A WWW-based Melody Retrieval System

Tomonari Sonoda   Masataka Goto   Yoichi Muraoka

School of Science and Engineering, Waseda University
3-4-1 Ohkubo Shinjuku-ku, Tokyo 169-8555 JAPAN
sonoda@muraoka.info.waseda.ac.jp, goto@etl.go.jp, muraoka@muraoka.info.waseda.ac.jp

ABSTRACT: This paper describes a WWW-based melody retrieval system which takes a sung melody as a query and retrieves the song's title from a music database. In previous works, the pitch information was mainly used as a search clue while the span information was not used effectively, and it was difficult to improve the matching accuracy by using only the pitch information. We therefore propose the following two methods for effective matching: a method for obtaining search clues with the maximum quantity of information from pitch and span and a method for effectively reducing the number of answer candidates.

1 Introduction

WWW-based melody retrieval services that utilize a sung melody as a query require a reliable retrieval method that is robust enough to deal with anonymous users' inputs.

The main issue in building a melody retrieval system is that an input melody can consist of various errors which are caused by the uncertainty of the user's memory or by the user's singing ability. Those errors are usually tolerated by using several thresholds and converting the input melody into approximate relative-value sequences of pitch change (the pitch difference between adjacent notes) and of span change (the inter-onset interval ratio of adjacent notes). For instance, an approximate relative-pitch value is expressed by using three characters (U (p), D (own), and E(qual)), and the sequence "do-re-mi-mi-re-do" is converted into "X-U-U-E-D-D" ("X" indicates the first note which does not have a relative value). These approximate sequences are then utilized in the matching process which compares the input with all songs in a database.

Previous melody retrieval systems [Kageyama et al., 1993; Ghias et al., 1995] utilized static heuristic thresholds to obtain these approximate sequences. Since it was difficult to determine the optimum thresholds for all songs, those systems did not take advantage of the span information which can potentially be useful. Moreover, the public use of music databases over the network was not considered.

We therefore propose the following two methods for effective DP matching. The first method, *dynamic threshold determination*, can obtain the approximate relative-value sequence of pitch/span with the maximum quantity of information by determining thresholds dynamically. The second method, *coarse-to-fine matching*, effectively reduces the number of answer candidates by considering the trade-off between coarse matching and fine matching. Coarse matching can tolerate input errors but it can not reduce the number of answer candidates while fine matching does the opposite.

In our experiment, the matching accuracy of our system was 98 % when we used 112 inputs from 12 subjects, which is high enough for melody retrieval services over the network.

2 WWW-based Melody Retrieval System

Figure 1 is an illustration of our retrieval system. The system consists of a database server and JAVA Applet clients. This section describes each process of them and proposes two methods for effective matching.

2.1 User-side Client

The main function of each client is to extract the pitch and span information from a user's input. After A/D conversion, the client detects the frames that have voiced sounds and determines the pitch and span values of each note during those frames. The client converts the pitch and span values into the relative-values of pitch and span and transmits the relative-value sequences of pitch and span to the server. The client then shows a list of song titles received from the server.

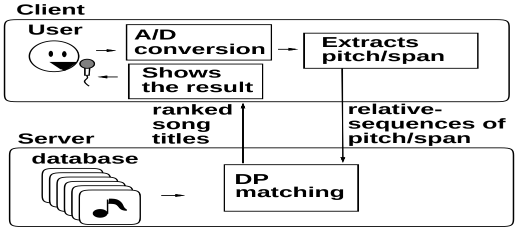

Figure 1: WWW-based melody retrieval system.

The process which the client uses is as follows.

- **Input a sung melody** - The client allows a user to sing from an arbitrary part of a song and to choose an arbitrary key and tempo. The client assumes that each input note begins with a voiceless consonant and ends with a vowel (e.g. ta-ta-ta, cha-cha-cha).
- **Detect voiced sound frames** - The length of each frame is defined as 16 msec. On the assumption that environmental sounds are not included in the input, the client regards the frames with enough amplitude as voiced sound frames.
- **Estimate pitch and span values** - The span of each note is defined as the inter-onset interval, which is the number of frames between the adjacent onset times. The onset time is the first frame of the successive voiced sound frames.

The pitch of each note is defined as the highest pitch value of all the frames during its span. We use FFT to estimate the pitch value of each frame.

- **Determine relative-pitch/span values** - The relative-pitch value is the normalized pitch difference between the adjacent notes in which a semitone difference is normalized to the value of 100. On the other hand, the relative-span value is the ratio of the span value to its previous value and is represented by percentage.

2.2 Song Retrieval Server

The main function of the database server is matching. The server receives an input melody from a client and ranks song titles in an order based on the distance calculated by DP matching between the input melody and each song of its music database. The server then transmits the list of the ranked song titles to the client as a matching result.

The process which the server uses is as follows.

- **Music Database** - The database consists of a number of songs, which can be inputted by singing. For each song, the database stores its title and the relative-value sequences of the pitch and span.
- **DP matching** - The distance between the input melody and each database song is the sum of the pitch and span distances. The pitch and span distances are respectively calculated by DP matching that utilizes the approximate relative-pitch/span values categorized by several thresholds.

2.3 Methods for Effective Matching

The most important point in the whole process is determining the thresholds to categorize the relative-pitch/span values since it directly has a great influence on the accuracy of the matching. Because previous methods utilized static heuristic thresholds, they had the following problems.

1. The span information was not utilized effectively since it was difficult to determine the appropriate thresholds for categorization.
2. It was difficult to reduce the number of answer candidates because several songs in a large database tended to have the same patterns of the approximate relative-value pitch/span sequences.

In order to solve these problems, we propose the following two methods.

1. **Dynamic Threshold Determination**- This method is used to determine the optimum thresholds by utilizing the histogram of relative-pitch/span values of all the songs in the database.
2. **Coarse-to-Fine Matching**- This method is used to effectively reduce the number of answer candidates by gradually increasing the number of categories of relative-values used in the DP matching.

These methods are respectively described in the following two sections.

3 Dynamic Threshold Determination

This section describes the problem found in threshold determination and proposes the algorithm of the *dynamic threshold determination* method.

3.1 Thresholds for Categorization

Figure 2 and 3 respectively illustrates the histogram of relative-pitch/span values that appear in all of the database songs we have utilized in this study. The x-axis shows the relative values; the relative-pitch value is normalized so that a semitone difference has the value of 100, and the relative-span value is represented in percentage. The y-axis shows the frequency of the relative-values.

Previous systems [Kageyama et al., 1993; Ghias et al., 1995] mainly utilized three categories for obtaining approximate relative-pitch/span values such as U(p), E(qual), and D(own) for pitch, and L(onger), E(qual), and S(horter) for span. [Kageyama et al., 1993] utilized pitch thresholds of +100 and -100 and span thresholds of 50 percent and 200 percent. While the pitch thresholds, shown at the top of Figure 2, divided the relative-values equally, the span thresholds, shown at the top of Figure 3, did not divide the span values successfully. Since most relative-span values were categorized in the middle part (E), several database songs had similar sequences, and the span information was not effectively used for the search clue. It was, however, difficult to heuristically determine the optimum thresholds especially for span.

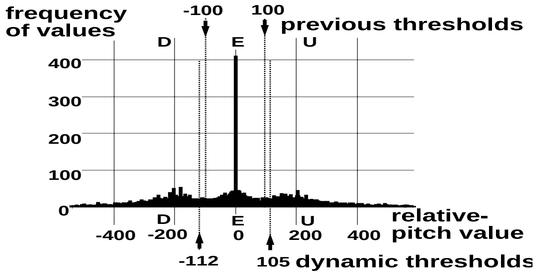

Figure 2: Histogram of the relative-pitch values.

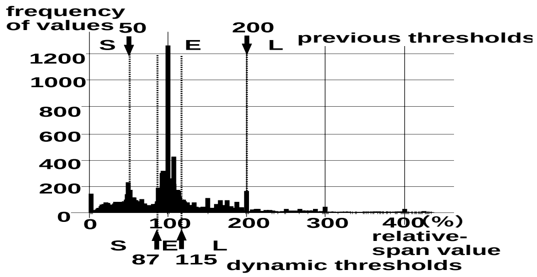

Figure 3: Histogram of the relative-span values.

We therefore propose a dynamic threshold determination method which can obtain optimum thresholds dynamically. Those dynamic thresholds for pitch/span are shown at the bottom of Figure 2 and 3. This method considers such histograms of all the songs in the database in order to determine the thresholds so that each category has an equal amount of relative-value frequencies. Since each pitch/span value has the maximum quantity of information, both sequences are able to be utilized effectively for search clues. Moreover, this method facilitates creating any number of categories in order to generate approximate relative-value sequences. If three categories are not enough to reduce the number of answer candidates, more categories can be utilized. This enables to use coarse-to-fine matching which is described in the section **4**.

3.2 Algorithm for Dynamic Threshold Determination

The algorithm for dynamic threshold determination is as follows.

1. Make the relative-pitch/span histogram like Figure 2 or 3.
2. Count all of the relative-pitch/span values in all of the database songs, and let it be Sum.
3. Assume that the number of category is $CategoryNum$, and that the average number of values that each category has is defined as $Ave = Sum/CategoryNum$.
4. Determine $threshold(j),\ (j = 0, 1, ..., CategoryNum - 1)$ so that the number of frequencies in each category is as equal to Ave as possible.

With these thresholds, the relative-pitch/span values of the input and each database song are converted into approximate values.

4 Coarse-to-Fine Matching

This section describes the problem in reducing the number of answer candidates and proposes the algorithm of the *coarse-to-fine matching* method.

4.1 Answer Candidates Reduction

If relative-pitch/span values are coarsely categorized into three parts, the more songs a database has, the more difficult to reduce the number of answer candidates since the same melody patterns are likely to appear in several songs. On the other hand, if too many categories are used for finer matching, the retrieval result is easily affected by the input errors and fluctuations.

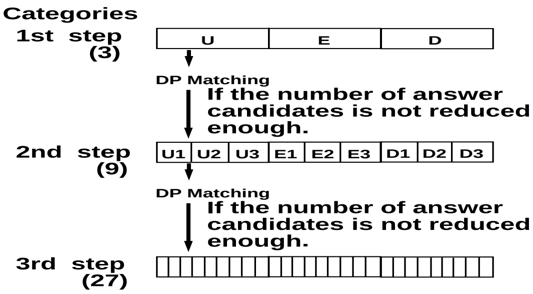
Categories
1st step
(3)
U
E
D
DP Matching
If the number of answer candidates is not reduced enough.
2nd step
(9)
U1
U2
U3
E1
E2
E3
D1
D2
D3
DP Matching
If the number of answer candidates is not reduced enough.
3rd step
(27)

Figure 4: Example of coarse-to-fine matching.

We propose a method which takes advantage of both coarse matching and fine matching. We call this method *coarse-to-fine matching* since the basic idea came from the pattern matching technique for images that takes coarse matching first and then takes finer matching after that. Figure 4 shows an example of the method used for pitch. In converting a relative-pitch sequence into an approximate one, our system first uses the three categories (U, E, and D) for the coarse matching. It then repeats the finer matching, and increases the number of categories until the number of answer candidates is reduced to one.

4.2 Coarse-to-Fine Matching Algorithm

We assume that input relative-pitch/span sequences Q_p/Q_s are converted into approximate relative-value sequences QR_p/QR_s, and that relative-pitch/span sequences of a database song S_p/S_s are similarly converted into SR_p/SR_s. Our system calculates the distance between QR_n $(n = p, s)$ and each SR_n from the database songs by DP matching. If the number of answer candidates are not reduced enough, the system prunes several songs that are ranked lower in the matching, and converts the pitch/span sequences of the higher-ranked songs into finer sequences by increasing the number of categories. If the number of categories is increased M times such as $CategoryNum(0), CategoryNum(1), ...CategoryNum(M - 1)(CategoryNum(i) < CategoryNum(i + 1), i = 0, 1, 2, ..., M - 1)$, the coarse-to-fine matching algorithm is as follows.

1. Set $i = 0$.
2. To generate the $QR_n(i)$ and $SR_n(i)$, each relative value in Q_n and S_n is categorized into $CategoryNum(i)$ parts by the set of dynamic thresholds $DTH(i)$.
3. For all database songs, the distance D_n between each QR_n and SR_n is calculated and ranked in order of distance.
4. Take $N(i)(N(i) > N(i + 1))$ songs from the top of the ranked songs.
5. If several songs from $N(i)$ songs have the same values of D_n, the system assumes that the number of answer candidates have not been successfully reduced. In that case, it increases i and repeats the process from 2. to 4. while $i < M$ or the number of answer candidates are not reduced enough. Otherwise, if all D_n from $N(i)$ differ from one another, the answer candidate is successfully found and it stops the coarse-to-fine matching method.

5 Experiments and Results

In this section, we evaluate the two proposed methods and verify their effectiveness. We then evaluate the overall performance of our system.

5.1 Experimental Conditions

Our system consisted of a database retrieval server on WWW, which was written in Java, and user-side JAVA Applet clients with a specially designed Netscape Plug-in for recording an input sung melody. In our experiment, the server had a 200 song database which contained various genres of songs such as rock, pop, and folk music.

We tested our system using 112 inputs from 12 subjects (8 men and 4 women). The matching accuracy was evaluated in 3 ways; (1) used only span for retrieval, (2) used only pitch, (3) used both pitch and span. The recording time was limited within 8 seconds and the sampling rate was 8 kHz. FFT with 512 points a frame (16 msec) was utilized to convert the acoustic data into melody data.

5.2 Evaluation for Dynamic Threshold Determination

We compared the performance of our dynamic thresholds with previous thresholds [Kageyama et al., 1993] that used pitch thresholds of +100 and -100 and span thresholds of 50 percent and 200 percent. Table 1 shows the result when we utilized three categories to determine the dynamic thresholds. The matching accuracy for the dynamic thresholds obviously improved when we used the span information, and the accuracy of both pitch and span information also improved.

Table 1: Performance of dynamic thresholds.

	pitch	span	pitch and span
previous thresholds	47.3%	13.4%	90.2%
dynamic thresholds	49.1%	35.1%	97.3%

112 inputs for 200 songs.

5.3 Evaluation for Coarse-to-Fine Matching

We tested the robustness against the input errors by comparing normal DP matching with coarse-to-fine matching. The normal DP matching utilized 27 categories for the approximate relative-pitch/span values, and the coarse-to-fine matching utilized 3 steps to increase the categories (3, 9, 27). We intentionally added errors to the input melodies so that each song had 2 pitch errors (added +200 or -200 to 2 relative-pitch values that were randomly selected) and 2 span errors (changed the ratio of the relative-span to 4 times of its value or to a quarter of its value). Table 2 shows the result with no input errors, and Table 3 shows the result with input errors. The tables show that the coarse-to-fine matching was robust enough to tolerate the input errors.

Table 2: Matching accuracy with no intentional errors.

	pitch	span	pitch and span
normal	87.5%	82.1%	97.3%
coarse-to-fine	83.0%	79.5%	98.2%

112 inputs for 200 songs.

Table 3: Matching accuracy with intentional errors.

	pitch	span	pitch and span
normal	75.9%	34.8%	90.1%
coarse-to-fine	80.3%	50.0%	94.6%

112 inputs for 200 songs.

Table 4: Matching accuracy of the system.

	pitch	span	pitch and span
ranked in the top	83.0%	79.5%	98.2%
ranked in top 3	88.4%	88.4%	100.0%

112 inputs for 200 songs.

5.4 Matching Accuracy of the System

The matching accuracy of our system with the two proposed methods is shown in Table 4. The result shows that our system achieved an accuracy high enough for WWW-based melody retrieval services.

6 Conclusion

We described a system that enables a user to retrieve a song's title from a music database by singing via WWW. For an effective DP matching, we proposed two methods, dynamic threshold determination and coarse-to-fine matching, and we achieved higher matching accuracy than the previous methods.

The current system utilized a fixed music database which was prepared in advance. There are, however, a lot of song data that are open to public on the Internet. We therefore plan to build a WWW Search Robot for such song data and we plan to extend the system so that it can retrieve various songs from WWW databases all over the world.

References

[Ghias et al., 1995] Asif Ghias and Jonathan Logan : *Query By Humming – Musical Information Retrieval in an Audio Database*, ACM Multimedia 95, Electronic Proc., 1995.

[Kageyama et al., 1993] T.Kageyama, K.Mochizuki, and Y.Takashima : *Melody Retrieval with Humming*, ICMC Proc., pp.349-351, 1993.