A QUERY-BY-EXAMPLE TECHNIQUE FOR RETRIEVING COVER VERSIONS OF POPULAR SONGS WITH SIMILAR MELODIES

Wei-Ho Tsai **Hung-Ming Yu** **Hsin-Min Wang**

Institute of Information Science, Academia Sinica
Taipei, Taiwan, Republic of China
wesley,donny,whm@iis.sinica.edu.tw

ABSTRACT

Retrieving audio material based on audio queries is an important and challenging issue in the research field of content-based access to popular music. As part of this research field, we present a preliminary investigation into retrieving cover versions of songs specified by users. The technique enables users to listen to songs with an identical tune, but performed by different singers, in different languages, genres, and so on. The proposed system is built on a query-by-example framework, which takes a fragment of the song submitted by the user as input, and returns songs similar to the query in terms of the main melody as output. To handle the likely discrepancies, e.g., tempos, transpositions, and accompaniments between cover versions and the original song, methods are presented to remove the non-vocal portions of the song, extract the sung notes from the accompanied vocals, and compare the similarities between the sung note sequences.

Keywords: cover version, main melody, query-by-example, accompaniments.

1 INTRODUCTION

Rapid advances in Internet connectivity and signal processing technologies have led to a dramatic and unprecedented increase in the availability of music material in recent years. Ironically, it has become more and more difficult to locate desired items from the innumerable options. Thus, techniques that could enable users to quickly acquire the music they want are being extensively explored to keep pace with the rapid proliferation of music material. Among such techniques, retrieving audio material based on audio queries is of particular interest in the domain of accessing popular music. Its root concept of query-by-humming or query-by-song continues to motivate the development of many promising solutions for retrieving music beyond the conventional text-processing paradigm, such as allowing users to retrieve a song by humming a catchy tune without needing to name the song [1-9], or helping users find songs performed by their favorite singers [10], genre [11], mood [12], etc., by playing an excerpt of the music as a query. In tandem with the above solutions, this study presents our preliminary investigation of the retrieval of cover recordings, which aims to find songs with melodies similar to the melody of a user's song query.

A cover version of a song refers to a new rendition of a song that was originally recorded and made popular by another artist. It is often used as a means to attract audiences who like a familiar song, or to increase the popularity of an artist by adapting a proven hit. Sometimes pop musicians gain publicity by recording a cover version that contrasts with the original recording. Over several years, thousands upon thousands of cover versions have been recorded, some of which are virtually identical to the original version, while some are radically different. The only feature that is almost invariant in the different recordings is the main melody of the vocals. Usually, the most frequent difference between a cover song and the original version is that they are performed by different singers. In such cases the associated tempos, ornaments, accompaniments, etc., may be changed to cater to the taste of contemporary audiences, or to fit the theme of an album. Thus, in a music retrieval system, it would be useful if a search function for a single song rendered by different singers or belonging to different genres could be provided.

Other common differences between cover versions and the original song are that they may have different lyrics and titles, or they are sung in different languages. In particular, a hit song can often be translated into different languages, thereby making it more popular worldwide. Since a translation is usually not literal, cover-version retrieval based on the main melody would be more feasible than text-based retrieval for those wishing to listen to a song rendered in a different language. In addition, it is commonplace for live performances to be recorded and then released as authorized cover songs. The method of cover-song retrieval could thus be applied to index and classify such undocumented live recordings. This would also help copyright holders detect unauthorized or bootleg concert recordings.

In this work, we address the problem of cover-version retrieval by investigating how to determine if one or more music collections contain similar melodies to a specified song query. This task belongs to the problem of retrieving *polyphonic* music documents based on polyphonic music queries. In contrast to *monophonic* music, in which at most one note is played at any given time, polyphonic music often contains many notes that are played simultaneously. Thus, it is difficult to extract the main melody automatically from polyphonic music [13]. Due to this difficulty, a large number of current query-by-humming systems [1-4] work within the monophonic domain, which converts a monophonic audio query into a symbolic format to match a monophonic symbolic collection. Some studies [14,15] focus on locating the major themes from a piece of polyphonic symbolic music, in which the note information is given as *a priori*. However, very few systems operate in the mode of monophonic audio queries on a polyphonic audio collection [5,6], or entirely polyphonic audio queries on a polyphonic audio collection [7-9]. This work further differs from the above systems by the need to compare the main melody present in the vocals of polyphonic music. Thus, the proposed methods, though drawn from the query-by-humming paradigm, are specifically tailored to solve the problem of cover-version retrieval.

2 METHOD OVERVIEW

Our goal is to design a system that takes as input an audio query from a fragment of a song, and produces as output a ranked list of songs that are similar to the query in terms of the main melody. Songs ranked high are then considered as the cover or original versions of the song requested by the user. However, as cover versions may differ significantly from the original song in the way that the accompaniments are introduced, an arbitrary audio query could contain non-vocal (accompaniment-only) segments whose melody patterns are not present in the songs requested by the user, or vice versa. To simplify the problem during this initial development stage, we assume that a user's query does not contain salient non-vocal segments.

In general, the structure of a popular song can be divided into five sections: 1) *intro*, usually the first 5-20 seconds of the song, which is simply an instrumental statement of the subsequent sections; 2) *verse*, which typically comprises the main theme of the story represented in the song's lyrics; 3) *chorus*, which is often the heart of a song, where the most recognizable melody is present and repeated; 4) *bridge*, which comes roughly two-thirds into a song, where a key change, tempo change or new lyric is usually introduced to create a sensation of something new coming next; 5) *outro*, which is often a fading version of the chorus or an instrumental restatement of some earlier sections to bring the song to a conclusion. In essence, the verse and chorus contain the vocals sung by the lead singer, while the intro, bridge, and outro are largely accompaniments. Since a vast majority of popular songs follow the structure of "intro-verse-chorus-verse-chorus-bridge-chorus-outro", we further assume that a user would submit a fragment of the region between the intro and the bridge (if at all) of a song to the system.

Figure 1 shows a block diagram of our cover-version retrieval system, which operates in two phases: indexing and searching. The indexing phase generates the melody description for each of the songs (documents) in the collection. It commences with removal of the non-vocal segments longer than two seconds[1], which very likely belong to the intro, bridge, or outro. Then, main melody extraction proceeds by converting each song from the waveform samples into a sequence of musical note symbols. In the searching phase, the task is to determine which of the songs (documents) are relevant to a music query. This phase begins with the main melody extraction, which converts the audio query into a sequence of musical note symbols, and is followed by comparison of the similarities between the query's note sequence and each document's note sequence. The more similar the document's note sequence, the more relevant the document will be to the song requested by the user. Then, a ranked list of the similarities between the query's sequence and the document's sequence is presented to the user.

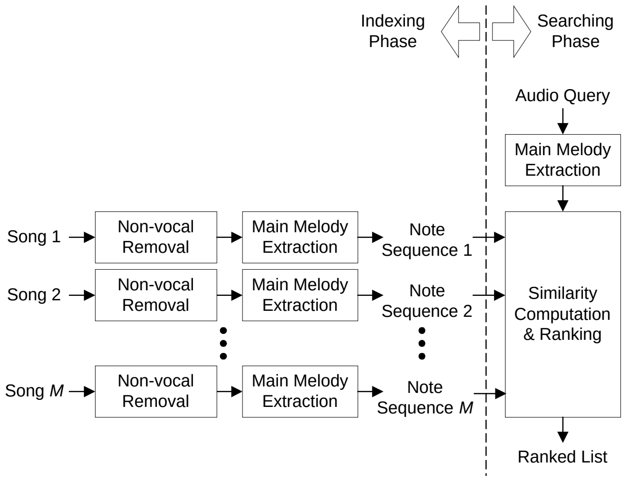


Figure 1. The proposed cover-version retrieval system.

3 NON-VOCAL SEGMENT REMOVAL

Although it would be desirable if all the non-vocal regions within a music recording could be located automatically, the task of accurately distinguishing between the segments with and without singing is rather difficult. Our previous work [16] on this problem found that a vocal segment tends to be classified as non-vocal if it is mixed with loud background accompaniment. Although discarding a low "vocal-to-accompaniment-ratio" segment is almost harmless in some applications, such as singer clustering [16], it could result in a very fragmented and unnatural melody pattern being extracted from a song. Thus, instead of locating all the vocal and non-vocal boundaries of a song document, we only try to detect the non-vocal segments that are longer than two seconds.

The basic strategy applied here is adapted from our previous work [16], in which a stochastic classifier is

[1] This corresponds to a *whole rest*, if 120 BPM is assumed.

constructed to distinguish vocal from non-vocal regions. As shown in Figure 2, the classifier consists of a front-end signal processor that converts waveform samples to cepstral-based feature vectors, followed by a backend statistical processor that performs modeling and matching. In modeling the acoustic characteristics of the vocal and non-vocal classes, two Gaussian mixture models (GMMs), λ_V and λ_N, are created using the respective feature vectors of the manually-segmented vocal and non-vocal parts of the music data collected beforehand. When an unknown song is received, the classifier takes as input the T-length feature vectors $\mathbf{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$ extracted from that song, and produces as output the frame likelihoods $p(\boldsymbol{x}_t|\lambda_V)$ and $p(\boldsymbol{x}_t|\lambda_N)$.

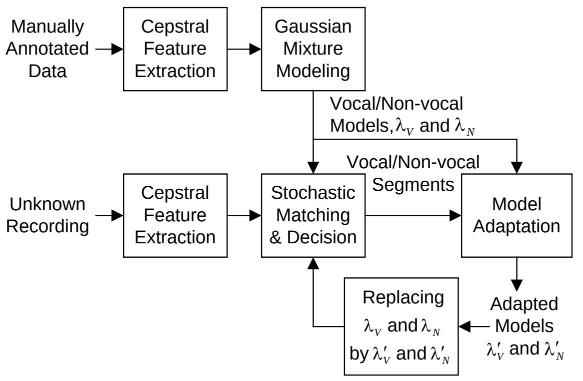

Manually Annotated Data
Cepstral Feature Extraction
Gaussian Mixture Modeling
Vocal/Non-vocal Models, λ_V and λ_N
Vocal/Non-vocal Segments
Unknown Recording
Cepstral Feature Extraction
Stochastic Matching & Decision
Model Adaptation
Replacing λ_V and λ_N by λ'_V and λ'_N
Adapted Models λ'_V and λ'_N

Figure 2. Vocal/non-vocal classification.

Since singing tends to continue for several frames, classification can be made in a segment-by-segment manner. Specifically, a W-length segment is classified as either vocal or non-vocal using

$$\sum_{i=1}^{W} \log p(\boldsymbol{x}_{sW+i} \mid \lambda_V) - \sum_{i=1}^{W} \log p(\boldsymbol{x}_{sW+i} \mid \lambda_N) \underset{\text{non-vocal}}{\overset{\text{vocal}}{\gtrless}} \eta, \qquad (1)$$

where s is the segment index. However, to avoid the risk that a large W might cross multiple vocal/non-vocal change boundaries, the classification is only valid for the segments where the classification results obtained with W and $W/2$ are consistent.

In addition, recognizing that the accuracy of classification crucially depends on the reliability of the vocal/non-vocal models, it seems necessary to use training data that exhaustively covers the vocal/non-vocal characteristics of various music styles. However, acquiring such a large amount of training data is usually cost prohibitive, since it requires considerable effort to manually label the music. To circumvent this problem, we tailor vocal/non-vocal models for each of the individual test music recordings, instead of designing models that can cover the universal vocal/non-vocal characteristics.

Similar to [17] and [18], the idea is to refine the vocal/non-vocal models by means of the classification results. It is assumed that the acoustic characteristics of the true vocal/non-vocal segments within each music recording can be inferred largely from the classified vocal/non-vocal segments. Thus, the classified segments can be used to refine the models, so that the classifier with the refined models then repeats the likelihood computation and decision-making, which should improve recognition. There are a number of ways to perform model refinement. This study uses a model adaptation technique based on maximum *a posteriori* estimation [19]. The procedure of classification and model adaptation is performed iteratively, until the resulting vocal/non-vocal boundaries do not change further. Finally, non-vocal segments longer than 2 seconds are located and removed from the recording.

4 MAIN MELODY EXTRACTION

4.1 Note sequence generation

Given a music recording, the aim of main melody extraction is to find the sequence of musical notes produced by the singing part of the recording. Let $e_1, e_2, \ldots, e_N$ be the inventory of possible notes performed by a singer. The task, therefore, is to determine which among N possible notes is most likely sung at each instant. To do this, the music signal is first divided into frames by using a fixed-length sliding window. Every frame is then convolved with a Hamming window and undergoes a fast Fourier transform (FFT) with size J. 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 spectral intensity in the frequency region where the F0 of the note is located.

Let $x_{t,j}$ denote the signal's energy with respect to FFT index j in frame t, where $1 \le j \le J$. If we use the MIDI note number to represent $e_1, e_2, \ldots, e_N$, and map the FFT indices into MIDI note numbers according to the F0 of each note, the signal's energy on note e_n in frame t can be estimated by

$$y_{t,n} = \underset{\forall j, U(j)=e_n}{\arg\max} \; x_{t,j}, \qquad (2)$$

and

$$U(j) = \left\lfloor 12 \cdot \log_2\left(\frac{F(j)}{440}\right) + 69.5 \right\rfloor, \qquad (3)$$

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, the resulting energy, $y_{t,n}$, should be the maximum among $y_{t,1}, y_{t,2}, \ldots, y_{t,N}$. However, due to the existence of harmonics, the note numbers that are several octaves higher than the sung note can also receive a large proportion of the signal's energy. Sometimes the energy on a harmonic note number can be even larger than the energy on the true sung note number; hence, the note number receiving the largest energy is not necessarily what is sung. To determine the sung note more reliably, this study adapts Sub-Harmonic Summation (SHS) [20] to this problem.

The principle applied here is to compute a value for the "strength" of each possible note by summing up the signal's energy on a note and its harmonic note numbers. Specifically, the strength of note e_n in frame t is computed using

$$z_{t,n} = \sum_{c=0}^{C} h^c \, y_{t,n+12c} \,, \tag{4}$$

where C is the number of harmonics that are taken into account, and h is a positive value less than 1 to discount the contribution of higher harmonics. The result of this summation is that the note number corresponding to the signal's F0 will receive the largest amount of energy from its harmonic notes. Thus, the sung note in frame t could be determined by choosing the note number associated with the largest value of the strength, i.e.,

$$o_t = \arg\max_{1 \le n \le N} \; z_{t,n} \,. \tag{5}$$

However, since most popular music contains background accompaniment during most or all vocal passages, the note number associated with the largest value of the strength may not be produced by a singer, but by the concurrent instruments instead. To alleviate the interference of the background accompaniment, we propose suppressing the strength pertaining to the notes that are likely produced by the instruments. The proposed method is motivated by an observation made in popular music that the principal accompaniments often contain a periodically-repeated note, compared to the vocals. Figure 3 shows an example of a fragment of a pop song, in which the tune is converted into a MIDI file. It is shown by software Cakewalk™ for ease of illustration. We can see from Figure 3 that the melody produced by the principal accompaniment tends to be repeated in the adjacent measures, compared to the main melody produced by singing. Therefore, it can be assumed that a note number associated with the constantly-large value of the strength within and across adjacent measures is likely produced by the instruments. In response to this assumption, we modify the computation of strength in Eq. (4) by

$$\tilde{z}_{t,n} = z_{t,n} - \frac{1}{2(L_2 - L_1 + 1)} \left(\sum_{l=-L_2}^{-L_1} z_{t+l,n} + \sum_{l=L_1}^{L_2} z_{t+l,n} \right), \tag{6}$$

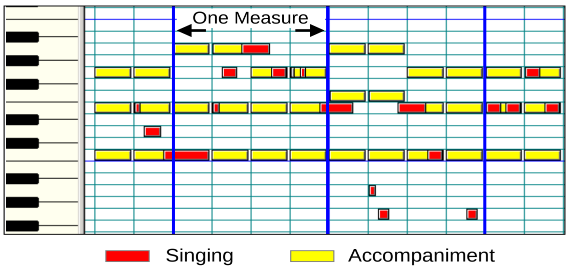


Figure 3. A fragment of the pop song "Let It Be" by *The Beatles*, in which the tune is converted manually into a MIDI file.

where L_1 and L_2 specify the regions $[t-L_2, t-L_1]$ and $[t+L_1, t+L_2]$, in which an average strength of note e_n is computed. Implicit in Eq. (6) is that the strength of note e_n in frame t will be largely suppressed, if the average strength of note e_n computed from the surrounding frames is large. Accordingly, the sung note in frame t is determined by

$$o_t = \arg\max_{1 \le n \le N} \; \tilde{z}_{t,n} \,. \tag{7}$$

4.2 Note sequence rectification

The above frame-based generation of note sequences may be improved by exploiting the underlying relation or constraints between frames. The most visible constraint between frames is that the length of a note is usually several times longer than a frame; hence, there should not be a drastic change like jitter between adjacent frames. To remove the jitters in a note sequence, we apply median filtering, which replaces each note of the frame with the local median of its neighboring frames.

In addition to the short-term constraint between adjacent frames, we further exploit a long-term constraint to rectify a note sequence. This constraint is based on the fact that the notes sung in a music recording usually vary far less than the range of all possible sung notes. Furthermore, the range of the notes sung within a verse or chorus section can be even narrower. Figure 4 shows a segment of a pop song, in which the singing part is converted into a MIDI file. It is clear that the range of the notes within the verse can be distinguished from that of the chorus, mainly because the sung notes within a section do not spread over all the possible notes, but are only distributed over their own narrower range. An informal survey using 50 pop songs shows that the range of sung notes within a whole song and within a verse or chorus section is around 24 and 22 semitones, respectively. Figure 5 details our statistic results. The range of sung notes serves as a long-term constraint to rectify a note sequence.

The basic idea of rectification is to locate incorrectly estimated notes that result in a note sequence beyond the normal range. Since the accompaniment is often played several octaves above or below the vocals, the incorrectly estimated notes are likely the octave of their true notes. Therefore, we may adjust some suspect notes by moving them several octaves up or down, so that the range of notes within an adjusted sequence conforms to the normal range. To be specific, let $\mathbf{o} = \{o_1, o_2, \ldots, o_T\}$ denote a note sequence estimated using Eq. (7). An adjusted note sequence $\mathbf{o}' = \{o'_1, o'_2, \ldots, o'_T\}$ is obtained by

$$o'_t = \begin{cases} o_t \,, & \text{if } |o_t - \bar{o}| \le (R/2) \\ o_t - 12 \times \left\lfloor \dfrac{o_t - \bar{o} + R/2}{12} \right\rfloor, & \text{if } o_t - \bar{o} > (R/2) \\ o_t - 12 \times \left\lfloor \dfrac{o_t - \bar{o} - R/2}{12} \right\rfloor, & \text{if } o_t - \bar{o} < (-R/2) \end{cases} \,, \tag{8}$$

where R is the normal range of the sung notes in a sequence, say 24, and $\bar{o}$ is the mean note computed by averaging all the notes in **o**. In Eq. (8), a note, o_t, is considered incorrect 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 performed by moving the incorrect note $\lfloor (o_t - \bar{o} + R/2)/12 \rfloor$ or $\lfloor (o_t - \bar{o} - R/2)/12 \rfloor$ octaves.

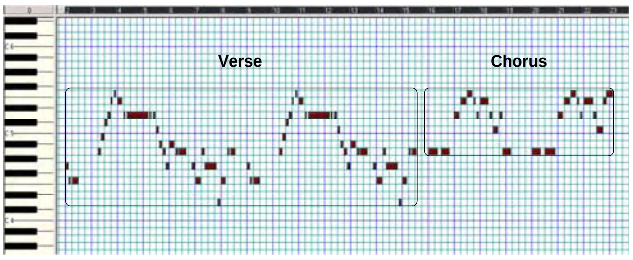
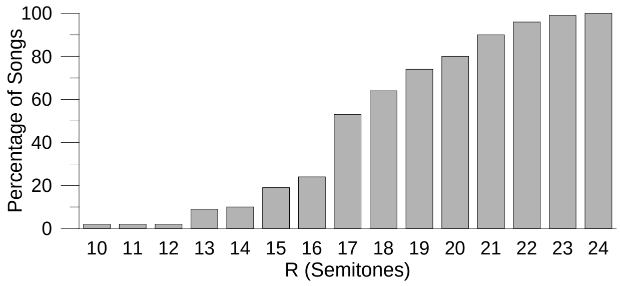
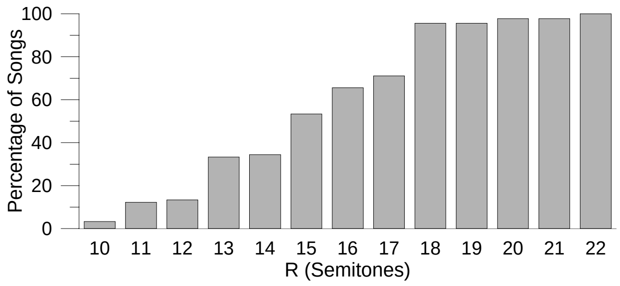


Figure 4. A fragment of the pop song "Yesterday" by *The Beatles*, in which the singing is converted into a MIDI file.



(a) The range of sung notes within a whole song.



(b) The range of sung notes within a verse or chorus.

Figure 5. Statistics of the range of sung notes in 50 pop songs, in which the percentage of songs whose range of sung notes less than R semitones is shown.

5 SIMILARITY COMPUTATION

After representing music data as a sequence of note numbers, cover-version retrieval can be converted into a problem of comparing the similarity between a query's sequence and each of the documents' sequences. Since cover versions are often different from the original song in terms of key, tempo, ornament, etc., it is virtually impossible to find a document sequence that matches the query sequence exactly. Moreover, main melody extraction is known to be frequently imperfect, which further introduces errors of substitution, deletion, and insertion into the note sequences. For reliable melody similarity comparison, an approximate matching method tolerable to occasional note errors is therefore needed.

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 document to be compared, respectively. The most apparent problem we face is that the lengths of **q** and **u** are usually unequal. Thus, it is necessary to temporally align **q** and **u** before computing their similarity. For this reason, we apply Dynamic Time Warping[2] (DTW) to find the mapping between each q_t and u_ℓ, $1 \le t \le T$, $1 \le \ell \le L$. DTW operates by constructing 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 $\{u_1, u_2,\ldots, u_\ell\}$. It is computed by

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

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 $\{u_1, u_2,\ldots, u_{\ell-1}\}$. The boundary conditions for the above recursion are defined by

$$\begin{cases} D(1,1) = d(1,1) \\ D(t,1) = \infty,\ 2 \le t \le T \\ D(2,2) = d(1,1) + d(2,2) - \varepsilon \\ D(2,3) = d(1,1) + d(2,2) \\ D(3,2) = d(1,1) + 2\times d(2,2) \\ D(t,2) = \infty,\ 4 \le t \le T \end{cases}. \tag{11}$$

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

$$S(\mathbf{q},\mathbf{u}) = \begin{cases} \max\limits_{T/2 \le \ell \le \min(2T,L)} [1/D(T,\ell)], & \text{if } L \ge T/2 \\ \infty, & \text{if } L < T/2 \end{cases}, \tag{12}$$

where we assume that the end of a query's sequence should be aligned to a certain frame between $T/2$ and $\min(2T,L)$ of the document's sequence, and assume that a document whose length of sequence is less than $T/2$ is not a relevant document to the query.

Since a song query may be performed 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. To deal with this problem, the dynamic range of a query's note sequence needs to be adjusted to that of the document to be compared. This can be done by moving the query's note sequence up or down several semitones, so that the mean of the note sequence is equal to that of the document to be compared. Briefly, a query's note sequence is adjusted by

$$q_t \leftarrow q_t + (\bar{u} - \bar{q}), \tag{13}$$

[2] Similar work can be found in [3,4,21].

where $\bar{q}$ and $\bar{u}$ are the means of the query's note sequence and the document's note sequence, respectively. However, our experiments find that the above adjustment can not fully overcome the transposition problem, since the value of ($\bar{q} - \bar{u}$) can only reflect a global difference of key between a query and a document, but cannot characterize the partial transposition or key change over the course of a query. To handle this problem better, we further modify the DTW similarity comparison by considering the key shifts of a query's note sequence. Specifically, a query sequence $\mathbf{q}$ is shifted with ±1, ±2,..., ±K semitones to span a set of note sequences $\{\mathbf{q}^{(1)}, \mathbf{q}^{(-1)}, \mathbf{q}^{(2)}, \mathbf{q}^{(-2)},\ldots, \mathbf{q}^{(K)}, \mathbf{q}^{(-K)}\}$. For a document sequence $\mathbf{u}$, the similarity $\mathbf{S}(\mathbf{q}, \mathbf{u})$ is then determined by choosing the one among $\{\mathbf{q}^{(0)}, \mathbf{q}^{(1)}, \mathbf{q}^{(-1)}, \mathbf{q}^{(2)}, \mathbf{q}^{(-2)}, \ldots, \mathbf{q}^{(K)}, \mathbf{q}^{(-K)}\}$ that is most similar to $\mathbf{u}$, i.e.,

$$S(\mathbf{q},\mathbf{u}) = \max_{-K\le k\le K} S(\mathbf{q}^{(k)},\mathbf{u}), \tag{14}$$

where $\mathbf{q}^{(0)} = \mathbf{q}$.

6 EXPERIMENTS

6.1 Music data

The music database used in this study consisted of 794 tracks[3] from pop music CDs, which mainly comprised five genres: soundtrack, country, folk, jazz, and rock. It was divided into three sub-sets. The first sub-set, denoted as DB-1, contained 47 pairs of tracks involving cover/original songs. In this sub-set, the difference between a cover version and the original song was characterized by the following factors: L: language (including English, Mandarin, and Japanese); S: singer; A: principal accompaniments; T: tempo; and N: non-vocal melodies. A summary of the characteristic differences within each pair of tracks is given in Table 1.

Table 1. A summary of the characteristic difference within each cover/original pair of tracks in sub-set DB-1.

Type of within-pair difference	No. of pairs
L	8
L + S	7
L + T	3
L + S + T	7
L + T + N	6
L + S + T + N	4
L + A + T + N	2
L + S + A + T + N	10

The second sub-set, denoted as DB-2, contained 500 tracks, none of which was a cover version of any track in DB-1. The third sub-set, denoted as DB-3, contained 200 tracks, performed by 13 female and 8 male singers, none of whom appeared in DB-1 and DB-2. The sub-sets DB-1 and DB-2 were used to evaluate the cover-version retrieval system, while DB-3 was used to create the vocal and non-vocal models. Manual annotation of vocal/non-vocal boundaries was only performed on DB-1 and DB-3. The waveform signals were down-sampled from a CD sampling rate of 44.1 kHz to 22.05 kHz, to exclude the high frequency components that usually contain sparse vocal information.

[3] The database did not contain the 50 pop songs used for analyzing the range of sung notes as described in Sec. 4.2.

6.2 Experimental results

Our first experiment was conducted using DB-1 as test data. It was run in a leave-one-out manner, which used one track at a time in DB-1 as a query trial to retrieve the remaining 93 tracks, and then rotated through all the 94 tracks. To roughly reflect a real-use scenario, each query was only a verse or chorus obtained with manual segmentation. The length of query ranged from 31 to 54 seconds. Performance of the song retrieval was evaluated on the basis of retrieval accuracy:

$$\frac{\#\,\text{queries whose target songs are ranked first}}{\#\,\text{queries}} \times 100\%.$$

We also computed the Top-N accuracy defined as the percentage of the queries whose target songs are among Top-N.

Table 2 shows the retrieval results for different configurations used in main melody extraction. In this experiment, each of the documents was a track with non-vocal segments removed manually. The inventory of possible sung notes consisted of the MIDI numbers from 41 to 83, which corresponds to the frequency range of 87 to 987 Hz. In FFT computation, the frame length and the overlap between frames were set to be 2048 and 1704, respectively. In addition, in melody similarity comparison, we used K = 2 in Eq. (14) to handle the transposition problem. We can see from Table 2 that the retrieval performance obtained by using Eq. (5) was the worst of the three methods compared, mainly because this method determines the sung notes based on the strength computed from the observed signal, which is vulnerable to the interference of background accompaniments. It is clear from Table 2 that a better estimation of the note strength can be obtained by using Eq. (7), which discounts the note numbers associated with the constantly-large values of the strength within and across adjacent measures. We can also see from Table 2 that melody extraction can be further improved by using the note sequence rectification of Eq. (8).

Table 3 shows the retrieval results for different configurations used in melody similarity comparison. In this experiment, main melody extraction was performed using the method of Eqs. (7) and (8) with R = 24, i.e., the best results shown in Table 2. We can see from Table 3 that the retrieval performance improves as the value of K increases. This indicates that the more the possible changes of key are taken into account, the greater the chance that a query's sequence will match the correct document's sequence. However, increasing the value of K substantially increases computational costs, because the similarity comparison requires two extra DTW operations whenever the value of K is increased by one. An economic value of $K = 2$ was thus chosen throughout our experiments.

Table 2. Performance of cover-version retrieval for different configurations used in main melody extraction, in which each method operates together with five-frame median filtering.

Main melody extraction method		Accuracy (%)		
		Top 1	Top 3	Top 10
Eq. (5)		60.64	71.28	78.72
Eq. (7) ($L_1 = 64$, and $L_2 = 192$)		70.21	73.40	80.85
Eqs. (7) and (8)	$R = 22$	65.96	72.34	74.47
	$R = 24$	76.60	78.72	87.23
	$R = 26$	74.47	77.66	86.17
	$R = 28$	70.21	77.66	85.11

Table 3. Performance of cover-version retrieval for different configurations used in melody similarity comparison.

Value of K in Eq. (14)	Accuracy (%)		
	Top 1	Top 3	Top 10
0	64.89	67.02	77.66
1	73.40	75.53	80.85
2	76.60	78.72	87.23
3	76.60	79.79	88.30

Next, we examined the performance of cover version retrieval based on the automatic removal of the non-vocal segments of each document. The number of Gaussian densities used in the vocal and non-vocal models was empirically determined to be 64. The length of segment, W, in Eq. (1) was set to be 200. Table 4 shows the experimental results, in which the results of "Manual removal" correspond to the results of "$K = 2$" in Table 3. We can see from Table 4 that although there is a significant performance gap between the manual and automatic removal of the non-vocal segments, the performance obtained with automatic non-vocal removal is much better than that obtained without non-vocal removal.

Experiments were further conducted to evaluate the retrieval performance of our system for a larger collection of songs. We used each of the 94 queries once at a time to retrieve the 593 tracks in DB-1 and DB-2. Since no manual annotation of vocal/non-vocal boundaries was performed on DB-2, the experiment was run on the basis of automatically removing the non-vocal segments of each document. Table 5 shows the experimental results. As expected, the increased number of non-target songs inevitably reduced the retrieval accuracy. By comparing Table 5 with Table 4, we can find that the retrieval accuracy deteriorates sharply when the system operates on a larger collection of songs without removing the non-vocal segments. Once again, this indicates the necessity of non-vocal region removal.

Figure 6 details the retrieval results for the 94 query trials, in which each point indicates the rank of each query's target song among the 593 documents. We can see from Figure 6 that almost all the target songs of queries belonging to "L" and "L + T" were ranked among the Top 3, whereas a large proportion of the target songs of queries belonging to "L + S + A + T + N" were ranked outside the Top 10. This reflects the fact that the greater the difference between the cover version and the original song, the more difficult it is to retrieve one song by using another song as a query. Although the overall performance leaves much room for further improvement, our system shows the feasibility of retrieving polyphonic cover recordings in a query-by-example framework.

Table 4. Performance of cover-version retrieval obtained with and without removing the non-vocal segments of each document.

Non-vocal segment removal method	Accuracy (%)		
	Top 1	Top 3	Top 10
Manual removal	76.60	78.72	87.23
Automatic removal	65.96	69.15	72.34
Without removal	54.26	59.57	64.89

Table 5. Results of cover-version retrieval for a collection of 594 tracks in DB-1 and DB-2.

Non-vocal segment removal method	Accuracy (%)		
	Top 1	Top 3	Top 10
Automatic removal	63.83	65.96	72.34
Without removal	47.87	54.26	60.64

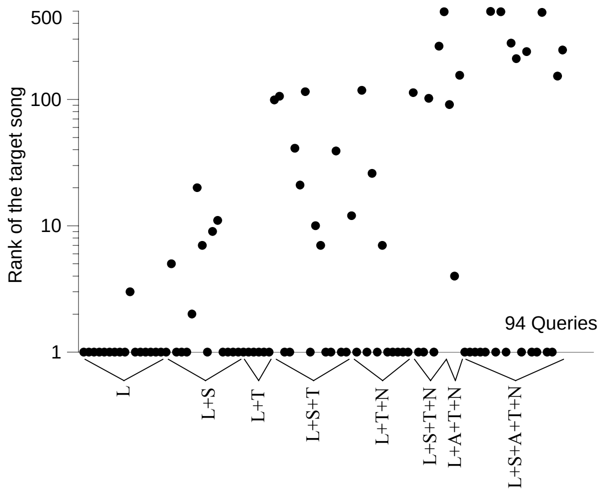


Figure 6. The ranks of the 94 queries' target songs.

7 CONCLUSIONS

In this study, we have examined the feasibility of retrieving cover versions of a song specified by a user. A query-by-example framework has been proposed to determine which among a collection of songs contain similar main melodies to a user's song query. In particular, to exclude factors that are irrelevant to the main melody of a song, we have proposed removing the non-vocal segments that are longer than a whole rest. In addition, to alleviate the interference of background accompaniments during the estimation of the sung note at each instant, we have proposed avoiding that a certain note number is regarded as a sung note if the strength of this note is continually large within and across adjacent measures. We have also proposed correcting the estimated sung note sequence by limiting the range of sung notes in a sequence to 24 semitones. Furthermore, we have studied the method of comparing the similarities between a query's note sequence and each of the documents' note sequences. This method has proven capable of handling the discrepancies in tempo and transposition between cover versions and the original songs.

Despite their potential, the methods proposed in this study can only be baseline solutions to the cover-version retrieval problem. Analogous to other research on retrieving polyphonic documents based on polyphonic queries, more work is still needed to improve melody extraction and melody similarity comparison. In addition, to further explore the cover-version retrieval problem, the essential work is to scale up the music database, which covers a wider variety of music styles, genres, singers, languages, and so on.

8 ACKNOWLEDGEMENT

This work was supported in part by the National Science Council, Taiwan, under Grant NSC93-2422-H-001-0004.

REFERENCES

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

[2] Kosugi, N., Nishihara, T. Sakata, S., Yamamuro, M., and Kushima, K. "A practical query-by-huming system for a large music database", Proceedings of the ACM Conference on Multimedia, 2000.

[3] Hu, N. and Dannenberg, R. B. "A comparison of melodic database retrieval techniques using sung queries", Proceedings of the ACM/IEEE-CS Joint Conference on Digital Libraries, 2002.

[4] Pauws, S. "CubyHum: A fully operational query by humming system", Proceedings of the International Conference on Music Information Retrieval, 2002.

[5] Nishimura, T., Hashiguchi, H. Takita, J. Zhang, J. X., Goto, M., and Oka, R. "Music signal spotting retrieval by a humming query using start frame feature dependent continuous dynamic programming", Proceedings of the International Symposium on Music Information Retrieval, 2001.

[6] Song, J., Bae, S. Y., Yoon, K. "Mid-level music melody representation of polyphonic audio for query-by-humming system", Proceedings of the International Conference on Music Information Retrieval, 2002.

[7] Doraisamy, S., and Ruger, S. M. "An approach towards a polyphonic music retrieval system", Proceedings of the International Symposium on Music Information Retrieval, 2001.

[8] Pickens, J., Bello, J. P., Monti, G., Crawford, T., Dovey, M., Sandler, M. and Byrd, D. "Polyphonic score retrieval using polyphonic audio queries: A harmonic modelling approach", Proceedings of the International Conference on Music Information Retrieval, 2002.

[9] Foote, J. "ARTHUR: Retrieving orchestral music by long-term structure", Proceedings of the International Symposium on Music Information Retrieval, 2000.

[10] Tsai, W. H., and Wang, H. M. "A query-by-example framework to retrieve music documents by singer", Proceedings of the IEEE Conference on Multimedia and Expo, 2004.

[11] Tzanetakis, G., and Cook, P. Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10 (5), 2002.

[12] Feng, Y., Zhuang, Y., and Pan, Y. "Popular music retrieval by detecting mood", Proceedings of the ACM Conference on Research and Development in Information Retrieval, 2003.

[13] Egglink, J., and Brown, G. J. "Extracting melody lines from complex audio", Proceedings of the International Conference on Music Information Retrieval, 2004.

[14] Meek, C., and Birmingham, W. P. Automatic thematic extractor. Journal of Intelligent Information Systems, 21 (1), 2003, 9–33.

[15] Typke, R., Veltkamp, R. C., and Wiering, F. "Searching notated polyphonic music using transportation distances", Proceedings of the ACM Conference on Multimedia, 2004.

[16] Tsai, W. H., Wang, H. M., and Rodgers D. Blind clustering of popular music recordings based on singer voice characteristics. Computer Music Journal, 28 (3), 2004, 68–78.

[17] Nwe, T. L., and Wang, Y. "Automatic detection of vocal segments in popular songs", Proceedings of the International Conference on Music Information Retrieval, 2004.

[18] Tzanetakis, G. "Song-specific bootstrapping of singing voice structure", Proceedings of the IEEE Conference on Multimedia and Expo, 2004.

[19] Reynolds, D. A., Quatieri, T. F., and Dunn, R. B. Speaker verification using adapted Gaussian mixture models. Digital Signal Processing, 10, 2000.

[20] Piszczalski, M., and Galler, B. A. "Predicting musical pitch from component frequency ratios", Journal of the Acoustical Society of America, 66(3), 1979, 710–720.

[21] Pardo, B., Birmingham, W. P., and Shifrin, J. Name that tune: a pilot study in finding a melody from a sung query. Journal of the American Society for Information Science and Technology 55(4), 2004.