Recursive Geometric Algorithm for Estimating Melodic Similarity

Kjell Lemström and Niko Mikkilä
C-BRAHMS Group, Department of Computer Science
P.O.Box 68 (Gustaf Hällströmin katu 2b)
FIN-00014 University of Helsinki, FINLAND
{klemstro,mikkila}@cs.helsinki.fi

Abstract

This extended abstract gives an overview on a content-based retrieval algorithm for symbolic music, developed in the C-BRAHMS group [1], that took part in the Query by Singing/Humming task of the MIREX 2007 contest. Given two excerpts of symbolically encoded monophonic or polyphonic music, the *query pattern* and the *target music*, the purpose of this algorithm is to find musically relevant occurrences of the query pattern within the target music.

Keywords: MIREX 2007, Melodic Similarity, Geometric Matching

1. Introduction and Background from MIREX 2006

2. Recursive Algorithm

The recursive search algorithm that we submitted to the QBSH task uses a geometric piano-roll representation of music, but it does not maximize the overlapping in the same way as our P3 algorithm does. Instead it performs an exhaustive depth-first search trying to scale the pattern note-by-note to 'fit' the target song, with costs applied to local time-scaling, note duration changes and pitch-shifting. Clearly irrelevant branches are cut with simple heuristics while searching, which keeps the average running time in an usable range, although the worst case time complexity is $O(nm^2)$.

First the algorithm divides the piano-roll representation into tiles that have a height of one MIDI pitch level and a width chosen so that there would not be many notes in one tile. A pointer to the first note that starts in each tile is stored to a table and subsequent notes at the same pitch level are linked together. This tile table is used for hashing: quickly finding notes that start within a specific range in the target music. The tile table size is a compromise between quick lookups and space consumption. We used a static tile length of 100 ms but a more optimal value for each pitch level could be chosen by scanning through the target music.

Next the target music is searched for the best occurrance of the pattern by checking recursively for a match at each note in the music, starting with each note in the pattern. For note T_i in the target music and note P_j in the pattern, the recursive check is started by calculating the expected pitch and starting time interval of the next matching note, or multiple notes when gaps are allowed in the matches. All potentially matching notes are looked up from the tile table, match score is updated and the same check is executed recursively for each of the notes, starting at the next position in the pattern.

The most adjustable part of the algorithm is the way how the following potentially matching notes are picked and scored at each recursion level. This procedure can be weighted by the already matched part of the pattern or it can be done independently for each position. To calculate the expected pitch level, we simply take the pitch interval between P_{j+1} and P_j, and add that to the pitch of T_i. Similarly, the difference between start times of the consecutive notes in the pattern is scaled in proportion to previously matched notes and added to the start time of T_i.

Pitch and tempo shifts are handled by retrieving all notes within a certain range from the expected position: ± 2 pitch levels and the delta time scaled by 0.5 – 2.0. Notes that start outside this area are not considered further at that point of recursion. Each melody line that continues from the retrieved notes is checked recursively and the match scores are updated. Notes that are closest to the expected note position and have similar duration to the corresponding note in the pattern receive the best score. This is done by multiplying together factors derived from all these differences. 1.0 is a perfect match of a note and 0 is a complete mismatch. Therefore the whole pattern has a maximal score of $m - 1$, and match scores are normalized by dividing them with this value.

3. Results and Analysis

In this section we analyze results from the MIREX QBSH contest. More information about the tasks and evaluation methods can be found through task descriptions in the MIREX Wiki. Abstracts from all the participants are published in the result pages.

4. Acknowledgments

Our best thanks to IMIRSEL for organizing MIREX 2007 and to Xiao Wu and Roger Jang for their contributions to the QBSH task.

References

[1] K. Lemström, V. Mäkinen, A. Pienimäki, M. Turkia and E. Ukkonen, "The C-BRAHMS Project," in *ISMIR 2003 Fourth*

Int. Conf. on Music Inf. Retr. Proc., Oct. 2003, pp 237-238, See: http://www.cs.helsinki.fi/group/cbrahms/

[2] E. Ukkonen, K. Lemström and V. Mäkinen, “Sweepline the Music!.” *Computer Science in Perspective — Essays dedicated to Thomas Ottmann*, vol 2598, pp 330-342, Springer-Verlag, 2003