Computer Music Query by Humming Considering Subsequence Matching Algorithm

Xiangli Jin*

School of Music, Suihua University, Suihua, Heilongjiang 152000, China

*Corresponding author e-mail: xianglijin69@su.edu.cn

Abstract. Facing massive music data, how to create its index in content-based retrieval to improve retrieval speed is a crucial research content. In the feature extraction from music content based on sentence and index creation, the subsequence matching algorithm of music melody is stored in the database. Database However, when users conduct query by humming (QBH), they may hum a piece of music containing multiple sentences for retrieval, which requires the generation of multiple sentence features. Hence, a solution to single sentence feature, multi-sentence conversion and matching is proposed and applied to the retrieval system, with an excellent retrieval effect. The relevant achievements are also applicable to the retrieval system of time series data with a similar structure.

Keywords: Index, Computer Music Retrieval, Feature Conversion

1. Introduction

Music is an essential type of audio data. With the rapid development of the Internet and the popularization of digital instruments, the study of content-based music retrieval has received widespread attention. Query by humming (QBH) is the most convenient retrieval method. Compared with the traditional retrieval of external information such as the name of the song and the singer, it is retrieved based on the inherent features of the music's melody and rhythm [1-2]. Retrieving music by humming allows the user to find a song in which he only knows part of the melody. Users simply hum the tone through the computer's microphone, and then the system queries the song melody database that contains this tone and returns a list of relevant songs as the result of the query [3-4]. This search method is more natural and convenient than text search (track title, singer, etc.). It has excellent potential for commercial development and has become a hot spot for many people in recent years. Ghias studied humming retrieval first. His proposal is to use three symbols U (increase), R (unchanged), and D (decrease), to represent the pitch changes of two adjacent notes of the melody [5-6]. Then apply the approximate string matching method to compare the similarity of the two melodies. However, this melody information description method is relatively simple and not ideal in recognition. Kosugi et al. adopted a melody expression method that combines pitch and rhythm information, applicable to the retrieval of large music databases. However, this system requires users to hum along with a metronome, which is inconvenient to use. Shih et al. used a Hidden Markov Model (HMM) in the QBH system to compare the similarity between the humming melodies and the target songs.
Experiments indicate that this method is more sensitive to pitch inaccuracies. However, its humming error tolerance in rhythm is relatively high. National Tsing Hua University (NTHU) used DTW and Line Scaling multi-level matching algorithms in the humming retrieval system. However, due to DTW, Line Scaling adopted the fundamental frequency curve for matching directly, resulting in a low retrieval speed of the system [7-8].

None of the existing music indexing methods consider the features of music - music is based on sentences. Users generally hum in complete sentences, but this paper uses this to search to reduce the amount of calculation, when the contour feature of the melody used is obtained by calculating the relationship between the following note and the preceding adjacent note in the unit of a sentence. The contour features stored in the feature database are all calculated in the unit of a sentence. However, users sometimes hum a piece of music that contains multiple sentences. At this time, it is impossible to use the features in the database directly. The sentences the user hums may be random and cannot be stored in the database. On this basis, the corresponding solutions based on the database are provided in this paper.

2. Retrieval System Framework
The process of the content-based music retrieval system is shown in Figure 1. Users hum the melody. The system processes the humming sound, extracts the features, uses the matching algorithm to match, and finally returns the retrieval result to the user.

In this paper, mainly the retrieval algorithm when users hum multiple sentences in the music retrieval method based on the sentence index is studied.

3. Music Retrieval Algorithm By Humming
The melody contour features stored in the content-based humming music database are sentence-based. The melody contour is calculated from the two notes before and after. Its composition is:

“Sentence length; first note interval, last note interval; interval difference between adjacent notes.”

On this basis, the multi-sentence melody contour retrieval needs to have corresponding features to match the humming melody. Hence, a set of candidate music fragments shall be generated. Since each sentence feature has sentence length information, the sentence length features of adjacent sentences can be added to obtain the corresponding candidate music fragment. Subsequently, the feature value of the candidate music fragment is generated. Sentence length feature refers to the number of musical features stored in the database in units of sentences. The number of intervals between adjacent notes in a sentence, corresponding to the number of features extracted from the user's humming sentence, in units. There are two ways to generate eigenvalues:

a) Generate the corresponding melody contour features directly. When the contour features of two sentences are connected, a feature value needs to be added between two sentences. The value is the interval difference between the first note of the next sentence and the last note of the previous sentence. The specific algorithm is described in 2.1. However, when the user hums, the two sentences are prone to inaccurate humming. The added feature value may cause the detection rate to decrease.

b) Ignore the feature value between sentences. In this case, only the existing features in the sentence are matched during retrieval.

3.1. Computer Music Retrieval Candidate Feature Generation Algorithm
To retrieve the humming music segment input by the user, after preprocessing, feature extraction, and melody contour extraction, the feature sequence $G=(g_1, g_2, \ldots, g_n)$ is obtained, and the sequence length is
m. For the music database \( D = \{ M_1, M_2, \ldots, M_d \} \), the sentence set is \( M_i = \{ S_1, S_2, \ldots, S_{ms_i} \} \), \( S_i(1 \leq i \leq ms) \) is any sentence of the song, ms is the total number of sentences in the song, and then the sequence of notes composing \( S_i \) is \( S = \{ w_1, w_2, \ldots, w_n \} \), where \( n \) is its length, that is, the number of notes in each sentence. For computer music retrieval algorithms, this paper defines a set of candidate music fragments:

Definition 1: A collection of candidate music pieces. With any music in the database

\[
M_j = \{ S_{1j}, S_{2j}, \ldots, S_{mj} \}
\]

The sentence length sequence \( L_i = \{ l_1, l_2, \ldots, l_{ms} \} \), for the music fragment \( G \) to be retrieved with the length of \( ml \), the candidate music fragment set \( H \) is defined as

\[
H = \left\{ S_i + S_{i+1} + \cdots + S_j \right\} \\
\left\| l_i + l_{i+1} + \cdots + l_j - ml \right\| \leq \varepsilon_H \quad (1 \leq i \leq j \leq k)
\]

Where \( \varepsilon_H \) represents the self-set length tolerance threshold, that is, the minimum length difference range of two music fragments to be compared.

The following is a detailed introduction to the computer music retrieval candidate feature generation algorithm.

1) Generate a set of candidate fragments

On the basis of the above definition, the algorithm of Figure 2 is used to generate a set of music candidate segments \( H \).

**Figure 2.** Process diagram of candidate fragment generation

2) Generate candidate features

Based on the set of candidate fragments, the following two methods are used to generate corresponding feature values:

a) Generate the corresponding melody contour features directly. Based on the generated candidate music fragment collection, the corresponding melody contour feature algorithm is described as follows:

(a) Generate the melody contour feature. If the user hums a multi-sentence melody, the extracted feature is the overall feature composed of the multi-sentence melody.

For the multi-sentence melody feature, a feature value needs to be added between two adjacent sentences. This value is the interval difference between the first note of the next sentence and the last note of the previous sentence. Then the features of the two adjacent sentences \( S_1 \) and \( S_2 \) are

\( S_1 \) interval feature + \( S_2 \) first note interval and \( S_1 \) last note interval difference + \( S_2 \) interval feature

Then get the melody contour feature of the candidate music segment.

(b) Melody rhythm features. Mark the characteristic of humming melody as QRL (query rhythm length).

\[
QRL = \{ qrd_1, qrd_2, \ldots, qrd_{ms} \}
\]

The song pitch field in the database saves the note length DRL (database rhythm length) of each song's sentence.

\[
DRL = \{ drl_1, drl_2, \ldots, drl_s \}
\]
Set $N = \min(m, n)$, define $RL[N] = \left\{ \frac{qrl_1}{drl_1}, \frac{qrl_2}{drl_2}, \ldots, \frac{qrl_n}{drl_n} \right\}$, calculate $dist_i = \sum_{i=2}^{N} |RL[i] - RL[1]|$, and use dynamic programming to get the minimum $dist_i$.

b) Ignore specific feature values for the humming sound are continuous, and there are extracted features between the two sentences, but the two sentences are connected by the user inaccurately. In the feature extraction process, one of the joint Characters sometimes extract multiple features. In this paper, the feature of the connection is set as a wildcard character*%, and any characters can be matched to obtain the feature of humming by users.

### 3.2. Computer Music QBH Algorithm

Since the humming errors of users can cause inaccuracy of the humming melody feature, the dynamic time warping (DTW) algorithm is used to the correspondence and matching of the feature characters. DTW is a non-linear warping technique that combines time warping and distance measurement calculations. It is generally used for time warping in speech recognition systems. In this paper, it is used to conduct position matching and similarity calculation when there are insertion or deletion errors in the input string, and obtain the most suitable matching subsequence and maximum similarity. If the corresponding characters of two substrings are not perfectly matched, the similarity of the two substrings is calculated using the definition of similarity in this paper. The algorithm is described as follows:

a) Extract the humming feature string of the user, denoted as

$$S = b_1b_2\cdots b_n \ (N \geq 0)$$

b) Based on the length $N$ of the humming string, two methods are used to generate candidate feature strings, which are recorded as

$$T = a_1a_2\cdots a_m \ (M \geq 0),$$

$$|M - N| < \varepsilon$$

(6)

c) Implement the humming song search based on DTW algorithm.

DTW mainly looks for the time warping function $m = w(n)$, which non-linearly maps the time axis $n$ of the input string to the time axis $m$ of the reference string, and the $w$ satisfies

$$dis = \min_{w(n)} \sum_{n=1}^{N} d \left( a_{w(n)}b_{w(n)} \right),$$

where $d \left( a_{w(n)}b_{w(n)} \right)$ is the distance between the $n$th feature in the input string and the $m$th feature in the reference string, and $dis$ is the distance measure of the two strings corresponding to the optimal time warping. The distance-based approximate string matching algorithm is described as follows:

Input: string $S = s_1s_2\cdots s_n (n \geq 0)$, where each $s_i \in \Sigma$ ; each record in the database.

Output: Several records in the database that are similar to $S$.

a) The distance matrix $D$ of each character in the $i$-th (initial value 1) is calculated. The $T$ and $S$ are recorded in the database.

b) The dynamic programming method is used to obtain the minimum path in the matrix.

c) Based on the definition of similarity, the similarity of $T$ and $S$ can be obtained by using the difference between the minimum sum and length of the two strings. The similarity value is saved, and 1 is added to the value of $i$. Return to a) Continue to loop until the last record.

d) Among all similarity values, return several records with higher similarity, which is the result.

Hence, users can detect the target song by humming any melody, realizing computer music retrieval.
3.3. Algorithm Application

The music database used by the system comes from the MIDI format music files collected from the Internet, single track mode, a total of 340 songs. The experimental humming record section requires users to hum with “Da” voice. The collection parameters are as follows: sampling frequency 11025Hz, monophonic Channel sampling, the number of quantization bits is 8 bits. The feature of this method is that there will be a low-energy interval between notes. The system can more accurately divide the notes by judging the change of signal energy over time.

In the experiment, a total of 12 experimenters from the laboratory were invited. The humming level was all at an ordinary level. Among them, there are 6 females and 6 males. The age distribution of the experimenters is 3 males and females 20-30 years old, and 3 males and females 40-50 years old. Different songs are hummed 10 times, respectively, with 120 hum sections in total for 10~15s. Figure 3 shows the comparison of the top three, top ten, and top twenty song hit rates of experimenters of different genders and different ages under the subsequence matching algorithm of this system. Based on the experimental results, the algorithm has little effect on gender and age, which proves that the algorithm has a relatively good tolerance for audio changes and pitch changes. In order to compare the effects of the new algorithm, the BF algorithm, the KMP algorithm, the traditional DTW algorithm, the pitch and pitch direct addition subsequence matching algorithm, the frame-based subsequence matching algorithm, and the subsequence matching algorithm of this paper were tested, respectively. The test results As shown in Table 1. Table 1 lists the hit rate and running time ratio results of the top three, top 10, and top 20 target songs of the 120 humming records under the six matching algorithms.

![Figure 3](image)

**Figure 3.** Classification statistics of the number of songs hits by the experimenter under the subsequence matching algorithm

Based on Table 1, the accuracy and efficiency of the DTW algorithm compared to the BF algorithm and the KMP algorithm have significant advantages. However, the improvement effect of directly adding pitch and length is not obvious. The frame-based subsequence matching algorithm avoids note cutting, and its accuracy is improved compared with the subsequence matching algorithm in this paper. However, the time is three times that of this paper. This will not work on large data sets. The subsequence matching algorithm in this paper has improved in speed and accuracy. Although the improvement in hit rate is small, the time used by the subsequence matching algorithm has been greatly reduced, and the system performance remarkably improved, which has provided a feasible solution to the future application of retrieval in large-scale databases.

**Table 1.** Comparison of experimental results of six matching algorithms

| Algorithm | Top three | Top ten | Top 20 | Elapsed time/DTW elapsed time (%) |
|-----------|-----------|---------|--------|---------------------------------|
| BF        | 11.6%     | 21.6%   | 31.7%  | 3.79                            |
| KMP       | 18.3%     | 28.3%   | 41.6%  | 2.31                            |
|                  |                |                |                |        |
|------------------|----------------|----------------|----------------|--------|
| Dynamic time bending | 72.5%          | 87.8%          | 95.4%          | 1.00   |
| DTW improvement two    | 76.9%          | 89.3%          | 96.8%          | 2.41   |
| Subsequence matching algorithm | 75.2%          | 88.4%          | 95.5%          | 0.84   |

4. Conclusions
In this paper, sentence-based retrieval idea is proposed based on the features of music. The solution to users humming multiple sentences and the calculation method for candidate music pieces are provided. Finally, two algorithms for computer music retrieval are implemented. The experiments have verified that the algorithm proposed in this paper can detect the target music effectively when users hum multiple sentences. The relevant study results of this paper are also applicable to other retrieval systems in time series data with a specific structure.

Acknowledgements
Funded by: "Research on the Inheritance and Rheology of Korean Nongmusic Culture in the Cold Black Soil Region" (Project No.: YWK10236200105), Project Host: Xiangli Jin, Project Type: Youth Project, Subject Type: Music Research.

References
[1] Kotsifakos, A., Karlsson, I., Papapetrou, P., Athitsos, V., & Gunopulos, D. (2015). Embedding-based subsequence matching with gaps–range–tolerances: a query-by-humming application. Vldb Journal, 24(4), 519-536.
[2] Barth, J., Oberndorfer, C., Pasilosta, C., Schülein, Samuel, Gassner, H., & Reinfelder, S., et al. (2015). Stride segmentation during free walk movements using multi-dimensional subsequence dynamic time warping on inertial sensor data. Sensors, 15(3), 6419-6440.
[3] Qu, J. F., Yuan, L., Huang, Y., & Wu, Z. (2016). An efficient algorithm for attribute-based subsequence matching. Information ences, 334-335, 323-337.
[4] CHI-YONG, LEE, WEN-HSING, & LAI. (2016). Query by singing/humming system using segment-based melody matching for music retrieval. WSEAS Transactions on Systems, 15, 157-167.
[5] Yunjing, W. (2013). Similarity matching method for music melody retrieval. Journal of Multimedia, 8,4(2013-08-01), 8(4), 386-393.
[6] Dong, Y. F., & Qi, B. (2012). A new approximate matching algorithm and its application in internet music search by humming. Advanced Materials Research, 433-440, 3662-3668.
[7] Deorowicz, S., & Grabowski, S. (2012). A hybrid algorithm for the longest common transposition-invariant subsequence problem. Computing & Informatics, 28(5), 729-744.
[8] Hong, L., & Center, N. I. (2013). Graph-based near-duplicate video subsequence matching algorithm. application research of computers, 30(12), 3857-3862.