IMPROVED QUERY-BY-TAPPING VIA TEMPO ALIGNMENT

Chun-Ta Chen
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
chun-ta.chen@mirlab.org

Jyh-Shing Roger Jang
Department of Computer Science
National Taiwan University
Taipei, Taiwan
jang@mirlab.org

Chun-Hung Lu
Innovative Digitech-Enabled Applications & Services Institute (IDEAS), Institute for Information Industry, Taiwan
enricoghlu@iii.org.tw

ABSTRACT

Query by tapping (QBT) is a content-based music retrieval method that can retrieve a song by taking the user's tapping or clapping at the note onsets of the intended song in the database for comparison. This paper proposes a new query-by-tapping algorithm that aligns the IOI (inter-onset interval) vector of the query sequence with songs in the dataset by building an IOI ratio matrix, and then applies a dynamic programming (DP) method to compute the optimum path with minimum cost. Experiments on different datasets indicate that our algorithm outperforms other previous approaches in accuracy (top-10 and MRR), with a speedup factor of 3 in computation. With the advent of personal handheld devices, QBT provides an interesting and innovative way for music retrieval by shaking or tapping the devices, which is also discussed in the paper.

1. INTRODUCTION

QBT is a mechanism for content-based music retrieval which extracts the note onset time from recordings of users' input tapping or symbolic signals, which it then compares against a song database to retrieve the correct song. Unlike query-by-singing/humming (QBSH) [1, 2], which takes the user's melody pitch for comparison, QBT only uses the note duration for comparison, with no pitch information. This makes QBT more difficult to implement than QBSH, because the note onset in QBT contains less information than the musical pitch in QBSH, raising the likelihood of collision. For example, musical pieces with different melodies but similar rhythmic patterns may be characterized by the same onset sequence.

One may argue that QBT is not a popular way of music retrieval. Some people may even think it is not useful. However, with the advent of personal handheld devices, we can think QBT as a novel way of human-computer interface. For instance, with the use of QBT, one may shake or click his/her mobile phones in order to retrieve a song. Moreover, one can even use a personal style of shaking or clicking as the password to unlock a phone. These innovative ways of human-machine interface indicate that QBT, though not the most popular way of music retrieval, is itself interesting and could pave the way for other innovative applications [10].

QBT system algorithms are based on the estimation of the similarity between two onset sequences. For example, G. Eisenberg proposed a simple algorithm called "Direct Measure" to accomplish such comparisons [3, 4]. R. Typke presented a variant of the earth mover's distance appropriate for searching rhythmic patterns [5]. Among these algorithms, the techniques of dynamic programming (DP) have been widely used, such as R. Jang's Dynamic Time Warping (DTW) [6], G. Peters's edit distance algorithm [7, 8], and P. Hanna's adaptation of local alignment algorithm [9].

In this paper, we propose and test a new QBT algorithm. In Section 2, we discuss the general frameworks of QBT and existing QBT methods. Section 3 describes the proposed method. Experiments with different QBT techniques are described in Section 4. Finally, Section 5 concludes this paper.

2. THE QBT SYSTEM

Fig. 1 illustrates the flowchart of our query-by-tapping sys-tem. In general, there are 2 kinds of inputs to a QBT sys-tem:

- Symbolic input: The onset time of the tapping event is provided symbolically with little or no ambiguity. For instance, the user may tap on a PC's keyboard or an iPad's touch panel to give the onset time.
- Acoustic input: The onset time is extracted from acoustic input of the user's tapping on a microphone. This input method requires additional onset detection to extract the onset time of the acoustic input. For example, we can estimate the onset time by local-maximum-picking of the input audio's intensity as in [5], or by detecting the transients of kurtosis variation as in [7].

The input onset sequence can be obtained as the inter onset interval (IOI) vector whose elements are the difference between two successive onset times. The note onset sequences extracted from the monophonic MIDIs (or the melody track of polyphonic MIDIs) in the song database are also converted into IOIs in advance. We can then apply a QBT algorithm to compare the query IOI vector to those in the database in order to retrieve the most similar song from the database. A QBT algorithm usually needs to perform IOI vector normalization before similarity comparison. Normalization can take care of tempo devia-

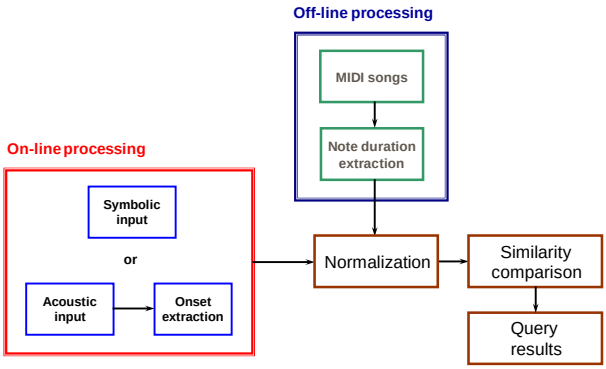


Fig. 1. QBT system flowchart

tion, which similarity comparison can handle possible insertion/deletion errors. Once normalization is performed, we can apply similarity comparison to find the similarity between the IOI query vector and that of each database song. The system can then return a ranked list of all database songs according to their similarity to the query input.

Normalization and the similarity comparison are detailed in the following sections.

2.1 Normalization of IOI Vectors

In most cases, the tempo of the user's query input is different from those of the candidate songs in the database. To deal with this problem, we need to normalize the IOI vectors of the input query and the candidate songs. There are 2 common methods for normalization. The first one is to convert the summation of all IOI to a constant value [5].

$$\tilde{q}_i = q_i \Big/ \sum_{k=1}^{m} q_k \qquad \tilde{r}_j = r_j \Big/ \sum_{k=1}^{\tilde{n}} r_k \tag{1}$$

where $\{q_i,\ i=1\sim m\}$ is the input query IOI vector, and $\{r_j,\ j=1\sim\tilde{n}\}$ is a reference IOI vector from the song database. Note that the reference IOI vector of a song is truncated to a variety of lengths in order to match the query IOI. For instance, $\tilde{n}$ may be set to a value from $m-2$ to $m+2$ in order to deal with possible insertions and deletions in the query input. Thus all these variant normalized versions of the IOI vectors for a song must be compared for similarity with the query IOI vector. The second method is to represent the normalized IOI vector as the ratio of the current IOI element to its preceding element [7]. That is:

$$\begin{cases} \tilde{s}_1 = 1 \\ \tilde{s}_i = s_i / s_{i-1}, \text{ if } i \geq 2 \end{cases} \tag{2}$$

where $\{s_i\}$ is the original input query or reference IOI vector, and $\{\tilde{s}_i\}$ is its normalized version. The advantage of this method is that computation-wise it is much simpler than the first one. However, this method is susceptible to the problem of magnified insertion and deletion errors of the original IOI vectors, if any. For example, an IOI vector is [1, 2, 1], then its normalized vector is [1, 2, 0.5]. If this IOI vector is wrongly tapped as [1, 1, 1, 1] (i.e., with one insertion in the second IOI), the normalized will become [1, 1, 1, 1], which has a larger degree of difference from the groundtruth after normalization. This kind of amplified difference is harder to recover in the step of similarity comparison.

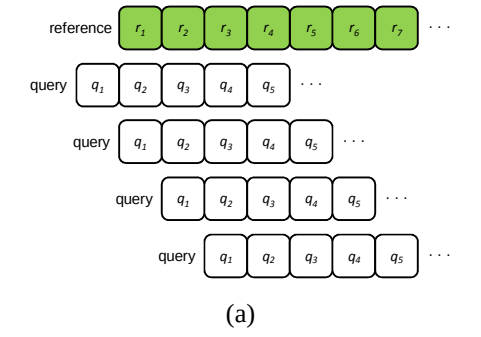
2.2 Similarity Comparison

A robust QBT system should be able to handle insertion and deletion errors since most of the common users are not likely to tap the correct note sequence of the intended song precisely. In particular, a common user is likely to lose one or several notes when the song has a fast tempo, which leads to deletion errors. On the other hand, though less likely, a user may have a wrong impression of the intended song and taps more notes instead, which lead to insertion errors. Several methods have been proposed to compare IOI vectors for QBT, including the earth mover's distance [4] and several DP-based methods [5], [6], [7] which can deal with two input vectors of different lengths. In general, the earth mover's distance is faster than DP-based methods, but its retrieval accuracy is not as good [11]. Our goal is to obtain a good accuracy with a reasonable amount of computing time. Therefore, the proposed method is based on a highly efficient DP-based method for better accuracy.

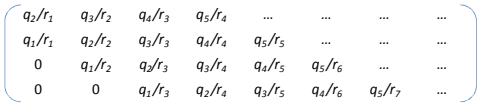
3. THE SHIFTED ALIGNMENT ALGORITHM

This section presents the proposed method to QBT. The method can also be divided into two stages of IOI normalization and similarity comparison. We shall describe these two steps and explain the advantages over the state-of-art QBT methods.

Normalization: In QBT, though the query IOI vector and its target song IOI vector are not necessarily of the same size, the ratio of their tempos should be close to a constant. In other words, the ratios of an IOI element of a query to the corresponding one of the target song should be close to a constant. To take advantage of this fact, we can shift the query IOI vector (relatively to the target song IOI vector) to construct an IOI ratio matrix in order to find the optimum mapping between IOI elements of these two sequences. An example is shown in Fig. 2(a), where the input query IOI vector is represented by $\{q_i,\ i=1\sim m\}$, and the reference IOI vector from the song database by $\{r_j,\ j=1\sim n\}$. As displayed in the figure, the reference IOI vector is shown at the top and the shifted query IOI vectors are shown below. Each element of a shifted query IOI vector is mapped to that of the reference IOI vector in the same column. Take the first shifted query IOI vector as an example, its second element q_2 is mapped to r_1 of the reference IOI vector, q_3 is mapped to r_2, etc. For each matched element pair, we divide the

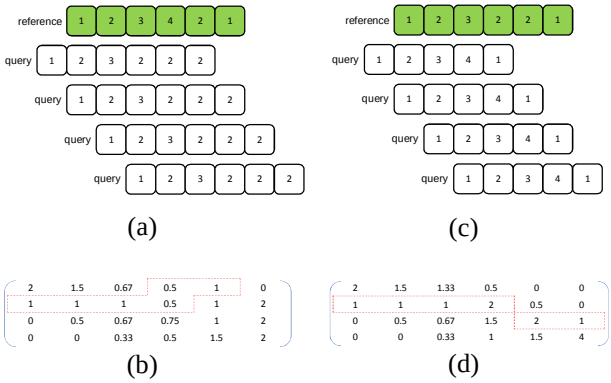


(a)



(b)

Fig. 2. Example of the normalization step of the shifted alignment algorithm: (a) Reference IOI vector and the shifted query IOI vectors. (b) IOI ratio matrix.



(a) (c) (b) (d)

Fig. 3. Typical examples of the shifted alignment algorithm: (a) is an example where the query IOI vector has an insertion; (b) is the corresponding IOI ratio matrix; (c) is another example where the query IOI vector has a deletion; and (d) is the corresponding IOI ratio matrix. The path enclosed by dashed line in (b) and (d) represents the optimal DP path.

query IOI by its mapping reference IOI to construct an IOI ratio matrix M according to the following formula:

$$M_{i,j} = \begin{cases} q_{i_s - i + j + 1} / r_j & \text{, if } 1 \le i_s - i + j + 1 \le \min(m,n) \\ 0 & \text{, otherwise} \end{cases} \quad (3)$$

where the size of the matrix M is min(*m*,*n*)*(i_s + i_e +1). i_s and i_e are the left- and right-shift amount of the query IOI vector, respectively. Fig. 2(b) is the IOI ratio matrix of fig. 2(a). In this example, i_s and i_e are 1 and 2, respectively. Since the length of the query is usually shorter, *m* is generally much less than *n*. Besides, in practice, if the anchor position is the beginning of a song, then we can improve the computation efficiency by truncating a reference IOI vector to a length slightly longer (e.g., 5-element longer) than the length of query IOI vector.

Unlike the equation (1) which requires many different versions of normalized reference IOI vectors for similarity comparison, the proposed approach requires only one-time normalization to generate a single IOI ratio table for computing the similarity. So the proposed approach is guaranteed to more efficient.

Similarity comparison: In order to handle insertions and deletions in a flexible yet robust manner, we propose a dynamic programming method to compute the similarity between the query and the reference IOI vectors. The basic principle is to identify a path over the IOI ratio matrix *M* where the elemental values along the path should be as close as possible to one another. In other words, the accumulated IOI ratio variation should be minimal along the optimal path. Fig. 3 illustrates two typical numeric examples that involve insertion and deletion in the optimal path. In fig. 3(a), query IOI vector and reference IOI vector have the same tempo, so their elements are pretty much the same except that there is an insertion in the query. That is, the fourth element of the reference IOI vector is equally split into 2 elements in the query. Fig. 3(b) is the IOI ratio matrix derived from the fig. 3(a), with the optimal path surrounded by dashed lines. The horizontal direction within the optimal path represent one-to-one sequential mapping between the two vectors without insertion or deletion. The vertical direction within the path indicates an insertion, where the 4th and 5th query IOI elements should be mapped to the 4th reference IOI element. On the other hand, Fig. 3(c) demonstrates an example of deletion where the query misses the 4th onset of the reference vector. Fig. 3(d) shows the corresponding IOI ratio matrix with the optimal path surrounded by dashed lines. The vertical shift of the path indicates a deletion where the 4th query IOI element should be mapped to the 4th and 5th reference IOI elements.

If there is no insertion or deletion in the query, each element along the optimal path should have a value close to its preceding element. With insertion or deletion, then the optimal path exhibits some specific behavior. Therefore our goal is to find the optimal path with minimal variations between neighboring elements in the path, with special consideration for specific path behavior to accommodate insertion and deletion. The variation between neighboring IOI ratio elements can be represented as the deviation between 1 and the ratio of one IOI ratio element to the preceding modified IOI ratio element, which takes into consideration the specific path behavior for accommodating insertion and deletion. The resulting recurrent equation for the optimum-value function $D_{i,j}$ for DP is shown next:

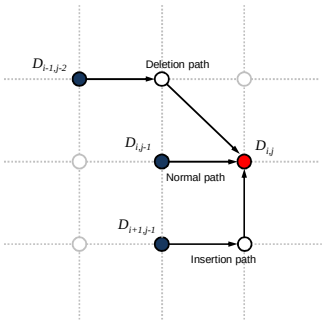


Fig. 4. Constrained DP paths

$$D_{i,j} = \min \begin{cases} D_{i,j-1} + \left| \dfrac{M_{i,j}}{H_{i,j-1}} - 1 \right| \\ D_{i+1,j-1} + \left| \dfrac{M_{i+1,j} + M_{i,j}}{H_{i+1,j-1}} - 1 \right| + \eta_1 \\ D_{i-1,j-2} + \left| \dfrac{M_{i-1,j-1} \cdot M_{i,j}}{M_{i-1,j-1} + M_{i,j}} \cdot \dfrac{1}{H_{i-1,j-2}} - 1 \right| + \eta_2 \end{cases} \quad (4)$$

where $H_{i,j}$ is the modified IOI ratio element with values in the set $\{M_{i,j}, \ M_{i+1,j}+M_{i,j}, \ M_{i-1,j-1}M_{i,j}/(M_{i-1,j-1}+M_{i,j})\}$. The actual value of $H_{i,j}$ depends on the backtrack index in the above formula. Specifically, $H_{i,j}$ will respectively be set to the first, second or third element in the set if $D_{i,j}$ takes its value from the first, second or third row of the above recurrent formula. The first row in the formula indicates one-by-one sequential mapping of the query and the reference IOI. The second row considers the case when the user commits an insertion error by taking one note as two, with the addition of a constant η_1 as its penalty. The third row considers the case when the user commits a deletion error by taking two notes as one, with the addition of a constant η_2 as its penalty. Fig. 4 illustrates these three conditions with three allowable local paths in the DP matrix D. Note that equation (4) does not consider some special cases of n-to-1 insertion or 1-to-n deletion when n is greater than 2. We can easily modify the equation in order to take such considerations, but we choose not to do so since these special cases rarely occur. Moreover, we want to keep the formula simple for straightforward implementation and better efficiency.

The degree of similarity between two IOI vectors can thus be determined from the matrix D. The strategy compares the elements in the corresponding positions of the last non-zeros element in each row of the matrix M. For example, if the DP matrix D is derived from the IOI ratio matrix M in Fig. 2(b), we need to compare the next-to-last element of the first row with the last element of the other rows in D. The optimal cost is the minimal value of these elements. The size of the DP matrix is $\min(m,n)*(i_s+i_e+1)$, which is less than the size $(m*n)$ of the DP algorithms in [6], [7], [9]. In addition, our algorithm can be easily extended to the QBT system with "anywhere" anchor positions by setting the i_e to the length of the reference IOI vector.

4. PERFORMANCE EVALUATION

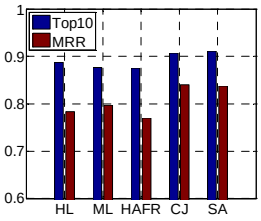
To evaluate the proposed method, we design 3 experiments and compare the performance with that of the state-of-art algorithm. The first experiment compares the recognition rate with algorithms in MIREX QBT task. The second experiment compares their computing speeds. The third experiment demonstrates the robustness of the proposed method using a larger dataset. These experiments are described in the following sub-sections.

4.1 MIREX QBT Evaluation Task

We have submitted our algorithm to the 2012 MIREX QBT task [12], which involves two subtasks for symbolic and acoustic inputs, respectively. Because the onset detection of acoustic input is not the focus of this paper, the following experiments only consider the case of queries with symbolic input. There are 2 datasets of symbolic input, including Jang's dataset of 890 queries (with groundtruth onsets to be used as the symbolic input) and 136 monophonic MIDIs, and Hsiao's dataset of 410 symbolic queries and 143 monophonic MIDIs. The queries of both datasets are all tapped from the beginning of the target song. These datasets are published in 2009 and can be downloaded from the MIREX QBT webpage. The top-10 hit rate and the mean reciprocal rank (MRR) are used as the performance indices of a submitted QBT method. Fig. 5 shows the performance of 5 submitted algorithms, with (a) and (b) are respectively the results of Jang's and Hsiao's datasets. Out of these five submissions, "HL" and "ML" do not have clear descriptions about their algorithms in the MIREX abstracts. Therefore, these 2 algorithms are not included in the experiments in section 4.2 and 4.3. "HAFR" is the implementation of [9], which claimed that its results outperformed other submissions, including the methods of [5] and [6], in MIREX 2008. The algorithm "CJ" is an improved version of [6]. The submission of "SA" is the proposed algorithm in this paper.

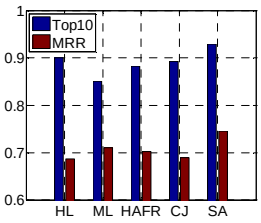
As shown in fig. 5(a), our algorithm outperforms almost all the other submissions except for the MRR in Jang's dataset where our submission is ranked the second. In fact, the MRR of our algorithm is only 0.3% lower than that of "CJ". On the other hand, the top-10 hit rate of our submission is 0.3% higher than that of "CJ". So the performances of "CJ" and "SA" are very close in this dataset. From fig. 5(b), it is obvious that our algorithm simply outperforms all the other submission in both MRR and top-10 hit rate. As a whole, the proposed method obtains good results in MIREX QBT contest.

Algorithm	Top 10	MRR
HL	0.888	0.784
ML	0.876	0.797
HAFR	0.876	0.770
CJ	0.908	***0.840***
SA	***0.911***	0.837



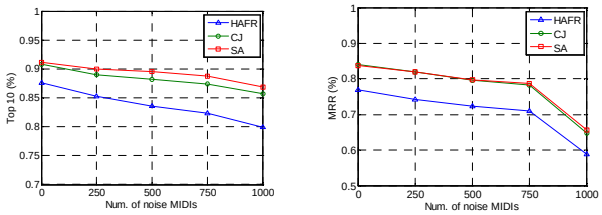
(a) Result 1: Jang's dataset

Algorithm	Top 10	MRR
HL	0.900	0.687
ML	0.851	0.711
HAFR	0.883	0.702
CJ	0.893	0.690
SA	***0.929***	***0.745***

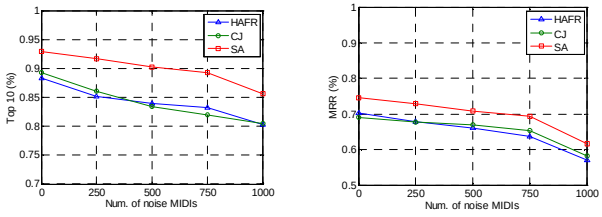


(b) Result 2: Hsiao's dataset

Fig. 5. Results of MIREX QBT evaluation task



(a) Result 1: Jang's dataset



(b) Result 2: Hsiao's dataset

Fig. 6. Results of the performance versus database sizes. (a) is the performance of top-10 hit rate (left subplot) and MRR (right subplot) using Jang's dataset. (b) is the performance of top-10 hit rate (left subplot) and MRR (right subplot) using Hsiao's dataset.

4.2 Evaluation of Computation Efficiency

In this experiment, we want to compare the efficiency of several QBT algorithms. We implemented three submissions (including ours) to 2012 MIREX QBT tasks in C language. The "ML" and "HL" algorithms were not included in this experiment due to the lack of clear descriptions about their algorithms in the MIREX abstracts. The experiment was conducted on a PC with an AMD Athlon 2.4GHz CPU and 1G RAM. Each algorithm was repeated 10 times over Jang's dataset to obtain the average computing time of a single query. The results are shown in Table 1 which indicates that our algorithm is at least 3 times faster than the other two algorithms. This is due to the fact that our algorithm has an efficient way of normalization for IOI vectors (as described in section 3), leading to a smaller table for DP optimal path finding.

Algorithm	Avg. time (ms)
HAFR	421
CJ	213
SA	65

Table 1. Speed comparison of QBT algorithms

From these two experiments, we can claim that our algorithm strike a good balance between the recognition rate and computation efficiency.

4.3 Experiment with Larger Databases

The MIREX QBT datasets are well organized for QBT research. However, both datasets contain song databases of slightly more than 100 songs. These small database sizes lead to high accuracy for all submissions in MIREX QBT task. Therefore, we designed another experiment to demonstrate how the performance varies with the dataset sizes. We collected 1000 MIDIs which are different from the MIDIs in the MIREX QBT datasets. And we enlarge the original databases by adding 250 noise MIDIs each time, and evaluate the performance in both MRR and top-10 hit rate.

Fig. 6 shows the experimental results. As the number of noise MIDIs increases, the recognition rate of each algorithm gradually decreases. In Jang's dataset of the fig. 6(a), the top-10 hit rate of "SA" is the best among all algorithms (left subplot). However, the MRR of "SA" and "CJ" are very close and the value of one is slightly higher than the other in different number of noise MIDIs (right subplot). In fig. 6(b), our algorithm notably outperforms the others in both top-10 hit rate (left subplot) and MRR (right subplot). It is interesting to note that the decay of the top-10 hit rate of "SA" is slower than the others in both datasets, especially in Jang's dataset. This indicates that our algorithm has better resistance to these noise MIDIs in top-10 hit rate. In both datasets, "SA" still had >85% top-10 rate and >60% MRR. Therefore we can conclude that the proposed method is more robust in dealing with a large song database.

5. CONCLUSION

In this paper, we have proposed a shifted-alignment algorithm for QBT by constructing an IOI ratio matrix, in which each element is the ratio of relative IOI elements of the query and a reference song. The similarity comparison is based on DP to deal with possible insertions and deletions of query IOI vectors. We evaluated the performance of the proposed method with two datasets. The experimental results showed that our algorithm exhibited

an overall better accuracy than other submissions to 2012 MIREX query-by-taping task. Moreover, the computation time is at least 3 times faster than others. We also conducted an experiment to demonstrate that our algorithm performs better and more robustly than other existing QBT algorithms in the case of large databases. In particular, our algorithm has a top-10 hit rate larger than 85% and MRR larger than 60% in both databases when the number of noise MIDIs is as high as 1000.

Although the proposed method performs well in the experiments, the recognition rate still has room for further improvement, especially in the case of "anywhere" anchor position, that is, the user is allowed to start tapping from anywhere in the middle of a song. From the experimental results, we can observe that each algorithm has its strength and weakness in dealing with different queries and database songs. Therefore, one direction of our immediate future work is to find an optimal way to combine these methods for better accuracy.

6. ACKNOWLEDGEMENT

This study is conducted under the "NSC 102-3114-Y-307-026 A Research on Social Influence and Decision Support Analytics" of the Institute for Information Industry which is subsidized by the National Science Council.

7. REFERENCES

[1] R. B. Dannenberg, W. P. Birmingham, B. Pardo, N. Hu, C. Meek, and G. Tzanetakis: "A Comparative Evaluation of Search Techniques for Query-by-Humming Using the MUSART Testbed," *Journal of the American Society for Information Science and Technology* (JASIST), vol. 58, no. 5, pp. 687–701, 2007.

[2] J.-S. Roger Jang, H. R. Lee, and M. Y. Kao: "Content-based Music Retrieval Using Linear Scaling and Branch-and-bound Tree Search," *IEEE International Conference on Multimedia and Expo*, pp. 289-292, 2001.

[3] G. Eisenberg, J. Batke, and T. Sikora: "Beatbank - an MPEG-7 Compliant Query by Tapping System," *116th Convention of the Audio Engineering Society*, Berlin, Germany, pp.189-192, May 2004.

[4] G. Eisenberg, J. M. Batke, and T. Sikora: "Efficiently Computable Similarity Measures for Query by Tapping System," *Proc. of the 7th Int. Conference on Digital Audio Effects* (DAFx'04), October, 2004.

[5] R. Typke, and A. W. Typke: "A Tunneling-Vantage Indexing Method for Non-Metrics," *9th International Conference on Music Information Retrieval*, Philadelphia, USA, pp683-688, September 14-18, 2008

[6] J.-S. Roger Jang, H. R. Lee, C. H. Yeh: "Query by Tapping A New Paradigm for Content-Based Music Retrieval from Acoustic input," *Second IEEE Pacific-Rim Conference on Multimedia*, pp590-597, October, 2001

[7] G. Peters, C. Anthony, and M. Schwartz: "Song Search And Retrieval by Tapping," *Proceedings of AAAI'05 Proceedings of the 20th national conference on Artificial intelligence*, pp. 1696-1697, 2005

[8] G. Peters, D. Cukierman, C. Anthony, and M. Schwartz: "Online Music Search by Tapping," *Ambient Intelligence in Everyday Life*, pages 178–197. Springer, 2006.

[9] P. Hanna and M. Robine: "Query By Tapping System Based On Alignment Algorithm," *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP), pp. 1881-1884, 2009.

[10] B. Kaneshiro, H. S. Kim, J. Herrera, J. Oh, J. Berger and M. Slaney. "QBT-Extended: An Annotated Dataset of Melodically Contoured Tapped Queries," *Proceedings of the 14th International Society for Music Information Retrieval Conference*, Curitiba, Brazil, November, 2013.

[11] L. Wang: "MIREX 2012 QBSH Task: YINLONG's Solution," Music Information Retrieval Evaluation eXchange 2012.

[12] The Music Information Retrieval Evaluation eXchange evaluation task of query by tapping: http://www.music-ir.org/mirex/wiki/2012: Query_by_Tapping