Query by Tapping: A New Paradigm for Content-Based Music Retrieval from Acoustic Input

Jyh-Shing Roger Jang[1], Hong-Ru Lee[2], Chia-Hui Yeh[3]

Multimedia Information Retrieval Laboratory
Computer Science Department, National Tsing Hua University, Taiwan
jang@cs.nthu.edu.tw[1], khair@ms24.hinet.net[2]
beball@wayne.cs.nthu.edu.tw[3]

Abstract. This paper presents a query by tapping" system, which represents a new paradigm for CBMRAI (content-based music retrieval via acoustic input) systems. Most CBMRAI systems take the user acoustic input in the format of singing or humming, and the timing or beat information (durations of notes) is sometimes discarded during the retrieval process in order to save computation. Our query by tapping" mechanism, on the other hand, takes the user input in the format of tapping on the microphone and the extracted duration of notes is then used to retrieve the intended song in the database. Since there is no singing or humming, no pitch information is used in the retrieval process at all. Most people would think that it is hard to do music retrieval via beat information alone. However, our experiments demonstrate that beat information is also an effective feature in the sense that it can be used to retrieve the intended song from a large collection of music database with a satisfactory recognition rate.

1 Introduction

As there are more and more digital music files (such as MP3, MIDI, ASF, RM) being created over the Internet, the corresponding issue of music information retrieval [1][2] is becoming increasingly important. In particular, most people do not have professional music skills and the best way to specify an intended song is to sing or hum it. As a result, CBMRAI (content-based music retrieval via acoustic input) systems are the most natural tools for common people needs of music information retrieval. Application domains of CBMRAI are immense, such as

1. Internet music search engine
2. Query engine for digital music libraries/museums
3. Intelligent interface for karaoke bars (which are quite popular in China, Japan, Korea and Taiwan)
4. Song-activated interactive toys
5. Education software for music/vocal training

Most CBMRAI systems [3][13][12][11][14][4] take the user input in the format of singing or humming, and then transform the audio signal into a pitch vector for retrieval purpose. To save computation time, the pitch duration (or equivalent, beat or timing information) is sometimes discarded to have a more compact representation and a shorter computation time.

Unlike most CBMRAI systems, our music retrieval system, called Super MBox" [7][9][8], has a unique feature that can take the user input in the format of tapping on the microphone, which represents the beat or timing information of the intended song. The system then uses the beat information alone to identify the intended songs from a database of 11744 candidate songs. Most people would think that CBMRAI based on beat information alone can not have good performance since amateur persons cannot identify a song solely based on its beat information. However, our experiments indicate otherwise. In fact, we have applied a dynamic-programming-based comparison procedure to achieve a top-100 (or top 0.85% of all candidate songs) recognition rate of about 80%, which is considered satisfactory performance. We also did error analysis to demonstrate the causes of cases of failure.

The rest of the paper is organized as follows. Section 2 explains the feature extraction process, which transforms the user's tapping input into a timing vector. Section 3 describes the dynamic-programming-based procedure that we employ to compare the input timing vector with those of the songs in the database. Section 4 demonstrates the performance evaluation of the system and explores the causes that might degrade the performance. Section 5 gives conclusions and future directions.

2 Feature Vector Extraction

The feature vector used in our system is simply a time vector in which each element represents the duration of a note. To extract such information, the user is required to gently tap on the microphone to indicate the beat information of the intended song. The recording conditions are

- Sample rate of 11025
- 8-bit resolution
- Recording duration of 15 seconds
- Single channel (mono)

A typical waveform of the user's tapping input of the song "You Are My Sunshine" is shown in plot (a) of the following Fig. 1; the corresponding log energy profile is shown in plot (b). (The plots only show the first 10 seconds.)

From plot (a) of the below figure, it is obvious that the user has input 16 complete notes. To extract the duration of each note, we need to do frame blocking first and then find the energy of each frame. The circles in (b) indicate where local maxima of the log energy are located. The local maxima are legal only when their values are greater than a heuristically determined threshold (about –20 dB, indicated by the horizontal line in plot (b)).

Once the legal local maxima are found, the duration of each note is equal to the distance between two neighboring local maxima. The beat information is then represented as a timing vector in which each element is a note's duration. For example, the timing vector extracted from the above example would be [0.4180, 0.3715, 0.3831, 0.7314, 1.1262, 0.4063, 0.3599, 0.3715, 0.7779, 1.1610, 0.3947, 0.3483, 0.3715, 0.8127, 1.0797, 0.3715], in where there are 16 elements, representing durations of 16 notes in seconds.

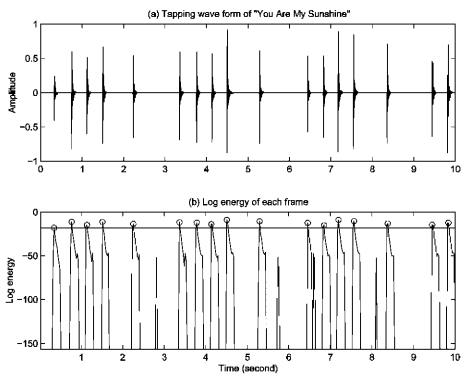

Fig. 1. (a) Tapping waveform of "You Are My Sunshine" (b) Corresponding log energy plot.

The parameters of the feature extraction process can be listed as follows:

1. The frame size is 256 points.
2. The threshold for legal local maxima is equal to the fifth global maximum of log energy minus 7 dB.

Note that the frame size determines the resolution of the resultant timing information. In our case, the resolution is equal to 256/11025 = 0.0232 seconds, which is good enough for our comparison procedure. If the frame size is smaller, the timing resolution is higher, but the energy profile becomes jagged and the exact location of the desired local maxima would be difficult to find.

3 Comparison Procedure

Once the timing vector from a user's input is obtained, we need to compare it with those of the songs in the database. In this section we shall propose a comparison procedure based dynamic programming [15]. In fact, the concept is similar to dynamic time warping [6] used in our previous work [7]. In this paper, for simplicity, we assume that the user always taps from the beginning of the intended song. However, this is not an absolute requirement since the proposed method can still start matching at anywhere in a middle of a song.

To match the input timing vector against those of the songs in the database, we need to be aware of two things:

1. The tempo of the user's input is usually different from those of the candidate songs in the database.
2. The user is likely to lose notes instead of gaining notes.

To solve the first problem, we need to normalize the input timing vector and those of the candidate songs. Suppose that the input timing vector (or test vector) is represented by vector t of length m, and the reference timing vector (reference vector) by r with length n. Usually n is greater than m. Suppose that the user did not lose/gain any notes, so we only need to compare t with the first m elements of r. However, since the user might lose or gain notes, we need to compare t with a selection of different versions of r with different lengths. Suppose that the first q elements of r is selected for comparison, then the normalization step convert both vectors to a total duration of 1000:

$$\begin{cases} \tilde{t} = round(1000 * t / sum(t)) \\ \tilde{r} = round(1000 * r(1:q) / sum(r(1:q))) \end{cases} \tag{1}$$

In the above equations, $r(1:q)$ indicates a vector formed from the first q elements of vector r; $sum(t)$ indicates the summation of all elements in vector t. The operation of *round* rounds all elements of the vectors into integers. The purpose of multiplication by 1000 is to increase the precision since our comparison procedure is based on integer operations to save computation time. After the above normalization step, the subsequent comparison procedure is based on $\tilde{t}$ and $\tilde{r}$. Actually, we need to vary the value of q to get different versions of $\tilde{r}$; the distance between $\tilde{t}$ and $\tilde{r}$ is taken to be the minimum of distances between $\tilde{t}$ and all variants of $\tilde{r}$. In our system, the value of q is varied from $p-2$ to $p+2$, where p is the length of $\tilde{t}$.

The comparison procedure is based on the concept of dynamic time warping (DTW) [6]. For notation simplicity, we shall remove the "tilde" temporarily. Suppose that the (normalized) input timing vector (or test vector) is represented by $t(i), i = 1,\ldots,m,$ and the (normalized) reference timing vector (reference vector) by $r(j), j = 1,\ldots,n$. These two vectors are not necessarily of the same size and we can apply DTW to match each point of the test vector to that of the reference vector in an optimal way. That is, we want to construct a $(m+1)\times(n+1)$ DTW table $D(i,j)$ according to the following forward dynamic programming algorithm:

Optimal value function:

$D(i,j)$ is the minimum accumulated distance starting from **(0, 0)** of the DTW table to the current position (i,j).

Recurrence relation:

$$D(i,j) = \min \begin{cases} D(i-1,j-2) + |r(i-1) + r(i) - t(j)| + \eta_1 \\ D(i-1,j-1) + |t(i) - r(j)| \\ D(i-2,j-1) + |t(i-1) + t(i) - r(j)| + \eta_2 \end{cases} \tag{2}$$

where η_1 and η_2 are a small positive numbers. In the above equation, $|r(i-1) + r(i) - t(j)| + \eta_1$ represents the cost when the user mistake two note for one; $|t(i-1) + t(i) - r(j)| + \eta_2$ represents the cost when the user mistake a note for two.

Boundary conditions:

The boundary conditions for the above recursion can be expressed as

$$\begin{aligned} D(i,0) &= 0, i = 0,\ldots,m, \\ D(0,j) &= 0, j = 0,\ldots,n. \end{aligned} \tag{3}$$

After we have the boundary conditions, the recursion formula to fill the DTW table can be performed either row-wise or column-wise.

The cost of the optimal DTW path is defined as $D(m,n)$

After finding $D(m,n)$, we can back track to obtain the entire optimal DTW path and the corresponding alignments between the two vectors.

In summary, the whole comparison procedure can be listed as follows:

1. Extract the input timing vector from the user's acoustic input of tapping. Normalized the timing vector.
2. Find the DTW distance between the input timing vector and that of each candidate song in the database. Note that the timing vector of a song has to be compressed or extended to have 5 versions of different lengths. Then the distance between the input timing vector and that of a song is taken to be the minimum among all 5 distances between the input timing vector and 5 variants of that of the song.
3. List the results according to DTW distances.

Fig. 2 illustrates the flowchart of our query-by-tapping system.

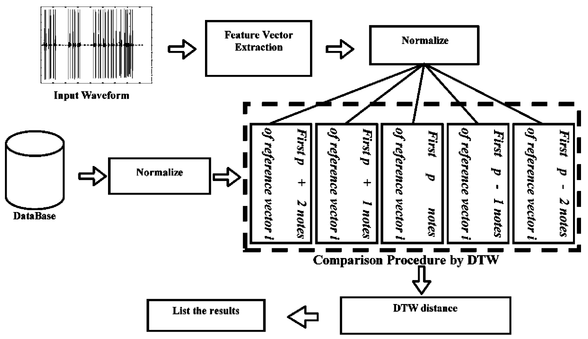
Input Waveform
Feature Vector Extraction
Normalize
DataBase
Normalize
First p + 2 notes of reference vector i
First p + 1 notes of reference vector i
First p notes of reference vector i
First p - 1 notes of reference vector i
First p - 2 notes of reference vector i
Comparison Procedure by DTW
DTW distance
List the results

Fig. 2. Flowchart of our query-by-tapping system

4 Performance Evaluation

In this section we present the performance evaluation of our query-by-tapping system. We have a dataset of around 269 clips of tapping from by 9 persons (7 males, 2 females). Specs of the dataset are:

●No. of clips: 269	●Resolution: 8 bits
●Duration: 15 seconds	●Type: Single channel (mono)
●Sample rate: 11025 samples/second	●Start position: Beginning of a song

All of the recordings start from the beginning of a song. The specs of our platform and candidate songs are

●CPU: Pentium-III 800MHz	●No. of candidate songs: 11744
●RAM: 128 MB	●Match position: Beginning of each song

In the first experiment, we take all the tapping clips of 15 seconds to evaluate our system. The average response time of a query with 15-second tapping clip is about

3.42 seconds. In average, a 15-second tapping clip contains about 29.98 notes. The performance of the system can be viewed from the following pie chart:

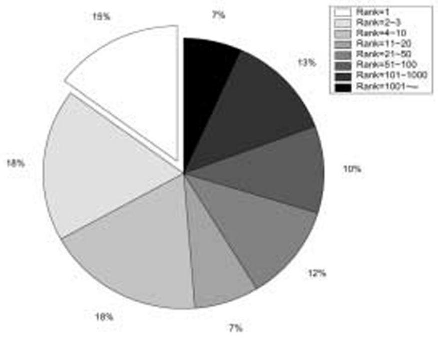

Fig. 3. The performance of our system

From Fig. 3, the top-1 recognition rate (percentage of recordings that Super MBox can find the intended songs in the top-1 ranking) is 15%, the top-10 recognition rate is 51%, and the top-100 recognition rate is 80%. Apparently it seems that these recognition rates are not so impressive. However, remember we are only using timing information to do the query, and there are 11744 candidate songs in the database (so top-100 is equivalent to top 0.85% of the whole candidate songs).

For those songs that fall below top-100, we performed an error analysis and found that most of the errors could be attributed to the following reasons:

1. Some users can only tap the songs correctly at the beginning but could not keep it through the whole recording duration of 15 seconds.
2. Some of the tapping clips are too noisy to extract the correct timing information.
3. Some MIDI files do not correctly represent the rendition of the original songs. This is mostly due to the fact that the composers of the MIDI files tried to add personal styles and made the rendition different from the original ones.
4. Some MIDI files include preludes. Since we match from the beginning of a song; the prelude will be matched incorrectly.

Reasons 3 and 4 are MIDI errors and can only be corrected manually. This MIDI correction task is very labor intensive and time consuming. What is worse is that the task can only be accomplished by someone familiar with the songs. Obviously it is impossible to find a person who knows all 11744 songs. Hence MIDI correction is a long-term task that plays a pivotal role to improve our system performance incrementally.

Reason 2 typically occurs when the user microphone has an abnormal high/low gain, or the sound card has a wrong driver. On the other hand, we can also improve our feature extraction procedure to make it more robust to adversary recording conditions.

Reason 1 demonstrates an interesting fact that some people cannot tap a song through 15 seconds. The reasons are two fold. First, the user might be familiar with only the first 10 seconds of the song, but not the whole 15 seconds. Second, the user might not be able to keep the same tempo through out 15 seconds. To have an in-depth evaluation, we perform an experiment that evaluation the system performance on the duration of tapping clips. Fig. 4 shows the results of recognition rates with respect to recording duration.

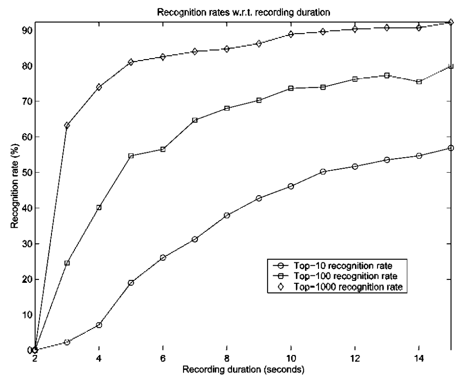

Fig. 4. Recognition rates with respect to recording duration

The above plot shows the curves of top-10 (top-0.085%), top-100 (top-0.85%), and top-1000 (top-8.5%) recognition rates with respect to tapping durations. It is observed that top-10 curve goes up with the increase of recording duration. On the other hand, top-100 and top-1000 curves level off at about 10 seconds. In particular, for the top-100 curve, the recognition rate for 14 seconds is even lower than those of 12 and 13 seconds. This indicates that a longer tapping duration does not imply a better recognition rate, which comes from the observation that some users can only tap the songs correctly at the beginning but could not keep it through the whole recording duration of 15 seconds.

5 Conclusions and Future Work

In this paper, we have presented a new paradigm for content-based music retrieval system via acoustic input. The new retrieval paradigm, called query by tapping", allows the user to query a music database by simply tapping on the microphone to input the durations of the first several notes of the intended song. Most people would think this retrieval paradigm is ineffective for large-scale music database. However, our experiments demonstrate that query by tapping" is not only interesting but also effective way of retrieving music from a large-scale music database. The top-10 and top-100 recognition rates of query by tapping" on a database of 11744 songs are about 51% and 80%, respectively. Moreover, we found that long tapping clips do not always lead to better performance due to the facts that most people cannot keep correct tapping or consistent tempo over the whole 15 seconds of recording.

We have implemented query by tapping" as a query method besides query by humming" and query by singing" to our content-based music retrieval system called

Super MBox". The online version of the system can be downloaded from the link n-line demo of Super MBox" at the author homepage at

http://www.cs.nthu.edu.tw/~jang

This is on-going research and our goal is to make Super MBox an intelligent content-based music retrieval system that allows as many different ways of music retrieval as possible. Future work involves the following tasks:

1. Allow the user to start tapping from the middle of a song.
2. Modify the feature extraction procedure to make it more robust to noisy environment.
3. Combine query by tapping" and query by humming" to have a multi-modal user interface for better performance.

6 References

[1] 1st International Symposium on Music Information Retrieval (MUSIC IR 2000), Plymouth, Massachusetts, Oct. 23-25, 2000. (http://ciir.cs.umass.edu/music2000/)

[2] 2nd Annual International Symposium on Music Information Retrieval (MUSIC IR 2001), Indiana University, Bloomington,Indiana, Oct.15-17, 2001.(http://ismir2001.indiana.edu/)

[3] A. J. Ghias, and D. Logan, B. C. Chamberlain, Smith, query by humming-musical information retrieval in an audio database", ACM Multimedia '95 San Francisco, 1995.

[4] A. Uitdenbogerd, and J. Zobel, "Melodic Matching Techniques for Large Music Databases", (http://www.kom.e-technik.tu-darmstadt.de/acmmm99/ep/uitdenbogerd/)

[5] B. Chen, and J.-S. Roger Jang, "Query by Singing", 11th IPPR Conference on Computer Vision, Graphics, and Image Processing, PP. 529-536, Taiwan, Aug 1998.

[6] J. R. J. G. Proakis, and J. H. L. Hansen, "Discrete-time processing of speech signals," New York, Macmillan Pub. Co., 1993.

[7] J.-S. Roger Jang, and Ming-Yang Gao, "A Query-by-Singing System based on Dynamic Programming", International Workshop on Intelligent Systms Resolutions (the 8th Bellman Continuum), PP. 85-89, Hsinchu, Taiwan, Dec2000.

[8] J.-S. Roger Jang, Hong-Ru Lee, and Ming-Yang Kao, "Content-based Music Retrieval Using Linear Scaling and Branch-and-bound Tree Search", IEEE International Conference on Multimedia and Expo, Waseda University, Tokyo, Japan, August 2001. (Submitted)

[9] J.-S. Roger Jang and Hong-Ru Lee, "Hierarchical Filtering Method for Content-based Music Retrieval via Acoustic Input", The ninth ACM Multimedia Conference, Ottawa, Ontario, Canada, September 2001.

[10] N. Y. Kosugi, Kon'ya, S. Nishihara, M. Yamamura, and K. Kushima, "Music Retrieval by Humming – Using Similarity Retrieval over High Dimensional Feature Vector Space," pp 404-407, IEEE 1999.

[11] R. J. McNab, and L. A. Smith, Melody transcription for interactive applications" *Department of Computer Science University of Waikato*, New Zealand.

[12] R. J. McNab, L. A. Smith, and Jan H. Witten, Signal Processing for Melody Transcription" *Proceedings of the 19th Australasian Computer Science Conference*, 1996.

[13] R. J. McNab, L. A. Smith, and Jan H. Witten, Towards the Digital Music Library: Tune Retrieval from Acoustic Input" ACM, 1996.

[14] R. J. McNab, L. A. Smith, I. H. Witten, and C. L. Henderson, "Tune Retrieval in the Multimedia Library,"

[15] Stuart E. Dreyfus, and Averill M. Law, The art and theory of dynamic programming", New York :Academic Press, 1977.