Journal of Science and Engineering Technology, Vol. 3, No. 1, pp. 49-58 (2007)

Using an N-gram-Based Mapping Approach to Content-Based Music Information Retrieval

CHUEH-CHIH LIU[1], TE-WEI CHIANG[2] and TIENWEI TSAI[3]
1 Library & Information Center, Chihlee Institute of Technology
2 Department of Accounting Information Systems, Chihlee Institute of Technology
3 Department of Information Management, Chihlee Institute of Technology
313, Sec. 1, Wunhua Rd., Banciao, Taipei County, Taiwan

ABSTRACT

Studies on query-by-humming (QBH) have recently become increasingly popular. On the basis of the fact that MP3 and MIDI formats have the advantages of small storage space and high audio quality, thereby making them suitable for Internet applications, we propose a novel approach based on QBH to retrieve information from MP3 and MIDI formats. In the database establishment phase, each music file is first partitioned into a set of musical data objects encoded via mapping function, including bi-gram, tri-gram and four-gram approaches, to establish a music database. In the retrieval phase, the file most similar to the musical test segment can be retrieved from the music database. Experiments were conducted to demonstrate the effectiveness of our approach.
***Key Words*:** query-by-humming, content-based retrieval, music information retrieval, nusic databases

基於 N-gram 映射函數之內涵式音樂檢索法

劉爵至[1]　蔣德威[2]　蔡殿偉[3]
[1]致理技術學院圖資中心
[2]致理技術學院會計資訊系
[3]致理技術學院資訊管理系
台北縣板橋市文化路一段 313 號

摘　要

近幾年來，哼唱式查詢的研究一直受到大家的矚目。事實上，MP3 和 MIDI 音樂格式都具有檔案小和音樂品質高的優點，目前在網際網路上被廣泛地使用。本論文中，我們提出了一個利用哼唱式查詢的方法去進行 MP3 和 MIDI 格式的音樂檢索。在音樂資料庫（music database）的建置階段，資料庫中的每首音樂會先經過斷句（segmentation）的處理，形成一段一段的樂句（phases），再透過映射函數（mapping functions），分別針對二個音符（bi-gram）、三個音符（tri-gram）和四個音符（four-gram）進行編碼，作爲檢索的依據。在樂句的檢索階段，和測試樂句最相似的音樂會從音樂資料庫中被檢索出來。我們進行了一系列的實驗，結果證明我們的方法有很好的成效。
關鍵詞：哼唱式查詢，內涵式檢索，音樂資訊檢索，音樂資料庫

I. INTRODUCTION

In traditional query systems, data are retrieved via the names of the data (e.g., file names) or the keyword(s) of the data. For example, many portal websites allow users to enter the keyword(s) of the desired data. Although this kind of query is convenient, it does not guarantee to obtain the information we really want, especially for the searching of multimedia data such as image, audio, or video data. For instance, when we want to search for an image with a boat under the sunset by the side of a river, the keyword-based query is useless at all. On the contrary, the content-based retrieval (CBR) [13] method can be applied to retrieve such information. In that way, we can retrieve the image with a boat, sunset and river by using sunset or boat images. Other media such as audio can be retrieved in a similar way. For general users, one of the best ways to query a song is humming a section of the song. The query that uses a section of song to find the song containing the section from a music database is called Query by Humming (QBH) [4, 7, 9, 10, 14], which is the very problem we want to tackle in this paper .

Typically, two distinct types of music file formats, WAV and MIDI (musical instrument digital interface), are widely used. The WAV file records the real musical data while the MIDI file only records the simplified musical data, i.e., musical notations (such as Do, Re, Mi, etc.) and the durations between two consecutive musical notations. The WAV file requires much more memory space than the MIDI file does. Therefore, the MIDI file is more suitable for the Web application. Apart from the WAV and MIDI formats, a new compressed format, MPEG Audio Layer-3 (MP3) [11, 12], has been developed. MP3 format has the advantage of small space requirement due to its high compression ratio, along with the high music quality almost competing with that of CD music. It is believed that it will become the mainstream of the music format in the near future.

In this paper, we propose a novel approach for CBR in MP3 and MIDI formats. In the database establishing phase, each music file is first partitioned into a set of music data objects. Then, these data objects are encoded via a mapping function, which includes approaches of bi-gram, tri-gram and four-gram, to establish a music database. In the retrieving phase, the music that is most similar to the test music segment can be retrieved from the music database based on the Euclidean distance.

II. SYSTEM ARCHITECTURE

In this section, we will explain the system architecture of our approach, including how the music characteristic data are produced, how the music object is separated, and how the music content-based retrieval is conducted.

Figure 1 illustrates the procedures of establishing the music database. All music objects in the database must pass through the segmentation and truncation module to obtain the music phrases. Each song is segmented into a sequence of phrases according to the artists' pause [8]. In general, a song is segmented into about twenty to thirty phrases. After the feature extraction process, the music phrases together with their features are stored in the database. On the other hand, after users' humming passes through the feature extraction module, the similarity measuring module will compare the query object with those objects in the database, as shown in Figure 2. Then, the system outputs a list of targets ranked according to their scores. In this paper, the Euclidean distance is applied to evaluate their similarity.

1. Feature Extraction

Pitch tracking is a widely used method to extract features from a music object. In our approach, the auto-correlation approach with center clipping [4] is applied to pitch tracking. After the pitch contour of a phrase is obtained, the contour is further converted into a note sequence. Its underling concept is to transform the features of a music phrase into a vector. Since the partition policy of a song is not the main issue of this paper, we simply use Cool Edit Pro 2.0 to segment a MP3 song into phrases according to the artists' pause. In this way, each song is segmented into about twenty to thirty phrases. On the other hand, we used Cakewalk Pro Audio 9 to segment the MIDI objects. The segmentation is based on the artists' pause as well.

2. Vector Transformation

In this paper, we propose two vector transformation methods. The first method uses the mapping function [2, 3] based on the N-gram information to encode relative notes. The second method is also based on the relative notes; however, it uses the differences between the current musical notes and

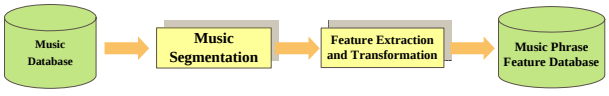

Fig. 1. The procedures of establishing the music database

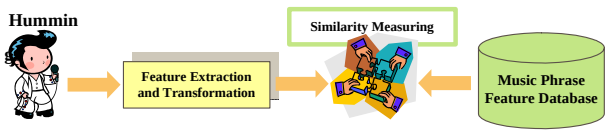

Fig. 2. Content-based music retrieval

their preceding ones to form a vector. The details of the two methods will be introduced in the next section.

III. THE PROPOSED METHODS

1. Method I: N-gram and Baseline Mapping Functions

Relative note is defined as the pitch change between the current musical note and its preceding one. For example, if the order of the musical notes for a song is 79, 76, 76, 77, 74, 74, 72, 74, 76, 77, 79, 79, 79, the order of relative notes can be derived as 79 > 76 = 76 < 77 > 74 = 74 > 72 < 74 <76 < 77 < 79 = 79 = 79.

The N-gram approach considers the relationship among N contiguous musical notes. The baseline mapping functions for bi-gram, tri-gram, and four-gram are shown in Tables 1, 2, and 3, respectively. For the bi-gram case, 79 > 76 = 76 < 77 > 74 = 74 > 72 < 74 <76 < 77 < 79 = 79 = 79 is encoded to 3, 2, 1, 3, 2, 3, 1, 1, 1, 1, 2, 2 according to Table 1, and the code sequence finally forms a 12-dimensional vector. As for tri-gram, according to Table 2, an 11-dimensional vector (8, 4, 3, 8, 6, 7, 1, 1, 1, 2, 5) is generated. For four-gram, a 10-dimensional vector is generated according to Table 3 in a similar way.

Table 1. Baseline mapping function based on bi-gram

Note relationship	Value
$N_i < N_{i+1}$	1
$N_i = N_{i+1}$	2
$N_i > N_{i+1}$	3

Table 2. Baseline mapping function based on tri-gram

Note relationship	Value
$N_i < N_{i+1} < N_{i+2}$	1
$N_i < N_{i+1} = N_{i+2}$	2
$N_i < N_{i+1} > N_{i+2}$	3
$N_i = N_{i+1} < N_{i+2}$	4
$N_i = N_{i+1} = N_{i+2}$	5
$N_i = N_{i+1} > N_{i+2}$	6
$N_i > N_{i+1} < N_{i+2}$	7
$N_i > N_{i+1} = N_{i+2}$	8
$N_i > N_{i+1} > N_{i+2}$	9

The singers' humming and the music phrases in the database are encoded in the same way. We can evaluate the similarity between two vectors via the Euclidean distance as follows:

$$dist(MF(\vec{q}_i), MF(\vec{d}_j)) =$$

$$\sqrt{\sum_{m=1}^{N}(MF(\vec{q}_i)_m - MF(\vec{d}_j)_m)^2},$$

$$when\ \dim(MF(\vec{q}_i)) = \dim(MF(\vec{d}_j)),$$

$$Min_{p=0}^{\dim(MF(\vec{q}_i))-\dim(MF(\vec{d}_j))}\sqrt{\sum_{m=1}^{N}(MF(\vec{q}_i)_{m+p} - MF(\vec{d}_j)_m)^2},$$

$$when\ \dim(MF(\vec{q}_i)) > \dim(MF(\vec{d}_j)),\ and.$$

$$Min_{p=0}^{\dim(MF(\vec{d}_j))-\dim(MF(\vec{q}_i))}\sqrt{\sum_{m=1}^{N}(MF(\vec{q}_i)_m - MF(\vec{d}_j)_{m+p})^2}$$

$$otherwise. \qquad (1)$$

The notations used in Eq.(1) are defined as follows:

- N: the smaller dimension between $MF(\vec{q}_i)$ and $MF(\vec{d}_j)$,
- $\vec{q}_i$: the vector of query i by a singer,
- $\vec{d}_j$: the vector of phrase j in the music phrase characteristic database,
- p: the beginning offset of the sliding windows for the longer vector,
- MF($\vec{v}$): the mapping function for the vector $\vec{v}$, and
- m: the m-th dimension of the vector transformed by the mapping function.

We first calculate the distance for bi-gram, tri-gram, and four-gram using Eq. (1). Then, the total dissimilarity can be calculated using Eq. (2).

Table 3. Baseline mapping function based on four-gram

Note relationship	Value	Note relationship	Value	Note relationship	Value
$N_i < N_{i+1} < N_{i+2} < N_{i+3}$	1	$N_i = N_{i+1} < N_{i+2} < N_{i+3}$	10	$N_i > N_{i+1} < N_{i+2} < N_{i+3}$	19
$N_i < N_{i+1} < N_{i+2} = N_{i+3}$	2	$N_i = N_{i+1} < N_{i+2} = N_{i+3}$	11	$N_i > N_{i+1} < N_{i+2} = N_{i+3}$	20
$N_i < N_{i+1} < N_{i+2} > N_{i+3}$	3	$N_i = N_{i+1} < N_{i+2} > N_{i+3}$	12	$N_i > N_{i+1} < N_{i+2} > N_{i+3}$	21
$N_i < N_{i+1} = N_{i+2} < N_{i+3}$	4	$N_i = N_{i+1} = N_{i+2} < N_{i+3}$	13	$N_i > N_{i+1} = N_{i+2} < N_{i+3}$	22
$N_i < N_{i+1} = N_{i+2} = N_{i+3}$	5	$N_i = N_{i+1} = N_{i+2} = N_{i+3}$	14	$N_i > N_{i+1} = N_{i+2} = N_{i+3}$	23
$N_i < N_{i+1} = N_{i+2} > N_{i+3}$	6	$N_i = N_{i+1} = N_{i+2} > N_{i+3}$	15	$N_i > N_{i+1} = N_{i+2} > N_{i+3}$	24
$N_i < N_{i+1} > N_{i+2} < N_{i+3}$	7	$N_i < N_{i+1} > N_{i+2} < N_{i+3}$	16	$N_i > N_{i+1} > N_{i+2} < N_{i+3}$	25
$N_i < N_{i+1} > N_{i+2} = N_{i+3}$	8	$N_i < N_{i+1} > N_{i+2} = N_{i+3}$	17	$N_i > N_{i+1} > N_{i+2} = N_{i+3}$	26
$N_i < N_{i+1} > N_{i+2} > N_{i+3}$	9	$N_i < N_{i+1} > N_{i+2} > N_{i+3}$	18	$N_i > N_{i+1} > N_{i+2} > N_{i+3}$	27

$$Dissimilarity(\vec{q}_i, \vec{d}_j) =$$

$$w_B \times dist(MF_B(\vec{q}_i), MF_B(\vec{d}_j)) +$$

$$w_T \times dist(MF_T(\vec{q}_i), MF_T(\vec{d}_j)) +$$

$$w_F \times dist(MF_F(\vec{q}_i), MF_F(\vec{d}_j)) \quad (2)$$

where MF_B, MF_T, *and* MF_F denote the mapping functions based on bi-gram, tri-gram, and four-gram, respectively. Weights w_B, w_T, and w_F represent the importance of bi-gram, tri-gram, and four-gram, respectively. These weights can be adjusted empirically, and $w_B + w_T + w_F = 1$.

2. Method II: Note Difference

From another viewpoint, we can also use the difference between two adjacent musical notes to form a vector. Given a note sequence $\vec{v}$ as 79, 76, 76, 77, 74, 74, 72, 74, 76, 77, 79, 79, 79, the note sequence g($\vec{v}$) is -3, 0, +1, -3, 0, -2, +2, +2, +1, +2, 0, 0, which is derived from the following equation:

$$g(\vec{v}) = (v_2 - v_1, v_3 - v_2, ..., v_{n-1} - v_{n-2}, v_n - v_{n-1}) \quad (3)$$

where v_m is the value of the *m*-th dimension of $\vec{v}$. g($\vec{v}$) is the vector constructed from the note difference sequence $\vec{v}$. Again, the singers' humming and the music phrases in the database are encoded in the same way. The Euclidean distance between vectors $\vec{q}_i$ and $\vec{d}_j$ can be defined as follows:

$$Dissimilarity(\vec{q}_i, \vec{d}_j) = dist(g(\vec{q}_i), g(\vec{d}_j)) =$$

$$\sqrt{\sum_{m=1}^{N}[(q_{i_m} - q_{i_{m-1}}) - (d_{j_m} - d_{j_{m-1}})]^2},$$

$$when \ \dim(g(\vec{q}_i)) = \dim(g(\vec{d}_j)),$$

$$Min_{p=0}^{\dim(g(\vec{q}_i))=\dim(g(\vec{d}_j))} \sqrt{\sum_{m=1}^{N}[(q_{i_{m+p}} - q_{i_{m+p-1}}) - (d_{j_m} - d_{j_{m-1}})]^2},$$

$$when \ \dim(g(\vec{q}_i)) > \dim(g(\vec{d}_j)), and$$

$$Min_{p=0}^{\dim(g(\vec{d}_j))-\dim(g(\vec{q}_i))} \sqrt{\sum_{m=1}^{N}[(q_{i_m} - q_{i_{m-1}}) - (d_{j_{m+p}} - d_{j_{m+p-1}})]^2},$$

$$otherwise, \quad (4)$$

where N is the smaller dimension between $g(\vec{q}_i)$ and $g(\vec{d}_j)$, and p is the beginning offset of the sliding windows for the longer vector. Note that the comparison of the query and the data is performed on the basis of a phrase. Therefore, each vector in both Eq (1) and Eq (4) is transformed from a single phrase. Therefore, the sliding windows is moved in a single phrase when the comparison is performed.

3. Method III: N-gram and Improved Mapping Functions

Method I has an intrinsic problem that the mapping function involves improper distance assignment for different kind of note relationships, which makes the resulting distance unjustifiable. Table 2 lists the baseline mapping function based on tri-gram. If note N_i is less than note N_{i+1} and note N_{i+1} is less than note N_{i+2}, then relationship "<<" is encoded to 1. Table 4 shows an example of such problems for better illustration. If the correct answer in the database is 74, 73, 74, Humming 1 is 72, 73, 74, and Humming 2 is 74, 73, 72, we can see that the fist note of Humming 1 is wrong; i.e., 74 hummed as 72; and the third note of Humming 2 is wrong; i.e., 74 hummed as 72. According to the baseline mapping function of tri-gram, the relationship of [74, 73, 74], " > < ", is encoded to 7. The relationship of Humming 1, " < < ", is encoded to 1, which gives the distance 6 (i.e., $\sqrt{(1-7)^2}$). The relationship of Humming 2, " > > ", is encoded to 9, which gives the distance 2 (i.e., $\sqrt{(9-7)^2}$). Based on the baseline mapping function, the same type of error could result in different distances. It is obvious that this method is improper in measuring the similarity.

To overcome the above mentioned problem, we have summarized a set of relationships as shown in Table 5, each of which is denoted by an ID symbol, i.e., R1, R2, …, or R9. Each ID symbol represents a relationship between the humming query and the data. For example, ID is set to R1 when the relationship is " < < ", which means the humming query is " < " and the data is " < ". It is set to R9 when the humming query is " > " and the data is " > ". The distance value is given for different relationship between the query and

Table 4. An example to explain the distance calculation based on the baseline mapping function

	Note		Note		Note	Relationship	Encode	Distance
Humming 1	72	<	73	<	74	< <	1	6
Correct Answer	74	>	73	<	74	> <	7	
Humming 2	74	>	73	>	72	> >	9	2

Table 5. Relationships between the humming query and the data

Relationships	<	=	>
<	R1	R2	R3
=	R4	R5	R6
>	R7	R8	R9

the data, as shown in Table 6. The relationship between " = " and " < " or " > " is a little closer, and the relationship between " > " and " < " seems further. Therefore, the distance between " = " and " < " or " > " is set to 1, and the distance between " > " and " < " is set to 2.

The improved mapping functions for bi-gram, trigram, and four-gram are illustrated in Table 7, Table 8, and Table 9, respectively. For clarity, Table 10 explains the process for calculating the distance based on the improved mapping functions using the same example. T7-T1 means it is T7 but misconstrued to T1, where T7 is " > < " and T1 is " < < ". Since the distance between " > " and " < " is 2 and the distance between " < " and " < " is 0, the total distance between Humming 1 and the correct data is 2 (2 + 0 = 2). For Humming 2, T7-T9 means it is T7 but misconstrued to T9, where T7 is " > < " and T9 is " > > ". Since the distance between " > " and " > " is 0 and the distance between "< " and " > " is 2, the total distance between Humming 2 and the correct data is 2 (0+2=2), too. Obviously, this result is more reasonable than that of the baseline mapping function. The distance values between any two note relationships (IDs) in the cases of bi-gram, tri-gram, and four-gram can be obtained in a similar way.

IV. EXPERIMENTAL SYSTEM

We have performed several experiments to investigate the performance of the mapping method and probe into some influence factors in order to achieve better performance. Experiments have been conducted on two types of musical formats: MP3 and MIDI. The results show that the performance of melody extraction from MP3 is not good enough and still has a room for improvement in the future. As shown in Yu's approach [14, 15], extracting the main melody from MP3 music objects is a very difficult issue. Moreover, the error propagation gives much influence on the retrieval performance because the MP3 phrase database contains many error musical notes. Comparing with MIDI music, which only has foreground music, MP3 music is much more complicated. For a song in MP3 format, it has not only the voices of singers, but also the background music. Besides, it has no background music when users hum songs through the microphone in the retrieving phrase. This kind of inconsistency makes it more difficult to retrieve the desired song accurately. Based on above observation, the MIDI format is used to verity the

Table 6. Distance values for different relationships between the humming query and the data

Distance	>	=	<
>	0	1	2
=	1	0	1
<	2	1	0

Table 7. Improved mapping function based on bi-gram

Note relationship	ID
$N_i < N_{i+1}$	B1
$N_i = N_{i+1}$	B2
$N_i > N_{i+1}$	B3

Table 8. Improved mapping function based on tri-gram

Note relationship	ID
$N_i < N_{i+1} < N_{i+2}$	T1
$N_i < N_{i+1} = N_{i+2}$	T2
$N_i < N_{i+1} > N_{i+2}$	T3
$N_i = N_{i+1} < N_{i+2}$	T4
$N_i = N_{i+1} = N_{i+2}$	T5
$N_i = N_{i+1} > N_{i+2}$	T6
$N_i > N_{i+1} < N_{i+2}$	T7
$N_i > N_{i+1} = N_{i+2}$	T8
$N_i > N_{i+1} > N_{i+2}$	T9

Table 9. Improved mapping function based on four-gram

Note relationship	ID
$N_i < N_{i+1} < N_{i+2} < N_{i+3}$	F1
$N_i < N_{i+1} < N_{i+2} = N_{i+3}$	F2
$N_i < N_{i+1} < N_{i+2} > N_{i+3}$	F3
$N_i < N_{i+1} = N_{i+2} < N_{i+3}$	F4
$N_i < N_{i+1} = N_{i+2} = N_{i+3}$	F5
$N_i < N_{i+1} = N_{i+2} > N_{i+3}$	F6
$N_i < N_{i+1} > N_{i+2} < N_{i+3}$	F7
$N_i < N_{i+1} > N_{i+2} = N_{i+3}$	F8
$N_i < N_{i+1} > N_{i+2} > N_{i+3}$	F9
$N_i = N_{i+1} < N_{i+2} < N_{i+3}$	F10
$N_i = N_{i+1} < N_{i+2} = N_{i+3}$	F11
$N_i = N_{i+1} < N_{i+2} > N_{i+3}$	F12
$N_i = N_{i+1} = N_{i+2} < N_{i+3}$	F13
$N_i = N_{i+1} = N_{i+2} = N_{i+3}$	F14
$N_i = N_{i+1} = N_{i+2} > N_{i+3}$	F15
$N_i < N_{i+1} > N_{i+2} < N_{i+3}$	F16
$N_i < N_{i+1} > N_{i+2} = N_{i+3}$	F17
$N_i < N_{i+1} > N_{i+2} > N_{i+3}$	F18
$N_i > N_{i+1} < N_{i+2} < N_{i+3}$	F19
$N_i > N_{i+1} < N_{i+2} = N_{i+3}$	F20
$N_i > N_{i+1} < N_{i+2} > N_{i+3}$	F21
$N_i > N_{i+1} = N_{i+2} < N_{i+3}$	F22
$N_i > N_{i+1} = N_{i+2} = N_{i+3}$	F23
$N_i > N_{i+1} = N_{i+2} > N_{i+3}$	F24
$N_i > N_{i+1} > N_{i+2} < N_{i+3}$	F25
$N_i > N_{i+1} > N_{i+2} = N_{i+3}$	F26
$N_i > N_{i+1} > N_{i+2} > N_{i+3}$	F27

Table 10. An example to explain the distance calculation based on the improved mapping function

	Note		Note		Note	Relationship	Encode	Distance
Humming 1	72	<	73	<	74	<<	T1	
Distance		**2**		**0**		T7-T1	2	2
Correct Answer	74	>	73	<	74	><	T7	
Distance		**0**		**2**		T7-T9	2	2
Humming 2	74	>	73	>	72	>>	T9	

effectiveness of the proposed mapping method.

1. Data Collection

The selection of sample songs should be objective and practical. The top 20 hottest songs ranked by CashBox, the nationwide largest KTV chain store in Taiwan, are selected into our music database. In addition, we also store some other hot songs or the ones that were popular before into our database. For generality, the artists include males and females. Most of the songs are pronounced in Mandarin, and some of them in Taiwanese. Finally, there are 20 MP3 songs in our database, which yield a total of 675 MP3 phrases. On the other hand, there are 20 MIDI songs in our database, which yield a total of 705 MIDI phrases. Several testers are requested to hum the top 20 songs in our music database as the query set. To get closer to the real situation, we do not restrict their humming way.

2. Experiment 1: Retrieving MP3 Music Object with Baseline Mapping Functions

Experiment 1 is conducted to investigate how the baseline mapping function works in MP3 database. In the following experiments, the top *n* accuracy rate means the probability that we can find the correct answer in the top *n* songs of the retrieved results. For example, the top 5 accuracy rate is 14.9% means that there is 14.9% of chance to find the correct answer in the top 5 songs given by the system.

A. System setup

The MP3 database contains the Cashbox KTV billboard's hottest 20 songs. These 20 songs are further divided into 675 segments. Five people (3 males and 2 females) are asked to sing all the 20 songs in the database. In total, 47 queries are made and stored for test.

Two experiments are conducted for the MP3 database. In Experiment 1-1, the system uses N-gram (Method I) to measure the similarity. In Experiment 1-2, the system uses near note difference (Method II) to measure the similarity.

B. Experimental results

From Figure 3, we can find that the result of Experiment 1-2 is better than that of Experiment 1-1. However, it is still not good enough and will be further improved in the next

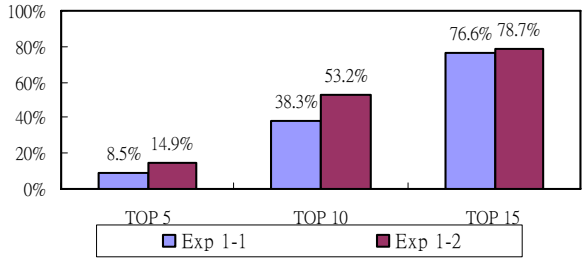

Fig. 3. Accuracy rate of Experiment 1-1 (Method I) and 1-2 (Method II)

experiment. The potential reasons are shown as follows:

- The performance of melody extraction on MP3 music objects is not good. MP3 music is not as simple as MIDI music. Since MP3 music object includes vocal, melody and some other background music, segmentation cannot be done accurately and thus pitch extraction is difficult as well.
- Singers do not sing the songs correctly. In other words, the query in fact does not express the users' actual need. Besides, other facts like the quality of the microphone, sound recording driver and the recording environments may also result in the inaccuracy.
- The mapping method is not good enough.

Among these influence factors, we believe that melody extraction of MP3 music objects is the most important one. For the fifth section of one song (AI QING ZI DIAN, in Chinese), its correct musical notes are: 83, 83, 80, 81, 80, 78, 76, 78, 78, 80, and 73. But the musical notes extracted by our tool are: 81, 83, 83, 80, 80, 81, 81, 79, 80, 83, 76, 83, 83, 78, 78, 78, 68, and 69. There are 7 semitones inserted. The insertion error is about 39% and the substitution error is about 36%. On the other hand, we use the same program to catch the musical notes from the query given by one singer who hums the selected music, which are: 76, 77, 77, 75, 75, 75, 74, 74, 72, 71, 77, 72, 72, 72, 73, 63, and 66. The melody values have 6 semitones inserted. The insertion error is about 38%. After shifting them to up by 8 semitones, the substitution error is about 64%. By the results, we can know the main factor which influences the retrieval result is the poor recognition accuracy of musical notes (no matter what melody it is, either

music data or humming data). In the long run, we intend to develop an auto-correlation-based method to extract melody on MP3 music object, which is still under an on-going research stage.

3. Experiment 2: Retrieving MP3 Music Object with Improved Mapping Functions

The goal of Experiment 2 is to examine if the improved mapping function works better than the baseline mapping function.

A. System setup

For comparison, Experiment 2 conducts the same queries on the same MP3 database as what is done in Experiment 1. However, Experiment 2 uses N-gram (Method III) with the improved mapping functions to measure the similarity.

B. Experimental Results

From Figure 4, we can observe that the improved mapping function outperforms the baseline one: the top 5 accuracy rate is improved from 8.5% to 27.6%; the top 10 accuracy rate is promoted from 38.3% to 66.0%; the top 15 accuracy rate is promoted from 76.6% to 91.5%. However, the performance is still not good enough. As we mentioned in Experiment 1, the major influence factor is melody extraction of the MP3 music object.

4. Experiment 3: Retrieving MP3 Object by Humming with Error

We randomly select a phrase from each song and substitute 30% of its notes. In this way, we collect 30% error phrases as the test query set to simulate users' humming. The aim of this experiment is to evaluate the proposed mapping method under a controlled mismatch condition between the database and the queries. Figure 5 illustrates how a simulated query is generated. In this example, there are 7 notes in a phrase of a song, "I Believe", in the database. An approximate estimate of 30% of the note number is 2. This means that we need to substitute 2 notes. Therefore, the process randomly selects 2 notes, and then adds or subtracts their values by 3.

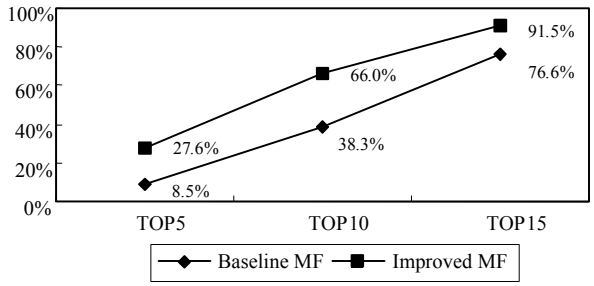

Fig. 4. Accuracy rate of Baseline MF and Improved MF in Experiment 2

A. System setup

For comparison, Experiment 3 uses the same MP3 database and queries as Experiments 1 and 2. However, Experiment 3-1 uses N-gram (Method III) with the improved mapping functions to measure the similarity and Experiment 3-2 uses near note difference (Method II) instead.

In this experiment, both inside test and outside test are performed. Inside test means that the original melody phrase that is used to generate the query set is still in the database. Outside test means that the original melody phrase is removed from the database. The purpose is to observe what is different between these two cases.

To make the outside test meaningful, the phrase being chosen as a query must be repeated in the song. Otherwise, the outside test could be meaningless because there is no phrase similar to the query at all. Therefore, the simulated queries are all selected from the chorus part. Another noticeable point is that the ways that the singer sings the repeated parts are similar but not exactly the same. The original phrase of a simulated query could be different to its corresponding repeated phrases. As such the error between the simulated query and the corresponding repeated phrases could be higher than 30%.

B. Experimental results

The results of Experiment 3-1 and Experiment 3-2 are shown in Figures 6 and 7, respectively. As expected, the result of the inside test is better than that of the outside test. For both the inside test and the outside test, Method III in general performs better than Method II. It shows the N-gram similarity measurement with improved mapping function is more flexible because it encodes the up and down relationships

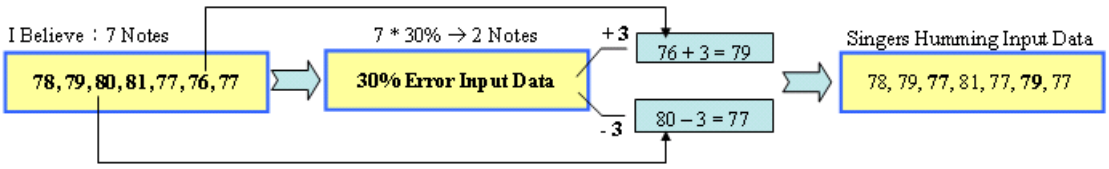

Fig. 5. Simulated query generation process

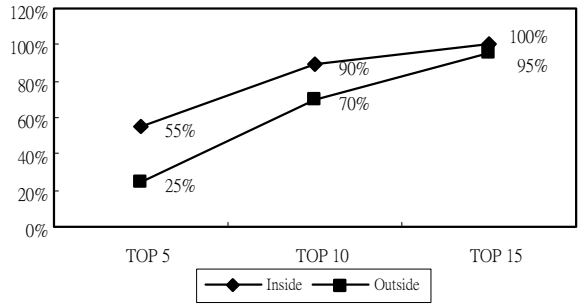

Fig. 6. Retrieval performance of Experiment 3-1 (Method III)

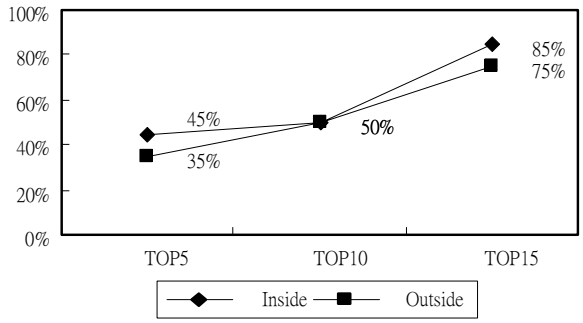

Fig. 7. Retrieval performance of Experiment 3-2 (Method II)

of contiguous notes. Since Method II keeps the pitch difference between two contiguous notes, it is thus more rigid. This experiment shows a promising result of retrieval using Method III: the top 5 accuracy is 55%, top 10 accuracy is 90% and the top 15 accuracy is 100%.

By comparing Figure 6 with Figure 4, it can be found that the result of Experiment 3-1 is better than that of Experiment 2. It is obvious that the mismatch between the humming query and the target phrase is higher than 30%, which is set for generating the simulated queries in Experiment 3-1. The mismatch could come from the melody extraction process, the improper way the user hums the song, the quality of the microphone, the recording driver for sound, the recording environments for query, and so on. As mentioned above, while generating simulated queries, 30% of the original notes are adjusted up or down 3 semitones. Note that the adjustment probably does not have significant influence or 30% is probably too small to have significant influence.

5. Experiment 4: Retrieving MIDI Music Object

In order to evaluate the matching method under the condition without any error in the database, we take MIDI music to conduct experiments. We use Cakewalk software to extract the "correct" melody (MIDI note) from the MIDI music. CakeWalk Pro Audio is the top of the Twelve-Tone audio line. It features up to 64 audio tracks, 256 virtual tracks (midi/audio), and 64 channels of real-time effects. It also supports for DirectX plug-in, RealAudio, Pitch to MIDI conversion, Notation, Track inserts, Real-time marker placement, Groove quantize, Pattern sequencing and tons, etc.

A. System setup

The MIDI database contains the Cashbox KTV billboard's hottest 20 songs. These 20 songs are further divided into 705 phrases. Experiment 4-1 uses N-gram (Method III) as similarity measuring method and Experiment 4-2 uses near note difference (Method II) for comparison.

B. Experimental results

In this experiment, the melodies of all songs in the database are extracted by Cakewalk 9.0. It is assumed that there is no error at all. Comparing the melodies of the MP3 music extracted by Yu's program [11, 15] with those of the corresponding MIDI music, the error rate for note extraction is about 36%. Figure 8 illustrates the retrieval performance of Experiment 4-1 and Experiment 4-2. We can see the top 1 accuracy can be as high as 80%. Method II is still worse than Method III. Comparing Figure 8 with Figure 4, it is obvious that the result of Experiment 4-1, which is tested on MIDI database, is better than that of Experiment 2, which is tested on MP3 database.

6. Experiment 5: Retrieving MIDI Object by Humming Simulation

In this experiment, we use the same 20 simulated queries that are used in Experiment 3 to test on the MIDI database.

A. Experimental setup

The MIDI database contains the Cashbox KTV billboard's hottest 20 songs. These 20 songs are further divided into 705 phrases. Five people (3 males and 2 females) are asked to sing all the 20 songs in the database.

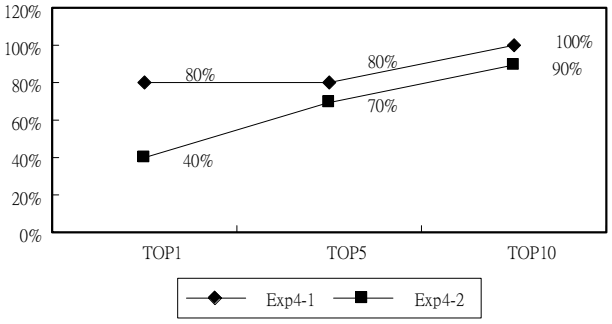

Fig. 8. Retrieval performance of Experiment 4-1 (Method III) and 4-2 (Method II) with simulated queries

B. Experimental results

Figure 9 illustrates the retrieval performance of Experiment 5-1 and Experiment 5-2. Comparing Figure 9 with Figure 8, it is obvious that the result for using simulated queries is better than that for using real humming queries, even though the simulated queries are constructed from the MP3 phrases by substituting 30% of their notes. This is probably because the real humming queries are performed by our laboratory members while the MP3 phrases are performed by professional artists. From Figure 9, we still find that Method III is better than Method II as well.

Figures 10 and 11 show the comparison of MIDI, MP3 inside test and MP3 outside test using Method III and Method II, respectively. It is obvious that the results for MIDI experiments are in general better than those for MP3 experiments.

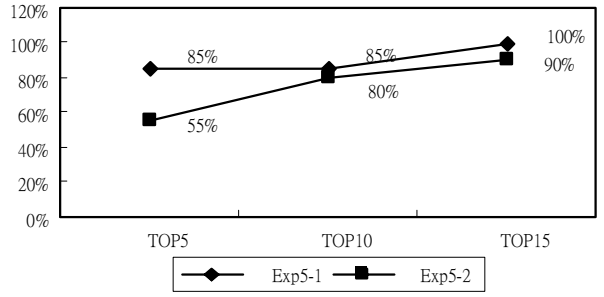

Fig. 9. Retrieval performance of Experiment 5-1 (Method III) and 5-2 (Method II) with humming queries

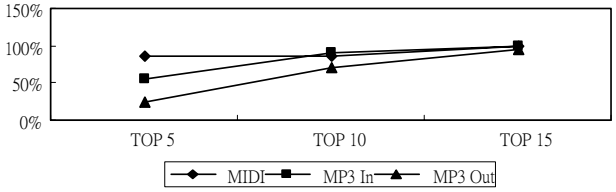

Fig. 10. Comparison of MIDI, MP3 inside test and MP3 outside test using Method III

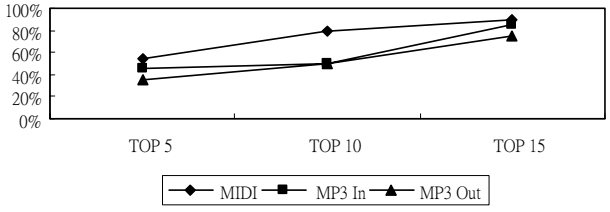

Fig. 11. Comparison of MIDI, MP3 inside test and MP3 outside test using Method II

V. CONCLUSIONS

The aim of this paper is to develop a system for users to retrieve the music object by simply humming. We first investigate the matching methods of music according to its characteristics. Then we conduct some experiments to examine our approach. The experimental results show that Method III (in both N-gram and Improved Mapping Functions) outperforms Method II (Note Difference). For Method III, how to set the proportion of bi-gram, tri-gram and four-gram is not straightforward. Bi-gram information is naturally loose, but it is more flexible and fault-tolerant. Four-gram is more rigid and strong, but it is less flexible. The behaviors of tri-gram are between those of bi-gram and four-gram. The experimental results indicate that it will have a better result if the proportion of bi-gram, tri-gram and four-gram is in the descending order (i.e., tri-gram > bi-gram > four-gram). On the other hand, Method II not only keeps the characteristics of the up and down relationships in music melody but also keeps the characteristic of pitch difference. It is too rigid in a sense. In view of the above-mentioned facts, if the user's humming is almost correct, the matching result using Method II is better than that achieved by Method III. However, if the user's humming makes some mistakes, the matching result using Method III is much better than that using Method II.

Since there is no simple and straightforward solution in conducting a content-based music retrieval task, our future work includes:

1. Develop more reasonable mapping functions: We will try to develop other mapping functions to better characterize the music features and further improve the matching result.
2. Develop more fault-tolerant techniques: In our experiments, we found that Method III is more fault-tolerant than others. However, it still has a room for improvement. We will try other techniques, e.g. dynamic time warping, to raise the fault-tolerance of the system.
3. Examine more kinds of characteristics of music: In this paper, we only use the pitch as characteristics of music. In the future, we intend to involve some other characteristics to describe music objects, e.g. duration, rhythm [1], harmony, etc.
4. Develop a better method of extracting the melody of MP3 object: In this study, we found that the melody extraction of MP3 objects has great influence on retrieval performance. Therefore, a more reliable pitch tracking method is required.

REFERENCES

1. Chen, J. C. C. and A. L. P. Chen (1998) Query by rhythm: an approach for song retrieval in music databases. 8th

International Workshop on Research Issues in Data Engineering, Adam's Mark Hotel, Orlando, Florida.

2. Doraisamy, S. and S. Rüger (2001) An approach towards a polyphonic music retrieval system. 2nd International Symposium on Music Information Retrieval, ISMIR2001, Indiana University, Bloomington, Indiana.
3. Doraisamy, S. and S. Rüger (2002) A comparative and fault-tolerance study of the use of n-grams with polyphonic music. 3rd International Symposium on Music Information Retrieval, ISMIR 2002, Ircam - Centre Pompidou, Paris, France.
4. Ghias, A, J. Logan, D. Chamberlin and B. C. Smith (1995) Query by humming: musical information retrieval in an audio database. 3rd ACM International Conference on Multimedia, San Francisco, California.
5. Jang, J. S. R. and H. R. Lee (2001) Hierarchical filtering method for content-based music retrieval via acoustic input. 9th ACM International Conference on Multimedia, Ottawa, Ontario, Canada.
6. Kosugi, N., Y. Nishihara, S. Kon'ya, M. Yamamuro and K. Kushima (1999) Music retrieval by humming. Pacific Rim Conference on Communications Computers and Signal Processing (PACRIM'99), Victoria, B.C. Canada.
7. Kosugi, N., Y. Nishihara, T. Sakata, M. Yamamuro and K. Kushima (2000) A practical query-by-humming system for a large music database. 8th ACM International Conference on Multimedia, Los Angeles, CA.
8. Kuo, W. Y. and C. C. Liu (2000) *Automatic Feature Extraction and Temporal Segmentation of MP3 Music Objects*. Master Dissertation. Chung Hua University, Hsinchu, Taiwan.
9. Liu, C. C. and P. J. Tsai (2001) Content-based retrieval of MP3 music objects. The ACM International Conference on Information and Knowledge Management, Atlanta, Georgia.
10. Mo, J. S., C. H. Han and Y. S. Kim (1999) A melody-based similarity computation algorithm for musical information. Knowledge and Data Engineering Exchange Workshop (KDEX '99), Chicago, Illinois.
11. Pan, D. (1995) A tutorial on MPEG/Audio compression. *Journal of IEEE Multimedia*, 2(2), 60-74.
12. Shlien, S. (1994) Guide to MPEG-1 audio standard. *Journal of IEEE Transactions on Broadcasting*, 40(4), 206-218.
13. Tseng, Y. H. (1996) A survey of technologies for multimedia information retrieval. *Journal of Information, Communication, and Library Science*, 3(2), 44-53.
14. Yu, H. M. and C. C. Liu (2001) *Query MP3 Music Ddatabase by Humming*, National Computer Symposium, Taipei, Taiwan.
15. Yu, H. M. and C. C. Liu (2002) A background reduction technique for effective retrieval of MP3 music objects. Master Dissertation. Chung Hua University, Hsinchu, Taiwan.

收件：94.07.06　修正：95.05.01　接受：95.08.04