Query by Humming - in Action with its Technology Revealed *

Yunyue Zhu
Courant Institute
New York University
yunyue@cs.nyu.edu

Dennis Shasha
Courant Institute
New York University
shasha@cs.nyu.edu

Xiaojian Zhao
Courant Institute
New York University
xiaojian@cs.nyu.edu

You have a tune lingering in your head for many days, but you don't know where you heard this tune or which song it is from. This demo will show a Query by Humming system that will tell you the name of that song.

Most of the research in pre-existing query by humming systems uses pitch contour to match similar melodies (for example [1]). The user's humming is transcribed to a sequence of discrete notes and the contour information is extracted from the notes. This contour information is represented by a few letters. For example, ("U", "D", "S") represents that a note is above, below or the same as the previous one. The tunes in the databases are also represented by contour information. The edit distance can be use to measure the similarity between two melodies. Unfortunately, it is very hard to segment a user's humming into discrete notes. Some recent work proposes to match the query directly from audio based on dynamic time warping to match the hum-query with the melodies in the music databases. But this quality improvement comes at a price because a brute-force search using DTW is very slow.

The database community has been researching problems in similarity query for time series databases for many years. The techniques developed in the area might shed light on the query by humming problem. In this demo, we treat both the melodies in the music databases and the user humming input as time series. Such an approach allows us to integrate many database indexing techniques into a query by humming system, improving the quality of such system over the traditional (contour) string databases approach. We design special searching techniques that are invariant to shifting, time scaling and local time warping. This makes the system robust and allows more flexible user humming input.

1. OVERVIEW OF THE SYSTEM

The user hum query is segmented into frames of 10ms and each frame is resolved into a pitch. This results in a time series of the pitches. Our music database is made up of a collection of short melodies (10 to 15 notes),each of which is a time series of pitches.

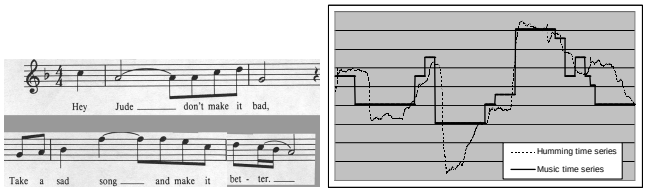

Figure 1: The sheet music of "Hey Jude", the time series of the melody and the hum query

The input query will be matched with each piece of melody in the database. Using similarity measure invariant under shifting and time scaling, we allow the user to hum at different absolute pitches and at different tempos. Using Local Dynamic Time Warping, we can make the matching process allow variations in tempo for each note. Figure 1 shows an example of the time series representations of the humming and the candidate music tune. The dynamic time warping distance between the time series will be used as the similarity measure.

2. DEMO DESCRIPTION

A SIGMOD participant will hum the query melody using a PC microphone. The user will see his humming being displayed as a time series of the pitches within 1 second. The user can listen to the playback humming when the database query is executed. The top K matches of the user hum-query will be returned. The user can listen to the results and see whether they include the target tune. The time series of the user query and the result will be displayed together as in figure 1. The concept of Local Dynamic Time Warping distance will become very obvious by just glancing at the figure. The user might also find that some other tunes in the results sound similar to his target tune. If the user doesn't get the query result he wants, he can try again. As an option, the user can also change the warping width for the DTW distance and repeat the query to improve recall.

3. REFERENCES

[1] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith. Query by humming: Musical information retrieval in an audio database. In *ACM Multimedia 1995*, pages 231–236, 1995.

*Work supported in part by U.S. NSF grants IIS-9988636 and N2010-0115586.

SIGMOD 2003, June 9-12, 2003, San Diego, CA.