Polyphonic Audio Matching and Alignment for Music Retrieval

Ning Hu, Roger B. Dannenberg and George Tzanetakis

Computer Science Department, Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213-3891 USA
ninghu,rbd,gtzan@cs.cmu.edu

ABSTRACT

We describe a method that aligns polyphonic audio recordings of music to symbolic score information in standard MIDI files without the difficult process of polyphonic transcription. By using this method, we can search through a MIDI database to find the MIDI file corresponding to a polyphonic audio recording.

1. INTRODUCTION

For most pieces of classical or popular music, there are printed scores, audio recordings, and often MIDI files created by manual transcription or score conversion. One of the challenges in Music Information Retrieval (MIR) is to find correspondences among these different music representations. In this paper, we propose a method to match and align polyphonic audio with MIDI data.

Our work is motivated by a Query By Humming (QBH) music retrieval system [1]. In QBH systems, typically audio queries are matched against a symbolic database, because matching to audio is simply too difficult. Therefore, we need symbolic representations (e.g. MIDI) corresponding to audio (e.g. MP3) files. One strategy is to search the web for MIDI files, but using filenames is not a reliable method to insure that the files correspond. One important goal of our work is to search for MIDI files that match audio.

The problem is: Given an audio recording, find the corresponding standard MIDI file (or vice versa) from a database. The matched audio and MIDI files should be as close to the actual score as possible, but the timing or even keys might be quite different. For example the MIDI data may be a "flat" performance using exact tempo markings from a score, while the audio may be an expressive performance by musicians.

A "standard" approach to this problem might be to perform some sort of polyphonic transcription on the music and then use a symbolic score-matching algorithm [2] to compare the query with every target in the database [3]. Unfortunately, accurate polyphonic transcription is yet to be achieved, and the error rates of the best systems are sufficiently high as to make matching difficult in many cases.

We present an alternative in which matching is performed on acoustic features rather than symbolic ones. We directly map MIDI data to corresponding audio features, and use a dynamic time warping algorithm [4] to align the resulting sequences. The result not only tells us the optimal alignment between the audio and MIDI data, but also returns a score reflecting how confident we are to say the audio and MIDI data are truly correspondent.

Our work is closely related to that of Orio and Schwarz [5], who also use dynamic time warping to align polyphonic music to scores. They obtain accurate alignment using small (5.8ms) analysis windows, and use a measure called Peak Structure Distance, which is derived from the spectrum of audio and from synthetic spectra computed from score data. In contrast, we use the chromagram, described below. Another novel aspect of our work is that we have demonstrated success with popular vocal music, in spite of obvious discrepancies between MIDI data and vocal performance. Finally, our work is motivated by MIR rather than score following or alignment.

2. POLYPHONIC AUDIO MATCHING

2.1. The Chroma Representation and Other Features

First, we convert audio data into *discrete chromagrams*: sequences of chroma vectors. The chroma vector representation is a 12-element vector, where each element represents the spectral energy corresponding to one pitch class (i.e. C, C#, D, D#, etc.). To compute a chroma vector from a magnitude spectrum, we assign each bin of the FFT to the pitch class of the nearest step in the chromatic equal-tempered scale. Then, given a pitch class, we average the magnitude of the corresponding bins. This results in a 12-value *chroma vector*. Each chroma vector in this work represents 0.25 seconds of audio data (non-overlapping).

The exact details of the chroma computation concerning how to deal with low-frequency bins that span more than one half-step, whether to average magnitude or sum power, etc., are not critical. Our work differs from the original chroma vector work [6] in that we use linear rather than logarithmic amplitudes.

The reason chroma *might* be good for this application is that the chroma vector depends on the pitch classes of strong partials in the signal. By design, all spectral energy is collapsed into one octave; chroma vectors are not sensitive to spectral shape, yet they are sensitive to prominent pitches and chords. Since we are comparing MIDI data to acoustic data, it is good to focus on pitch classes and more-or-less ignore details of timbre and spectral shape.

Besides the chroma representation, we considered features based on the pitch histogram [7] and mel frequency cepstral coefficients (MFCC). In Section 3, we compare the performance of these features for our task.

0-7803-7850-4/03/$17.00 © 2003 IEEE

A pitch histogram is calculated by performing automatic multiple pitch detection for each frame. The four most prominent detected pitches are converted to pitch class and the results are accumulated in a histogram.

MFCC is a perceptually motivated spectrum representation that is widely used not only in speech recognition but also for music modeling [8]. MFCC are calculated every 23 milliseconds and averaged across every 0.25 seconds. A variant we call NMFCC normalizes the MFCC vectors across each audio file so that they have mean 0 and standard deviation 1.

2.2. From MIDI to Chroma

The second step is to convert MIDI data to chromagrams. Our initial experiments were done conservatively. We first converted MIDI data to audio using a MIDI synthesizer, and then the rendered audio was converted to chromagrams. For these experiments, we used Timidity (http://www.onicos.com/staff/iz/timidity/index.html), which generates audio files from standard MIDI files.

We can also use a direct mapping from MIDI to chromagrams. In Figure 1, we show a score where each whole note is a piano sound with duration of 0.25 second played via MIDI and Timidity. We converted the rendered audio to a chromagram. The comparison between the original score and the generated chromagram demonstrates the obvious relationship between them.

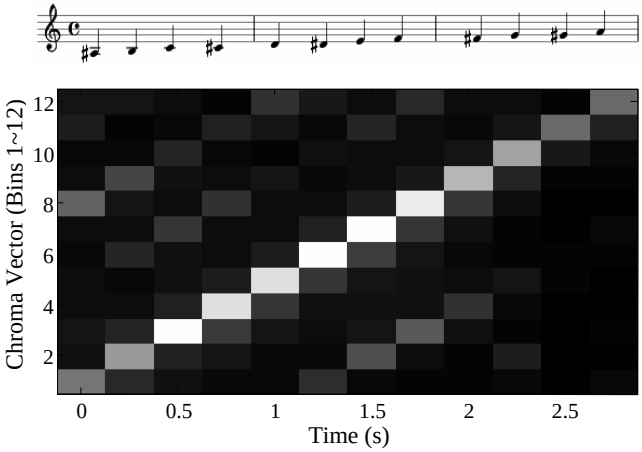


Figure 1. *Relationship between score and chromagram.*

To compute the chromagram directly from MIDI data, we first associate each pitch class with an independent unit chroma vector – the chroma vector with only one element value as 1 and the rest as 0; then, where there is polyphony in the MIDI data, the unit chroma vectors are simply multiplied by the loudness factors, added and normalized.

In addition to timbre and loudness, this approach ignores many other details that would be present in synthesized sound, including envelopes and vibrato. It is particularly hard to map the MIDI data with percussion instruments to chromagrams. Nevertheless, this simplified direct mapping has little impact on the results.

2.3. Matching Audio to MIDI

After computing chroma for audio recordings and MIDI data, we obtain two sequences of vectors. We want to find a correspondence between the two sequences such that corresponding vectors are similar. One way to think about this problem is that we will modify the tempo of the MIDI data in order to obtain the best agreement between the resulting sequences of vectors.

We must first define what "agreement between vectors" means. We first normalize the vectors to have a mean of zero and a variance of one. The normalization reduces differences due to absolute magnitude (loudness), which seems to be a good idea because loudness in MIDI files is rarely calibrated to absolute levels. We then calculate the Euclidean distance between the vectors. The distance is zero if there is perfect agreement. Figure 2 shows a *similarity matrix* where the vertical axis is a time index into the acoustic recording, and the horizontal axis is a time index into the MIDI data. The intensity of each point is the distance between the corresponding vectors, where black represents a distance of zero.

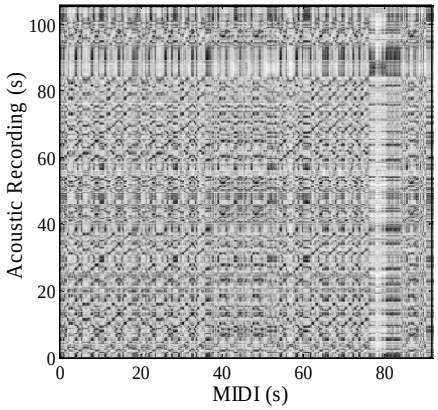


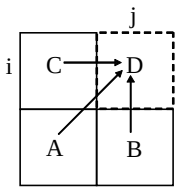
Figure 2. *Similarity matrix for Beatles' "I Will".*

The dark diagonal represents a path where the vectors are near one another. This path is the alignment we are after. Notice that the tempo of the MIDI performance [9] is substantially faster than the audio [10], so the acoustic recording is longer than the MIDI data. Also the repetition at the beginning of the song yields additional off-diagonal paths where the first repeat of the acoustic data matches the second repeat of the MIDI data and vice versa.

Although the path is visually clear in the figure, we need an automated method to locate the path. Alignment is computed using a dynamic time warping (DTW) algorithm. DTW computes a path in a matrix where the rows correspond to one vector sequence and columns correspond to the other. The path is a sequence of adjacent cells, and DTW finds the path with the smallest sum of distances.

For DTW, each matrix cell (i,j) represents the sum of distances along the best path from (0,0) to (i,j). We use the calculation pattern shown in Figure 3 for each cell. The best path up to location (i,j) in the matrix (labeled "D" in the figure) depends only on the adjacent cells (A, B, and C) and the distance between the vectors corresponding to row i and column j. The DTW algorithm requires a single pass through the

matrix to compute the cost of the best path. Then, a backtracking step is used to identify the actual path. We tried other formulations of DTW [11]; however, the difference between the resulting optimal paths is often too subtle to tell which one is best.



$$D = M_{i,j} = \min(A,B,C)+dist(i,j)$$

Figure 3. *Calculation pattern for cell (i,j).*

Using DTW, we can use the similarity matrix in Figure 2 to identify the path shown by the white line in Figure 4. As shown by the path, there is some non-matching MIDI at the beginning and non-matching audio at the end of the song.

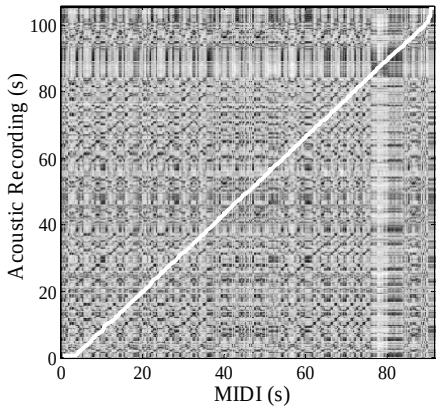


Figure 4. *The optimal alignment path is shown in white over the similarity matrix of Figure 2.*

An alternative to dynamic time warping is the hidden Markov model (HMM), which *might* lead to improved performance. However, HMM requires careful design and training. Since we have achieved good results with a simple model and *no* training, we believe our approach is attractive for a variety of applications.

After computing the optimal alignment between any audio recording and MIDI file, we can easily compute the average distance along the path. This turns out to be a useful feature to tell whether the audio recording and MIDI file represent the same piece of music. On one hand, when matching and alignment are possible, we would expect to see a low average distance along the alignment path. On the other hand, if we use MIDI data that is unrelated to the audio, then even the best path should exhibit a large average distance.

3. RESULTS AND ANALYSIS

3.1. Features Comparison

We chose 10 acoustic recordings of Beatles' songs as queries and rendered their corresponding MIDI files to audio as targets. From these audio files, we computed the following representations: chromagram, pitch histogram, and MFCC.

For each query, the program matches its particular feature sequence against that of every target in the database and returns the average distance along the path. Then the targets are sorted according to the average distance.

Table 1 shows the experimental results for features comparison. The first row "Top 1" tells how many correct matches are ranked first for each feature. The second row "MRR" is the mean of the reciprocals of the ranks of the correct matches (so 1 is best, 0 is worst). Overall, chroma returns the best result, and pitch histogram is the second best, while the results from MFCC and NMFCC are essentially random. This confirms that pitch-based representations, especially chroma, are better choices for this task.

Table 1. *Features Comparison.*

Features	**Chroma**	**Pitch Histogram**	**MFCC**	**NMFCC**
Top 1	9	7	0	1
MRR	0.95000	0.81667	0.21306	0.30361

3.2. Music Retrieval Using Chroma

For music retrieval using chroma, we set up a larger scale experiment with 51 acoustic recordings as queries and 259 MIDI files as targets in the database. Both the queries and targets are Beatles' songs.

The results are good, in spite of the fact that the songs feature vocal and percussion prominently and the music styles are similar (they are all Beatles' songs). If we use classical music as the queries and targets instead, the results should be even better. Figure 5 shows the rank distribution of correct matches from the experiment. 40 out of 51 queries returned the correct match ranked among top 10, and 25 of them are ranked first.

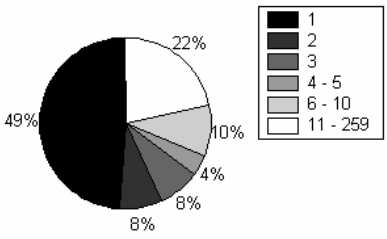


Figure 5. *Rank distribution of correct matches.*

Of course, the smaller the average distance value, the more confident we are to say it is a correct match. Figure 6 shows the distribution histograms of the average distance values on those correct and wrong matches. They are made up of 51 and 13676 data points respectively, and both are normalized to 1 so that the shapes are comparable.

Another feature of the polyphonic audio matching is the ratio between the average distance along the path and the average distance value in the matrix. The average distance along the path is always lower than the average matrix value as indicated by ratio values less than one. However, we found this feature less effective than the average distance value along the path in predicting successful matches.

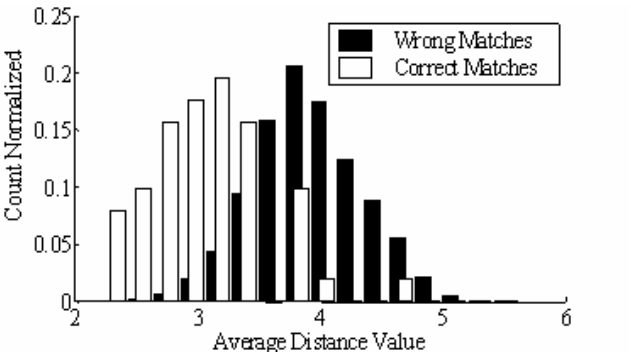


Figure 6. *Distribution histograms of the average distance values on correct and wrong matches.*

4. FUTURE WORK AND CONCLUSIONS

The method we describe in this paper is simple and easy to implement. There are many possible ways to improve the performance. A neural network might output the most likely chroma vector for each distinct polyphonic MIDI frame, and we could replace the simple dynamic time warping with a hidden Markov model. For better ranking of the matches, we can extract various features from the optimal path found by DTW including average distance value along the path, the ratio between the average distance along the path and the average distance value in the matrix, and the smoothness measurement of the path. Machine learning techniques can then be applied to build a better classifier. The current system runs slowly because of the DTW algorithm. We can use some tricks to speed up matching, such as only computing the cells close to the diagonal, as most optimal paths are along diagonal. A more advanced method would be approximate or exact indexing of DTW [12].

On the other hand, we appreciate the directness and simplicity of the current method. Matching polyphonic audio recording of music to symbolic score information is good not only for music search, but it also enables interesting applications, for instance, polyphonic score following, intelligent audio editors and analysis of expressive performance. We discuss those possible applications in more detail in another paper [13].

Many computer music studies and applications focus *solely* on either signal representations (namely audio data) or symbolic representations (specifically scores, MIDI and note lists) of music. Many lower-level tasks, such as beat tracking [14] and fundamental estimation [15], are performed on features extracted from audio. Operations at higher levels rely upon symbolic representations, for example, Music Retrieval based on melody contour and structural analysis [16]. Both the signal and symbolic representations have their advantages and disadvantages. If we can forge links between signal and symbolic representations, we can more fully utilize features and operations at different levels to solve a wider range of problems. In this paper, we suggest one possible way to automatically identify MIDI files that correspond to audio recordings. We believe the resulting combinations of audio and MIDI transcriptions will find many interesting applications.

5. ACKNOLEDGEMENTS

This work was supported by NSF Award #0085945. We would like to thank Greg Wakefield and Mark Bartsch for their chromagram code and fruitful discussions.

6. REFERENCES

[1] Birmingham, W.P., et al. "MUSART: Music Retrieval via Aural Queries", *Proc. 2nd International Symposium on Music Information Retrieval (ISMIR)*: 73-81, 2001.

[2] Bloch, J., & Dannenberg, R.B. "Real-Time Accompaniment of Polyphonic Keyboard Performance", *Proc. 1985 International Computer Music Conference (ICMC)*: 279-290, ICMA.

[3] Pickens, J., et al. "Polyphonic Score Retrieval Using Polyphonic Audio Queries: A Harmonic Modeling Approach", *Proc. 3rd ISMIR*, 2002.

[4] Sankoff, D., & Kruskal, J.B., *Time Warps, String Edits, and Macromolecules, the Theory and Practice of Sequence Comparison*, Addison-Wesley, 1983.

[5] Orio, N., & Schwarz, D. "Alignment of Monophonic and Polyphonic Music to a Score", *Proc. 2001 ICMC*: 155-158, ICMA.

[6] Bartsch, M., et al. "To Catch a Chorus: Using Chroma-Based Representations For Audio Thumbnailing", *Proc. 2001 WASPAA*, IEEE.

[7] Tzanetakis, G. et al. "Pitch Histograms in Audio and Symbolic Music Information Retrieval", *Proc. 3rd ISMIR*: 31-38, 2002.

[8] Logan, B. "Mel Frequency Cepstral Coefficients for Music Modeling", *Proc. 1st ISMIR*, 2000.

[9] "Doc Doc" Petro, Beatles' "I Will" [Standard MIDI File], http://home.texoma.net/~docdoc, 1996.

[10] Beatles. "I Will", *White Album (Disc 1 of 2)*: Apple Records, 1968.

[11] Hu, N., & Dannenberg, R.B. "A Comparison of Melodic Database Retrieval Techniques Using Sung Queries", *Proc. Joint Conference on Digital Libraries* (*JCDL) 2002*, ACM.

[12] Keogh, E. "Exact Indexing of Dynamic Time Warping", *Proc. 28th International Conference on Very Large Database (VLDB)*, VLDB Endowment, 2002.

[13] Dannenberg, R.B. & Hu, N. "Polyphonic Audio Matching for Score Following and Intelligent Audio Editors", *Proc. 2003 ICMC*, ICMA, (to appear).

[14] Goto, M. & Muraoka, Y. "Music Understanding at the Beat Level: Real-Time Beat Tracking of Audio Signals", *Computational Auditory Scene Analysis*, Lawrence Erlbaum Associates, New Jersey, 1998.

[15] Rabiner, L. "On the use of autocorrelation analysis for pitch detection", *IEEE Trans. ASSP*, 25 (1): 24-33, 1977.

[16] Dannenberg, R.B. and Hu, N. "Pattern Discovery Techniques for Music Audio", *Proc. 3rd ISMIR*: 63-70, IRCAM, 2002.