to the sung query as output. As shown in Fig. 4, the system has two phases: indexing and searching.

A. Indexing

The objective of indexing is to generate searchable entries for each of the songs in the collection. It is divided into two processes: main melody extraction and phrase onset detection.

Main melody extraction is concerned with the symbolic description of the melody related to the lead vocals in each song in the collection. Since the channel containing the lead vocals also contains the background accompaniment, the melody extracted from raw audio data may be the tune played on the instruments, instead of the one performed by the lead singer. To reduce interference from the background accompaniment during main melody extraction, we exploit the signal of the accompaniment-only channel to approximate the background accompaniment for the lead vocals. The desired lead vocals can thus be distilled by removing the background accompaniment. Then, the fundamental frequencies of the distilled lead vocals are estimated, thereby converting the waveform representation into a sequence of musical note symbols. To reduce the inevitable errors during the estimation of the fundamental frequencies, the resulting note sequences are further smoothed, so that the abnormal notes can be corrected.

Since a popular song normally lasts for several minutes, it is highly unlikely that a user would sing a whole song as a query. Instead, a sung query usually corresponds to only a short segment of the target song. In view of this, phrase onset detection tries to locate points in a song related to the beginning of the query a user may sing to the MIR system, whereby the MIR system can limit the possible starting points to be examined to the phrase onsets of each song in the searching phase, thereby improving the retrieval efficiency. This is motivated by the observation that a user would tend to sing a phrase from the beginning of a sentence in the lyrics. For example, a user may query the system by singing the following lines from "*Yesterday*" by The Beatles: "*Suddenly, I'm not half the man I used to be. There's a shadow hanging over me*." We believe a sung query like "*I used to be. There's a shadow*" or "*half the man I used to be.*" would be highly unlikely. It is expected that pre-locating the phrase onsets would not only improve the retrieval efficiency but also help the MIR system use the "right" segment of a song to determine if this song matches the query.

As shown in Fig. 4, after indexing, each song in the collection is represented by a note sequence marked with phrase onsets. A segment of a song starting at a phrase onset is referred to as a music document. The note sequences of music documents serve as the basis to determine which music document best matches a given query.

B. Searching

In the searching phase, the system determines the song the user wants based on what he/she is singing. It is assumed that a user's sung query would be either a complete phrase or the initial part of that phrase. Thus, the task is to find the music document that best matches the query. The system commences with end-point detection, which records the singing voice. The recorded waveform is then converted into a sequence of note symbols by a note sequence generation and smoothing module the same as that used in the indexing phase. Next, the system measures the similarity between the note sequence of the query and the note sequence of each music document. As will be explained later, we employ multiple-pass Dynamic Time Warping to measure the similarities of melodies, which attempts to resolve the tempo errors and key errors that may occur in a user's query. Finally, the song associated with the music document whose note sequence is the most similar to the query's note sequence is regarded as relevant and presented to the user.

[1] A semitone is one twelfth of the interval (called an *octave*) between two sounds, one of which has twice the frequency of the other.

[2] For ease of illustration, we convert the sung notes into MIDI note numbers.

IV. MAIN MELODY EXTRACTION

In contrast to the retrieval of MIDI music, where it is easy to acquire the main melody by selecting one of the symbolic tracks, retrieving a polyphonic object from CD or karaoke music involves extracting the main melody directly from the accompanied singing signals, a task that is difficult to achieve by using conventional pitch estimation only. In this section, we describe our melody extraction method, which is specifically designed for karaoke music.

A. Background Accompaniment Reduction

Audio data extracted from karaoke VCDs consists of the signals in two channels for each track: an accompaniment-only signal $\mathbf{m}$, and an accompanied vocal signal $\mathbf{c} = \mathbf{s} + \mathbf{m}'$, where $\mathbf{s}$ denotes the underlying pristine vocal signal, and $\mathbf{m}'$ denotes the background accompaniment for $\mathbf{s}$. Figs. 5(a) and 5(b) show the waveforms of two channels for a piece of karaoke music. Usually, $\mathbf{m}$ sounds like $\mathbf{m}'$; hence, the pristine vocal signal $\mathbf{s}$ could be distilled by approximating $\mathbf{m}'$ with $\mathbf{m}$. However, since $\mathbf{m}$ and $\mathbf{m}'$ are not identical, subtracting $\mathbf{m}$ from $\mathbf{c}$ directly is of little use for distilling $\mathbf{s}$. Instead of direct subtraction, a promising solution would be to use adaptive filters, such as Least Mean Square (LMS) [28] or Recursive Least Square (RLS) [29], to estimate $\mathbf{m}'$ from $\mathbf{m}$. For computational efficiency, we assume that the main difference between $\mathbf{m}$ and $\mathbf{m}'$ lies in the amplitude and the phase (or frame lag), where the phase difference reflects the asynchronism between $\mathbf{m}$ and $\mathbf{m}'$. In a related work [30], Zhu and Gao proposed estimating the difference of volume level between $\mathbf{m}$ and $\mathbf{c}$, and then subtracting $\mathbf{m}$ from $\mathbf{c}$ after $\mathbf{m}$ is compensated by the difference. However, their method could not handle the case when $\mathbf{m}$ and $\mathbf{m}'$ differed in both amplitude and phase.

The concept of adaptive filtering can be realized by the following two computationally more efficient methods, which operate in the time domain and the frequency domain, respectively.

1) Time-domain method: The signals in both channels are first divided into frames by using a non-overlapping sliding window with a length of W waveform samples. Let $\mathbf{c}_t = \{c_{t,1}, c_{t,2},\ldots, c_{t,W}\}$ and $\mathbf{m}_t = \{m_{t,1}, m_{t,2},\ldots, m_{t,W}\}$ denote, respectively, the t-th frame of samples in the accompanied vocal channel and the accompaniment-only channel. We assume 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 background accompaniment for $\mathbf{s}_t$. If only the amplitude difference and phase difference between $\mathbf{m}$ and $\mathbf{m}'$ are considered, then $\mathbf{m}'_t$ can be approximated by $a_t\mathbf{m}_{t+bt}$, where $\mathbf{m}_{t+bt}$ is the b_t-th frame next to $\mathbf{m}_t$ that corresponds to $\mathbf{m}'_t$; and a_t is a scaling factor reflecting the amplitude difference between $\mathbf{m}_{t+bt}$ and $\mathbf{m}'_t$. Provided that $\mathbf{s}_t$, $\mathbf{m}_t$, and $\mathbf{m}'_t$ are random vectors, the optimal values of a_t and b_t can be estimated by minimizing the mean square error, $E\{|\mathbf{m}'_t - a_t\mathbf{m}_{t+bt}|^2\}$, where $E\{\cdot\}$ represents the expectation. However, since $\mathbf{m}'_t$ is not observable, minimizing $E\{|\mathbf{m}'_t - a_t\mathbf{m}_{t+bt}|^2\}$ directly is infeasible. An expedient solution to this problem is to minimize an equivalent quantity, $E\{|\mathbf{s}_t + \mathbf{m}'_t - a_t\mathbf{m}_{t+bt}|^2\}$. By assuming that $\mathbf{s}_t$ and $(\mathbf{m}'_t - a_t\mathbf{m}_{t+bt})$ are zero-mean and uncorrelated, $E\{|\mathbf{s}_t + \mathbf{m}'_t - a_t\mathbf{m}_{t+bt}|^2\}$ can be re-written as $E\{|\mathbf{s}_t|^2\} + E\{|\mathbf{m}'_t - a_t\mathbf{m}_{t+bt}|^2\}$. Since $E\{|\mathbf{s}_t|^2\}$ is invariant with respect to a_t and b_t, we have $\min_{\{a_t,b_t\}} E\{|\mathbf{m}'_t - a_t\mathbf{m}_{t+bt}|^2\} = \min_{\{a_t,b_t\}} E\{|\mathbf{s}_t + \mathbf{m}'_t - a_t\mathbf{m}_{t+bt}|^2\} = \min_{\{a_t,b_t\}} E\{|\mathbf{c}_t - a_t\mathbf{m}_{t+bt}|^2\}$.

By converting the solution from the minimum mean square sense to the least square sense, the optimal b_t can be found by choosing the value within a pre-set range ($\pm B$) that yields the least square estimation error, i.e.,

$$\hat{b}_t = \arg\min_{-B \le b_t \le B} |\mathbf{c}_t - \hat{a}_t \mathbf{m}_{t+b_t}|^2, \tag{1}$$

where $\hat{a}_t$ is the optimal amplitude scaling factor given $\mathbf{m}_{t+bt}$. Then, by letting $\frac{\partial |\mathbf{c}_t - a_t \mathbf{m}_{t+b_t}|^2}{\partial a_t} = 0$, we have a least-square-error solution of a_t as

$$\hat{a}_t = \frac{\mathbf{c}_t \cdot \mathbf{m}_{t+b_t}}{\|\mathbf{m}_{t+b_t}\|^2}. \tag{2}$$

Accordingly, the underlying vocal signal in frame t can be estimated by $\mathbf{s}_t = \mathbf{c}_t - \hat{a}_t \mathbf{m}_{t+\hat{b}_t}$.

2) Frequency-domain method: Like the time-domain method, the signals in both channels are first divided into frames, and then converted into magnitude spectra by a fast Fourier transform (FFT). Let $\mathbf{C}_t = \{C_{t,1}, C_{t,2},\ldots, C_{t,J}\}$ and $\mathbf{M}_t = \{M_{t,1}, M_{t,2},\ldots, M_{t,J}\}$ denote, respectively, the magnitude spectrum of the t-th frame of samples in the accompanied vocal channel and the accompaniment-only channel, where J is the number of frequency components. It can be assumed that $\mathbf{C}_t = \mathbf{S}_t + \mathbf{M}'_t$, where $\mathbf{S}_t = \{S_{t,1}, S_{t,2},\ldots, S_{t,J}\}$ is the vocal magnitude spectrum and $\mathbf{M}'_t = \{M'_{t,1}, M'_{t,2},\ldots, M'_{t,J}\}$ is the background accompaniment's magnitude spectrum. To find $\mathbf{S}_t$, we approximate $\mathbf{M}'_t$ by $a_t\mathbf{M}_{t+bt}$, where $\mathbf{M}_{t+bt}$ is the b_t-th frame next to $\mathbf{M}_t$ that probably corresponds to $\mathbf{M}'_t$; and a_t is a scaling factor that reflects the amplitude difference between $\mathbf{M}_t$ and $\mathbf{M}'_t$. The optimal b_t can then be found by choosing the value within a pre-set range ($\pm B$) that yields the least square estimation error, i.e.,

$$\hat{b}_t = \arg\min_{-B \le b_t \le B} |\mathbf{C}_t - \hat{a}_t \mathbf{M}_{t+b_t}|^2, \tag{3}$$

where $\hat{a}_t$ is the optimal amplitude scaling factor given $\mathbf{M}_{t+bt}$. Then, by letting $\frac{\partial |\mathbf{C}_t - a_t \mathbf{M}_{t+b_t}|^2}{\partial a_t} = 0$, we obtain the following least-square-error solution of a_t:

$$\hat{a}_t = \frac{\mathbf{C}_t \cdot \mathbf{M}_{t+b_t}}{\|\mathbf{M}_{t+b_t}\|^2}. \tag{4}$$

Accordingly, the vocal magnitude spectrum for frame t can be estimated by $\mathbf{s}_t = \mathbf{c}_t - \hat{a}_t \mathbf{M}_{t+\hat{b}_t}$.

Figures 5(c) and 5(d) show the resulting waveforms of an accompanied vocal channel after background accompaniment reduction is performed in the time domain and frequency domain, respectively. The test waveform signals were extracted from a karaoke VCD, and then down-sampled from the sampling rate of 44.1 kHz to 22.05 kHz. The frame length W was set to 2,048 samples, and the FFT size J was set to 2,048.

The waveform in Fig. 5(d) was obtained by applying an inverse FFT on the estimated vocal magnitude spectrum with the accompanied vocal channel's phase spectrum. From Figs. 5(c) and 5(d), we observe that both methods reduce the accompaniment in the accompanied vocal channel substantially. The comparison of the MIR performance using the time-domain method and the frequency-domain method is discussed in Section VII.

The vocal magnitude spectrum estimated by frequency-domain background accompaniment reduction can be used to compute musical notes sung in a song directly, without having to convert the spectrum back into the waveform. However, as will be described in detail later, the sung notes are computed on the basis of dividing the signal into overlapping frames, rather than non-overlapping frames, so that the sung note sequence can be extracted smoothly. Therefore, to combine frequency-domain background accompaniment reduction with note sequence generation, the former is actually performed with overlapping frames in our experiments.

B. Note Sequence Generation

The next stage of main melody extraction involves finding the sequence of sung notes in the estimated vocal signal. Let e_1, e_2,…, e_N be an inventory of possible notes performed by a singer. The task, therefore, is to determine which of N possible notes is most likely sung at each instant. Since musical notes differ from each other by the fundamental frequencies (F0s) they present, we may determine if a certain note is sung in each frame by analyzing the signal's magnitude in the frequency region that the F0 of the note is located.

Let $X_{t,j}$ denote the signal's magnitude with respect to the FFT index j in frame t, where $1 \le j \le J$. If time-domain background accompaniment reduction is used as the front-end of main melody extraction, $X_{t,j}$ can be obtained by dividing the estimated vocal signal into overlapping frames using a J-length sliding Hamming window, performing a J-length FFT for frame t, and then taking the magnitude spectrum. If frequency-domain background accompaniment reduction is used, $X_{t,j}$ is the estimated vocal magnitude spectrum for frame t, which is obtained by dividing the signals in both channels into J-length overlapping frames, performing a J-length FFT for each frame, and then estimating the vocal magnitude spectrum for frame t. Next, using the MIDI note number to represent e_1, e_2,…, e_N, and mapping the FFT indices into MIDI note numbers according to the F0 of each note, the signal's magnitude in note e_n in frame t can be estimated by

$$Y_{t,n} = \max_{\forall j, U(j)=e_n} X_{t,j} \,, \tag{5}$$

and

$$U(j) = \left\lfloor 12 \cdot \log_2\left(\frac{F(j)}{440}\right) + 69.5 \right\rfloor, \tag{6}$$

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.

Ideally, if note e_n is sung in frame t, $Y_{t,n}$ should be the maximum among the N magnitude values, $Y_{t,1}$, $Y_{t,2}$,…, $Y_{t,N}$. However, due to the existence of harmonics, note numbers several octaves higher than the sung note can also receive a large proportion of the signal's magnitude. Sometimes the magnitude associated with a harmonic note number can be even larger than the magnitude associated with the true sung note number; hence, the note number with the largest magnitude is not necessarily the note that was sung. To determine the sung note more reliably, we adapted Sub-Harmonic Summation (SHS) [27] to address this problem.

The principle applied here is that we compute a value for the "strength" of each possible note by summing the signal's magnitude in a note and its harmonic note numbers [3]. Specifically, the strength of note e_n in frame t is computed by

$$Z_{t,n} = \sum_{c=0}^{C} h^c \, Y_{t,n+12c} \,, \tag{7}$$

where C is the number of harmonics considered; and h is a positive value less than 1, which discounts the effect of higher harmonics. The result of this summation is that the note number corresponding to the signal's F0 receives the largest amount of magnitude from its harmonic notes. Thus, the sung note in frame t can be determined by choosing the note number associated with the largest strength value, i.e.,

$$o_t = \arg\max_{1\le n\le N} \; Z_{t,n} \,. \tag{8}$$

In this study, the waveform signal is divided into overlapping frames of 2,048 samples. The length of the shift between the frames is set to 344 samples, and the sampling frequency is 22,050 Hz; therefore, 22,050/344 ≈ 64 note symbols per second are computed for a given signal. Note that we do not segment signals into actual musical notes.

C. Note Sequence Smoothing

Although the background accompaniment in the accompanied vocal channel can be reduced to some extent, the residual accompaniment inevitably affects the extracted note sequence. This problem can be alleviated by using the *tone chroma*, which maps all the notes into 12 tone classes (C, Db, D, Eb, E, F, Gb, G, Ab, A, Bb, and B) by ignoring the difference between octaves. Since the background accompaniment often contains notes a number of octaves above or below the singing level, an incorrectly estimated sung note could still map to a correct tone class. However, because there are only 12 classes, tone chroma cannot express a melody pattern with sufficient precision to distinguish notes from one another. Therefore, instead of using the tone chroma representation, we develop a method to correct the error estimation of sung notes.

Our method is based on the concept of rectification, which identifies abnormal elements in a note sequence and forces them back to the normal notes. The abnormality in a note sequence generally arises from two types of errors: short-term errors and long-term errors. Short-term errors refer to rapid changes (e.g., jitters) between adjacent frames. This type of error can be corrected by using median filtering, which replaces each note with the local median of the notes of its neighboring frames. Long-term errors, on the other hand, refer to a succession of estimated notes that are not produced by a singer.

[3] In [27], the "strength" of a possible fundamental frequency is computed by summing the signal's magnitude associated with the frequency and its harmonic frequencies. In our experience, the note-level SHS method outperforms the original frequency-level SHS method in note extraction.

These wrong notes are probably several octaves above or below the true sung notes, which could result in the range of the estimated notes in a sequence being wider than that of the true sung note sequence. As discussed in Section II-B, the sung notes in a verse or chorus section seldom vary by more than 22 semitones. Therefore, we can adjust the suspect notes by shifting them several octaves up or down, so that the range of the notes in an adjusted sequence conforms to the normal range. Specifically, let $\mathbf{o} = \{o_1, o_2,\ldots, o_T\}$ denote a note sequence estimated using Eq. (8). The adjusted note sequence $\mathbf{o}' = \{o'_1, o'_2,\ldots, o'_T\}$ is obtained by

$$o'_t = \begin{cases} o_t & , \text{ if } |o_t - \bar{o}| \le (R/2) \\ o_t - 12 \times \left\lfloor \dfrac{o_t - \bar{o} + R/2}{12} \right\rfloor, & \text{if } o_t - \bar{o} > (R/2) \\ o_t - 12 \times \left\lfloor \dfrac{o_t - \bar{o} - R/2}{12} \right\rfloor, & \text{if } o_t - \bar{o} < (-R/2) \end{cases} , \quad (9)$$

where R is the normal range of the sung notes in a sequence and $\bar{o}$ is the mean note computed by averaging all the notes in $\mathbf{o}$. In Eq. (9), a note o_t is deemed a wrong note that must be adjusted if it is too far 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.

In our implementation, median filtering is performed in a sliding 5-note window with 4-note shifts. Consequently, the number of notes per second in a signal is reduced from 64 to 16. The note sequence is then rectified by using Eq. (9) with R=18, as per our previous work [13].

V. Phrase Onset Detection

Since a user's sung query tends to start at the beginning of a sentence in the lyrics, we can consider the lyrics as a collection of phrases and the beginning of each phrase as a point in a song where a user's sung query may begin. Therefore, if the onset time of each phrase can be located in the indexing phase, the MIR system can use the "right" segment of each song to determine if the song matches the query in the searching phase, thereby improving the retrieval efficiency.

Our strategy for detecting phrase onsets involves locating the boundaries where the signal in the accompanied vocal channel changes from accompaniment-only to a mixture of lead vocals and accompaniment. As mentioned earlier, the accompaniment in one channel of a karaoke track often resembles that in the other channel. Thus, if a passage does not contain lead vocals, the difference in the signal spectra of the two channels is minimal. In contrast, if a passage contains lead vocals, there must be a significant difference between the two channels during this passage. Therefore, if we examine the difference in the signal spectra of the two channels, we can locate the phrase onsets. In our system, we apply the Bayesian Information Criterion (BIC) [25] to determine the difference between the spectra.

A. The Bayesian Information Criterion (BIC)

The BIC is a model selection criterion that 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 $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_I\} \subset R^d$ and a model set $\mathbf{H} = \{H_1, H_2,\ldots, H_K\}$, the BIC value for model H_k is defined as

$$BIC(H_k) = \log p(\mathbf{X}|H_k) - 0.5\, \lambda\, \#(H_k) \log I, \quad (10)$$

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 with the largest BIC value.

Assume that there are two audio segments represented by feature vectors $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_I\}$ and $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2,\ldots, \mathbf{y}_I\}$, respectively. If we want to determine whether $\mathbf{X}$ and $\mathbf{Y}$ belong to the same acoustic class, then we must consider two hypotheses: "yes" and "no". Assuming that "yes" and "no" are characterized by stochastic models H_1 and H_2, respectively, our objective is to determine which model among H_1 and H_2 is better. For this purpose, we represent H_1 by a single Gaussian distribution $G(\boldsymbol{\mu},\boldsymbol{\Sigma})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are, respectively, the sample mean vector and the covariance matrix estimated using vectors $\{\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_I, \mathbf{y}_1, \mathbf{y}_2,\ldots, \mathbf{y}_I\}$. Meanwhile, H_2 is represented by two Gaussian distributions $G(\boldsymbol{\mu}_x,\boldsymbol{\Sigma}_x)$ and $G(\boldsymbol{\mu}_y,\boldsymbol{\Sigma}_y)$, where the sample mean vector $\boldsymbol{\mu}_x$ and covariance matrix $\boldsymbol{\Sigma}_x$ are estimated using vectors $\{\mathbf{x}_1, \mathbf{x}_2,\ldots, \mathbf{x}_I\}$; and the sample mean vector $\boldsymbol{\mu}_y$ and covariance matrix $\boldsymbol{\Sigma}_y$ are estimated using vectors $\{\mathbf{y}_1, \mathbf{y}_2,\ldots, \mathbf{y}_I\}$. Then, the problem of judging which model is better can be solved by computing the difference value of BIC between $BIC(H_1)$ and $BIC(H_2)$, i.e.,

$$\Delta BIC = BIC(H_2) - BIC(H_1). \quad (11)$$

Obviously, the larger the value of ΔBIC, the greater the likelihood that segments $\mathbf{X}$ and $\mathbf{Y}$ are from different acoustic classes. Conversely, the smaller the value of ΔBIC, the smaller the likelihood that $\mathbf{X}$ and $\mathbf{Y}$ are from different acoustic classes. We can therefore set a threshold for ΔBIC to determine if two audio segments belong to the same acoustic class.

B. Phrase Onset Detection by 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 lead vocals over accompaniment. Thus, by considering $\mathbf{X}$ and $\mathbf{Y}$ as two concurrent channel signals, ΔBIC can be computed for the entire recording, and then plotted as a curve. Fig. 6 shows the phrase onset detection process for a karaoke music clip. The process begins by splitting the waveform in each of the channels into 20-ms non-overlapping frames. 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. 6(c). A positive ΔBIC value indicates that the frame of the accompanied vocal channel contains lead vocals, but the corresponding frame in the accompaniment-only channel does not. In contrast, a negative ΔBIC value indicates that both frames contain accompaniment only. Therefore, the intervals where positive values of ΔBIC appear are identified as vocal sections.

In the example shown in Fig. 6(c), two vocal sections are identified, each marked by a solid line and a dashed line. In both sections, the "significant" local minimums on the ΔBIC

curve can be labeled as phrase onsets because a frame corresponding to a rest or breathing between phrases usually yields a small ΔBIC value. A local minimum at time t is considered significant if $\Delta BIC(t)$ is the minimum within the range $[t\text{-}1, t\text{+}1]$ (a two-second window centered on t) and both $\Delta BIC(\max_r)-\Delta BIC(t)$ and $\Delta BIC(\max_l)-\Delta BIC(t)$ exceed 0.5σ, where $\Delta BIC(\max_r)$ and $\Delta BIC(\max_l)$ are the right and left maximal ΔBIC values around t, respectively, and σ is the standard deviation of the ΔBIC values along the curve. Fig. 6(d) depicts the detected phrase onsets in this way. Note that the tradeoff between retrieval accuracy and retrieval efficiency depends to a large extent on the number of detected phrase onsets. In general, the larger the number of likely phrase onsets detected, the higher the retrieval accuracy will be. However, increasing the number of candidate phrase onsets often reduces the retrieval efficiency drastically.

The phrase onset detection approach is very efficient. It takes less than 0.1 times realtime to process a song when running on a PC with a 3.4GHz Intel Pentium IV CPU. In other words, phrase onset detection for a 5-minute song can be finished within half a minute.

VI. MELODY SIMILARITY COMPARISON

Given a user's sung query and a set of music documents, each of which is represented by a note sequence, our task is to find the music document whose partial note sequence is the most similar to the query's note sequence. Here, a music document refers to a segment of a song starting at a phrase onset.

A. 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, respectively, the note sequences extracted from a user's query and a particular music document to be compared. As the lengths of **q** and **u** are usually different, computing the Euclidean distance between **q** and **u** directly is infeasible. This problem is usually resolved by using Dynamic Time Warping (DTW) to find the temporal mapping between **q** and **u.** 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 by

$$D(t,\ell) = \min\begin{cases} D(t-2,\ell-1) + 2\times d(t,\ell) \\ D(t-1,\ell-1) + d(t,\ell) - \varepsilon \\ D(t-1,\ell-2) + d(t,\ell) \end{cases}, \tag{12}$$

and

$$d(t,\ell) = |\, q_t - u_\ell |\,, \tag{13}$$

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

To compensate for the inevitable errors that arise from phrase onset detection, we assume that the true phrase onset time associated with the automatically detected phrase onset time t_{os} is within the range $[t_{os}\text{-}r/2,\ t_{os}\text{+}r/2]$, where r is a predefined tolerance. The DTW recursion in Eq. (12) indicates that the best mapping length of **u** to **q** is limited to between $T/2$ and $2T$. However, we prefer to limit the longest mapping length of **u** to **q** to kT, say $k = 1.2$, so that the mapping is more precise. In other words, the tempo of the query can be $1/k$ times to twice the tempo of the target music document. Therefore, in the implementation, we set each phrase onset time to $t_{os}–r/2$, instead of t_{os}, to form **u** using the note sequence of a song from $t_{os}–r/2$ to $t_{os}+r/2+kT$. The boundary conditions for the DTW recursion are then defined as,

$$\begin{cases} D(1,1) = d(1,1) \\ D(t,1) = \infty,\ 2 \le t \le T \\ D(t,2) = \infty,\ 4 \le t \le T \\ D(1,\ell) = \begin{cases} d(1,\ell), & 1 \le \ell \le r \\ \infty, & r < l \le L \end{cases} \\ D(2,\ell) = \begin{cases} d(1,\ell-1) + d(2,\ell), & 2 \le \ell \le r+1 \\ \infty, & r+1 < l \le L \end{cases} \\ D(3,2) = d(1,1) + 2\times d(3,2) \end{cases}. \tag{14}$$

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

$$S(\mathbf{q},\mathbf{u}) = \max_{T/2 \le \ell \le r+kT} 1/D(T,\ell). \tag{15}$$

B. Using 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 target music document could be rather different. To deal with this problem, the dynamic range of a query's note sequence must be adjusted to that of the music document to be compared. This can be done by shifting the query's note sequence up or down several semitones, so that the mean of the shifted query's note sequence is equal to that of the music document to be compared. Briefly, a query's note sequence is adjusted by

$$q_t \leftarrow q_t + (\bar{u} - \bar{q}), \tag{16}$$

where $\bar{q}$ and $\bar{u}$ are the means of the query's note sequence and the music document's note sequence, respectively. However, in our experiments, we found that the above adjustment could not fully resolve the transposition problem, since the value of $(\bar{q} - \bar{u})$ only reflects the global difference in the keys of a query and a music document. In other words, it cannot characterize a partial transposition or key change over the course of a query. To handle this problem better, we modify the DTW similarity comparison by considering the key shifts in a query's note sequence. Specifically, a query sequence **q** is shifted ±1, ±2,..., ±V semitones to span a set of note sequences $\{\mathbf{q}^{(-V)}, \ldots, \mathbf{q}^{(-2)}, \mathbf{q}^{(-1)}, \mathbf{q}^{(1)}, \mathbf{q}^{(2)}, \ldots, \mathbf{q}^{(V)}\}$. For a document sequence **u**, the similarity $S(\mathbf{q}, \mathbf{u})$ is then determined by choosing the sequence among $\{\mathbf{q}^{(-V)}, \ldots, \mathbf{q}^{(-2)}, \mathbf{q}^{(-1)}, \mathbf{q}^{(0)}, \mathbf{q}^{(1)}, \mathbf{q}^{(2)}, \ldots, \mathbf{q}^{(V)}\}$ that is most similar to **u**, i.e.,

$$S(\mathbf{q},\mathbf{u}) = \max_{-V \le v \le V} S(\mathbf{q}^{(v)},\mathbf{u}), \tag{17}$$

where $\mathbf{q}^{(0)} = \mathbf{q}$. However, increasing the value of V substantially increases the computational cost because the similarity comparison requires two extra DTW operations whenever the value of V is increased by one. Thus, the value of V should be decided according to the tradeoff between the retrieval accuracy and the retrieval efficiency.

C. Bypassing the Voiceless Regions of a Query

In addition to key and tempo differences between queries and documents, the voiceless regions in a sung query represent another issue that must be addressed when designing a query-by-singing MIR system. The voiceless regions, which may be due to a rest, a pause, or breathing, result in some notes in the query note sequence being tagged with "0". However, the corresponding non-vocal regions in the music document are usually not tagged with "0" because there is accompaniment in those regions. This discrepancy may severely discount the similarity $S(\mathbf{q}, \mathbf{u})$ for any $\mathbf{q}$ and $\mathbf{u}$ that have the same tune, as shown in Fig. 7. The gray regions in Fig. 7(b) do not contain a singing voice. Although the voiceless regions in a sung query can be detected by simply using the magnitude information, 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. (13) to

$$d(t,\ell) = \begin{cases} |q_t - u_\ell|, & q_t \neq 0 \\ \varphi, & q_t = 0 \end{cases}, \qquad (18)$$

where φ is a small constant. Implicitly, Eq. (18) is equivalent to bypassing the voiceless regions of a query. As noted in [13], the retrieval performance can be improved slightly if the non-singing regions of a sung query are detected and bypassed during the DTW similarity comparison. Therefore, in the following experiments, $d(t,\ell)$ is computed by Eq. (18).

D. Using Multiple-Level Data Abstraction (MLDA) to Improve Retrieval Efficiency

As shown in Fig. 8(a), the computational complexity of the DTW recursion in Eq. (12) for constructing a $T \times L$ table can be expressed as the polygon area surrounded by the red lines, i.e.,

$$\begin{aligned} Complexity &= T \times L - \frac{T \times T/2}{2} - \frac{L \times L/2}{2} \\ &= TL - \frac{(T^2 + L^2)}{4}. \end{aligned} \qquad (19)$$

Recall that L is preferably set to kT, where $1/2 \leq k \leq 2$; therefore, the computational complexity can be rewritten as

$$Complexity = \frac{(4k - k^2 - 1)T^2}{4} \equiv O(T^2). \qquad (20)$$

In our preliminary experiments, we found that the document length can simply be 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. (20), the computational complexity is $0.59T^2$. As shown in Fig. 8(b), the introduction of a phrase onset tolerance in DTW increases the computational complexity by rT. If r is small compared to T, the increase in complexity is negligible; however, if $r = 0.59T$, the computational complexity is twice that of a typical DTW.

Since the complexity is $O(T^2)$, the best way to speed up the search process is to reduce the value of T. Motivated by Keogh and Pazzani's Piecewise Aggregate Approximation (PAA) [23], we propose a dimensionality reduction technique, called Multi-Level Data Abstraction (MLDA). Unlike PAA, which divides a time series into frames of equal-length and then calculates the mean value of data within each frame, MLDA prunes music documents that are probably irrelevant to the query. The pruning is carried out in a level-by-level manner. At each level of pruning, the compression rate of the data is the power of 2. Specifically, deleting alternate notes in a sequence reduces it to half the original length; therefore, the computational complexity is reduced by 75%. However, since data abstraction could degrade the discriminability between note sequences, this process is only used to quickly identify documents that are probably irrelevant to the query. After such documents have been excluded from the search list, the remaining documents are examined according to their original note sequences. Therefore, if the original computational complexity without MLDA is CC, and the pruning rate is set to R_{pr}, then the total computational complexity of a two-level data abstraction DTW is $[1/4+(1-R_{pr})]CC$. If a three-level data abstraction DTW is applied, the complexity is reduced to $[(1/4)^2+(1/4)(1-R_{pr})+(1-R_{pr})^2]CC$. Thus, when the pruning rate R_{pr} is set to 0.75, the complexity of a two-level data abstraction DTW and a three-level data abstraction DTW is, respectively, 1/2 and 3/16 that of a regular DTW.

VII. EXPERIMENTS

A. 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. Extraction of the main melody yielded a note sequence of approximately 16 note symbols per second of music signal. To compare the system's performance using automatic and manual detection of phrase onsets, we manually labeled the phrase onsets with respect to 95 songs, which gave us 775 marked phrase onsets. For ease of discussion, the subset of 95 songs was denoted as DB-95, while the complete set of 1,071 songs was denoted as DB-1071.

We collected 90 sung queries from 9 male and 4 female users, in which each query was associated with one of the songs in DB-95. The queries were recorded at a sampling rate of 22.05 kHz. The duration of each query ranged from 15 to 45 seconds, but only the first 8-second portion was input to the system; hence, the length of a query's note sequence was 8 × 16 = 128 note symbols.

The retrieval performance was measured on the basis of *song accuracy* defined as,

$$Song\ accuracy(\%) = \frac{\#\,queries\ receiving\ the\ correct\ songs}{\#\,queries} \times 100\%. \qquad (21)$$

We also considered the scenario that the MIR system returned a list of Top-N ranked songs for users to choose from. In this case, we computed the Top-N accuracy, defined as the percentage of queries whose target songs were among the Top-N ranked songs.

B. Experiment Results

1) Effectiveness of background accompaniment reduction: We conducted the first experiment on DB-95 to evaluate the effectiveness of the proposed background accompaniment reduction approaches. For this case, the phrase onsets were labeled manually. The query's note sequence was shifted to the mean of the note sequence of the music document to be compared. Comparison of the melody similarity was based on

Eq. (15), i.e., a single-pass DTW was used. Table I shows the retrieval performance. Although both the frequency-domain and time-domain methods for background accompaniment reduction improve the retrieval performance, the former approach is more effective. A possible reason for this is that the frequency-domain background accompaniment reduction approach worked directly on the magnitude spectrum, which is consistent with the subsequent note sequence generation. Though the time-domain method could also reduce the background accompaniment to some extent, the effect was not direct to the reduction of the background accompaniment's frequency components that interfered with the determination of sung notes. Accordingly, the subsequent experiments were conducted using frequency-domain background accompaniment reduction.

2) Effectiveness of Multiple-Pass DTW: The second experiment was conducted to examine if the retrieval performance can be improved by addressing the transposition problem. Specifically, using Eq. (17), we shifted a query's note sequence up or down several semitones to compare its similarity to each music document's note sequence. Table II shows the experiment results evaluated using DB-95. Here, $V = 0$ means that no shifting was performed, and the results correspond to the best results (obtained with frequency-domain background accompaniment reduction) shown in Table I. The results in Table II show that the retrieval performance improves as the value of V increases, which indicates that the more we consider possible changes of key, the greater the chance that a query's note sequence will match the correct music document's note sequence. However, increasing the value of V increases the computational cost substantially, because the similarity comparison requires two extra DTW operations whenever the value of V is increased by one. In subsequent experiments, we used an economic value of $V = 1$.

3) Manual phrase onset labeling vs. automatic phrase onset detection: The third experiment was conducted to compare the retrieval performance obtained with manual phrase onset labeling and automatic phrase onset detection. The automatic phrase onset detection approach marked 2,548 phrase onsets in the 95 songs in DB-95, which was about 3.3 times the number of hand-labeled phrase onsets (i.e., 775); hence, the search time was approximately 3.3 times longer than that of the system based on manual phrase onset labeling. As mentioned in Sec. VI-A, we use a tolerance r to compensate for the inevitable errors that arise from automatic phrase onset detection. Obviously, the larger the value of r is set, the higher the chance of locating the right segment to perform document-query matching. However, the increase of the value of r also increases the search time and the risk of assigning a wrong document to the query. To tradeoff these conflicting factors, r was set to 3 seconds in this study. The recall rate of automatic phrase onset detection was 75.0%, when a hand-labeled phrase onset at time t was considered as being detected correctly if there was an automatically-detected phrase onset within the range [t-3/2, t+3/2] (a three-second window centered on t). It is worth mentioning that, although some hand-labeled phrase onsets were missed, a song could still be retrieved successfully as long as at least one of the repeating phrases' onsets was detected correctly. Table III shows the retrieval performance; the results of "manual phrase onset labeling" correspond to the results of "$V = 1$ (Three-pass DTW)" in Table II. We observe that, although the retrieval accuracy declines noticeably when the phrase onsets are marked automatically instead of manually, a vast majority of queries receives correct returned answers.

4) Evaluation using the database of 1,071 songs: In this experiment, we evaluated the performance of the karaoke retrieval system on DB-1071, with the same 90 queries. The system automatically marked 26,120 phrase onsets in the 1,071 songs. The retrieval performance is given in Table IV. We observe that the Top-1 accuracy drops from 72.2% to 62.2% as the database expands from 95 songs to 1,071 songs, while the Top-10 accuracy drops from 86.7% to 77.8%. To speed up the search, a four-level DTW, with two pruning rates, R_{pr}=0.75 and R_{pr}=0.9, was implemented. The results in Table IV show that using a multiple-level data abstraction DTW reduces the search time substantially at the expense of a small degradation of the retrieval accuracy. The running time on a PC with a 3.4GHz Intel Pentium IV CPU for an 8-second query is around 15 seconds, about twice the real time.

IV. Conclusion

We have proposed a karaoke music retrieval system that allows users to locate desired songs by singing part of the lyrics to the system. Since the lead vocals and various accompaniments are combined in an accompanied vocal channel, the proposed method reduces the accompaniment in that channel to enhance the accuracy of main melody extraction. We also employ a rectification method that is based on the characteristic structure of popular songs to correct the error estimation of sung notes. In addition, we use the Bayesian Information Criterion (BIC) to detect the onsets of the musical phrases in a song, corresponding to the points of the song where a sung query may begin. Phrase onset detection, in conjunction with multiple-level multiple-pass DTW matching for similarity comparison, facilitates an efficient and effective search of a large music database. In contrast to existing work on query-by-humming/singing techniques handling small polyphonic music databases, the results of experiments conducted on a music database comprised of 1,071 songs demonstrate the feasibility of retrieving a large karaoke music database by singing.

Despite the potential, the methods proposed in this study can only be regarded as a preliminary investigation in realistic karaoke applications. To be of more practical use, more work is needed to study the problems of query formulation and usability. In particular, as there are inevitable strong noises in a karaoke room, it is necessary to develop a system robust against various ambient noises.

References

[1] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith, "Query by humming: musical information retrieval in an audio database," in *Proc. ACM International Conference on Multimedia*, 1995.

[2] C. C. Liu, J. L. Hsu, and A. L. P. Chen, "An approximate string matching algorithm for content-based music data retrieval," in *Proc. IEEE International Conference on Multimedia Computing and Systems,* 1999.

[3] J. S. Mo, C. H. Han, and Y. S. Kim, “A melody-based similarity computation algorithm for musical information,” in *Proc. Workshop on Knowledge and Data Engineering Exchange*, 1999.

[4] N. Kosugi, Y. Nishihara, T. Sakata, M. Yamamuro, and K. Kushima, “A practical query-by-humming system for a large music database,” in *Proc. ACM International Conference on Multimedia*, 2000.

[5] J. S. R. Jang and H. R. Lee, “Hierarchical filtering method for content-based music retrieval via acoustic input,” in *Proc. ACM International Conference on Multimedia,* 2001.

[6] N. Hu and R. B. Dannenberg, “A comparison of melodic database retrieval techniques using sung queries,” in *Proc. ACM/IEEE Joint Conference on Digital Libraries*, 2002.

[7] S. Pauws, “CubyHum: A fully operational query by humming system,” in *Proc. International Conference on Music Information Retrieval*, 2002.

[8] T. Nishimura, 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,” in *Proc. International Symposium on Music Information Retrieval,* 2001.

[9] J. Song, S. Y. Bae, and K. Yoon, “Mid-level music melody representation of polyphonic audio for query-by-humming system,” in *Proc. International Conference on Music Information Retrieval*, 2002.

[10] J. Foote, “ARTHUR: retrieving orchestral music by long-term structure,” in *Proc. International Symposium on Music Information Retrieval*, 2000.

[11] S. Doraisamy and S. M. Rüger, “An approach towards a polyphonic music retrieval system,” in *Proc. International Symposium on Music Information Retrieval,* 2001.

[12] J. Pickens, J. P. Bello, G. Monti, M. Sandler, T. Crawford, M. Dovey, and D. Byrd, “Polyphonic score retrieval using polyphonic audio queries: a harmonic modeling approach,” *Journal of New Music Research*, vol. 32, no. 2, pp. 223–236, 2003.

[13] H. M. Yu, W. H. Tsai, and H. M. Wang, “A query-by-singing technique for retrieving polyphonic objects of popular music,” in *Proc. Asian Information Retrieval Symposium*, 2005.

[14] S. Doraisamy and S. Rüger, “Robust polyphonic music retrieval with *n*-grams,” *Journal of Intelligent Information Systems*, vol. 21, no. 1, pp. 53–70, July 2003.

[15] J. Shifrin and W. Birmingham, “Effectiveness of HMM-based retrieval on large databases,” in *Proc. International Conference on Music Information Retrieval*, 2003.

[16] C. Meek and W. P. Birmingham, “Automatic thematic extractor,” *Journal of Intelligent Information Systems*, vol. 21, no. 1, pp. 9–33, July 2003.

[17] R. Typke, R. C. Veltkamp, and F. Wiering, “Searching notated polyphonic music using transportation distances,” in *Proc. ACM Conference on Multimedia*, 2004.

[18] W. H. Tsai and H. M. Wang, “A query-by-example framework to retrieve music documents by singer,” in *Proc. IEEE International Conference on Multimedia and Expo*, 2004.

[19] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Trans. Speech and Audio Processing*, vol. 10, no. 5, pp. 293-302, July 2002.

[20] Y. Feng, Y. Zhuang, and Y. Pan, “Popular music retrieval by detecting mood,” in *Proc. ACM Conference on Research and Development in Information Retrieval*, 2003.

[21] M. Goto, “A predominant-F0 estimation method for CD recordings: MAP estimation using EM algorithm for adaptive tone models,” in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing,* 2001.

[22] J. Eggink, and G. J. Brown, “Extracting melody lines from complex audio,” in *Proc. International Conference on Music Information Retrieval*, 2004.

[23] E. Keogh and M. Pazzani, “Scaling up dynamic time warping for datamining applications,” in *Proc. ACM SIGKDD*, 2000.

[24] S. Salvador and P. Chan, “FastDTW: toward accurate dynamic time warping in linear time and space,” in *Proc. KDD Workshop on Mining Temporal and Sequential Data*, 2004.

[25] G. Schwarz, “Estimation the dimension of a model,” *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[26] Cakewalk, Inc., http://www.cakewalk.com/

[27] M. Piszczalski and B. A. Galler, “Predicting musical pitch from component frequency ratios,” *Journal of the Acoustical Society of America*, vol. 66, no. 3, pp. 710–720, Sep. 1979.

[28] B. Widrow, J. R. Glover, J. M. McCool, J. Kaunitz, C. S. Williams, R.H. Hearn, J. R. Zeidler, E. Dong, and R. C. Goodlin, “Adaptive noise cancelling: principles and applications,” *Proc. IEEE*, vol. 63, no. 12, pp. 1692–1716, Dec. 1975.

[29] S. Haykin, *Adaptive Filter Theory*, 4th ed., Prentice-Hall, NY, 2002.

[30] Y. Zhu and S. Gao, “Extracting vocal melody from karaoke music audio,” in *Proc. IEEE International Conference on Multimedia & Expo*, 2005.

[31] B. Pardo, W. Birmingham, and J. Shifrin, “Name that tune: a pilot study in finding a melody from a sung query,” *Journal of the American Society for Information Science and Technology*, vol. 55, no. 4, pp. 283–300, 2004.

[32] B. Pardo, “Finding structure in audio for music information retrieval,” *IEEE Signal Processing Magazine*, vol. 23, no. 3, pp. 126–132, 2006.

[33] T. D. Mulder, J. P. Martens, S. Pauws, F. Vignoli, M. Lesaffre, M. Leman, B. D. Baets, and H. D. Meyer, “Factors affecting music retrieval in query-by-melody,” *IEEE Transactions on Multimedia*, vol. 8, no. 4, pp.728–739, 2006.

[34] R. B. Dannenberg, W. P. Birmingham, G. Tzanetakis, C. Meek, N. Hu, and B. Pardo, “The MUSART testbed for query-by-humming evaluation,” in *Proc. International Conference on Music Information Retrieval*, 2003, pp. 41-50.

[35] R. B. Dannenberg, W. P. Birmingham, B. Pardo, N. Hu, C. Meek, G. Tzanetakis, "A comparative evaluation of search techniques for query-by-humming using the MUSART testbed," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 2, pp. 687–701, 2007.

[36] A. Duda, A. Nürnberger, and S. Stober, "Towards query by singing/humming on audio databases," in *Proc. International Conference on Music Information Retrieval,* 2007.

[37] S. Pauws, "Effects of song familiarity, singing training and recent song exposure on the singing of melodies," in *Proc. International Conference on Music Information Retrieval,* 2003.

[38] J. Foote, "Visualizing music and audio using self-similarity," in *Proc. ACM Multimedia*, 1999.

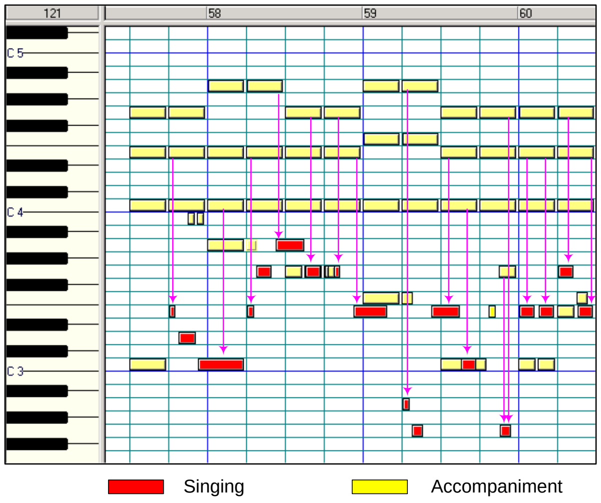


Fig. 3. A fragment of the pop song "*Let It Be*" by The Beatles in which the tune is manually converted to a MIDI file.

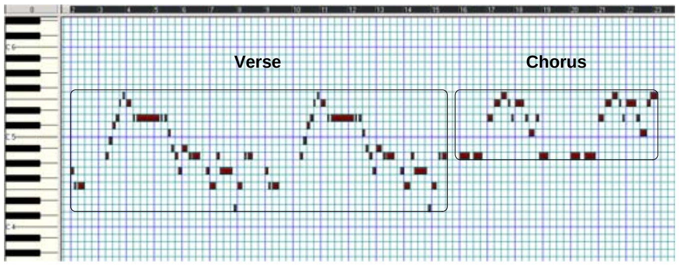


Fig. 1. A fragment of the pop song "*Yesterday*" by The Beatles in which the singing is converted into a MIDI file. For ease of illustration, it is shown by Cakewalk^{TM} [26] software.

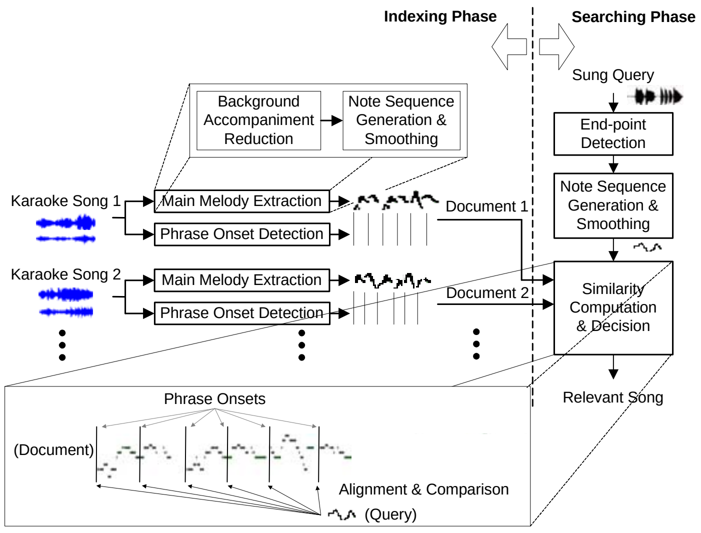
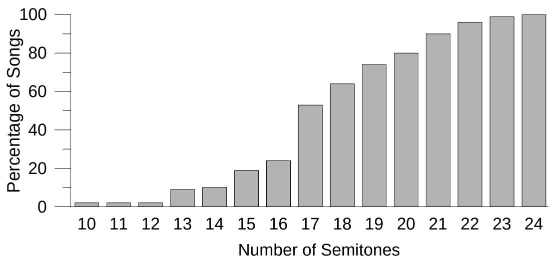
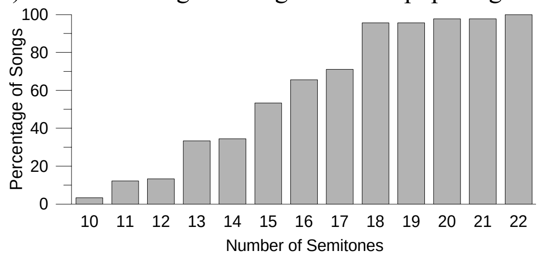


Fig. 4. The proposed karaoke music retrieval system.



(a) The range of sung notes in a pop song



(b) The range of sung notes in a verse or chorus section
Fig. 2. Statistics of the range of sung notes in 50 pop songs.

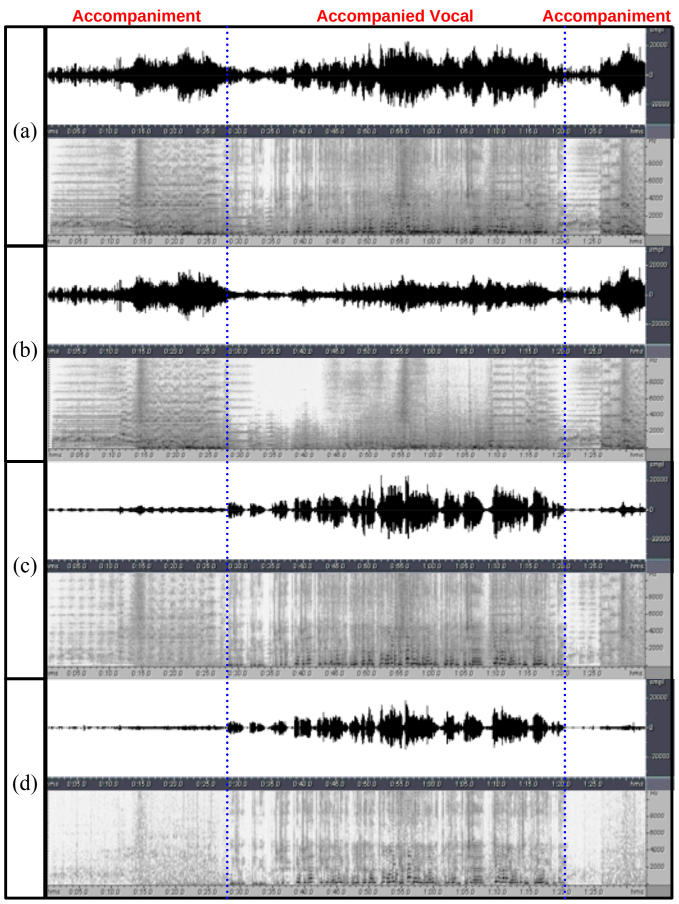


Fig. 5. (a) An accompanied vocal channel. (b) An accompaniment-only channel. (c) The accompanied vocal channel after time-domain background accompaniment reduction. (d) The accompanied vocal channel after frequency-domain background accompaniment reduction.

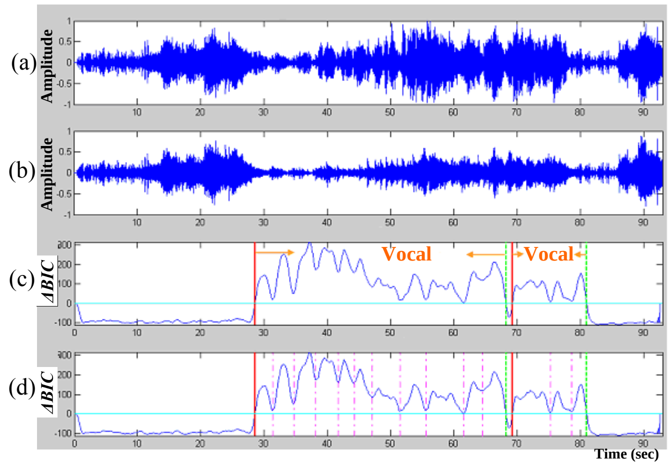


Fig. 6. An example of phrase onset detection: (a) the waveform of an accompanied vocal channel; (b) the waveform of an accompaniment-only channel; (c) the ΔBIC curve and the detected vocal sections; and (d) the results of phrase onset detection.

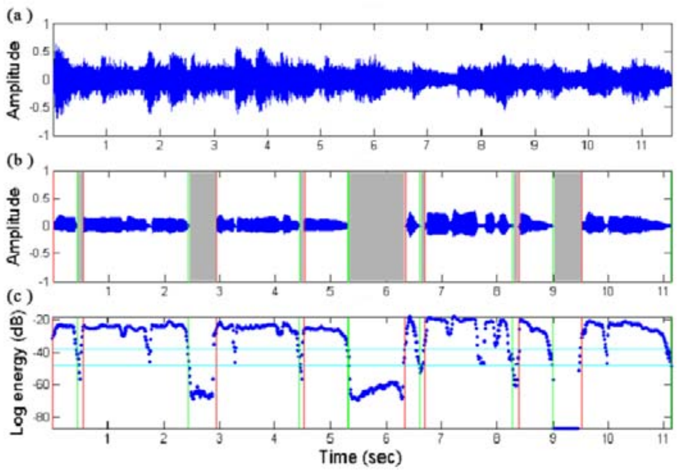


Fig. 7. (a) A phrase in a music document. (b) A query sung according to the phrase. (c) The log-energy profile of the sung query.

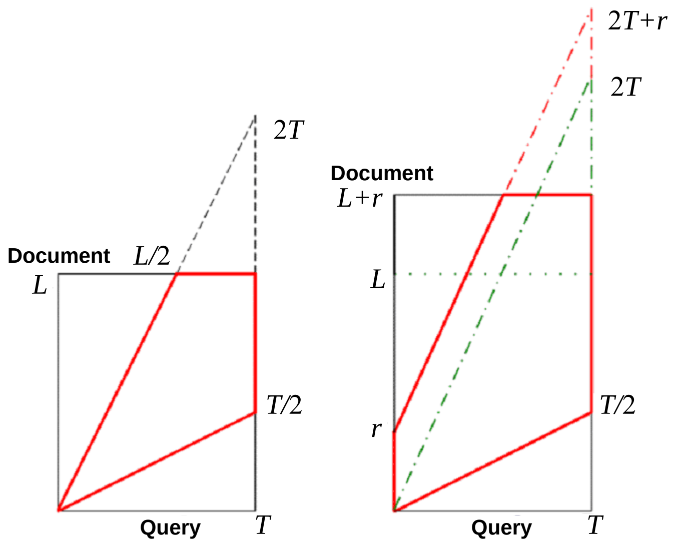


Fig. 8. (a) The search space for a typical DTW in which the initial frames of a query and the target phrase are aligned. (b) The search space for the DTW we apply, where the initial frame of a query can be mapped to any one of the first *r* frames of the target phrase.