Aligning Musical Audio with Symbols: A Case Study in Western Classical Music[*]

Iman S. H. Suyoto and Alexandra L. Uitdenbogerd
School of Computer Science and Information Technology, RMIT
GPO Box 2746V, Melbourne, Victoria 3001, Australia
[iman.suyoto,sandrau]@rmit.edu.au

Abstract

Searching recorded music using musical queries such as note sequences is a difficult problem due to the limitations of transcription technology. Thus, most current music retrieval systems use collections in symbolic rather than audio format. This paper is about an early investigation in the problem of audio music retrieval using symbolic data. Our novel method consists of five stages: transcription (of audio to symbolic data), noise removal, bass-part extraction, standardisation, and alignment. Our experimental results show that it is possible to match audio music with its symbolic equivalent representation.

1 Introduction

The way that users find music has changed considerably in the last decade, with much music available on-line. While music is often located by artist name, song title, or lyrics, there are many times when users wish to find a particular tune by content.

In the field of music information retrieval, the desired objective is the ability to retrieve music from an audio collection given a query, representing a portion of the music, that is either sung, played, or otherwise encoded. To date, most practical systems that retrieve music given a melody fragment as a query, are restricted to music collections that are in a symbolic format such as MIDI. The major difficulty lies in the automatic transcription of recorded music, due to the complexity of the audio signal when there is more than one note sounding simultaneously, multiple musical instrument timbres, and a complex acoustic environment.

There have been some previous attempts to solve the audio retrieval problem. Notably, the Shazam[1] system successfully uses audio signatures to retrieve specific recordings regardless of the audio quality of the version presented to the system [20]. The system does not find cover or other alternate versions of music, however. Other attempts to retrieve audio have exploited long-term structure, as represented by variation in loudness or volume [7] or timbral texture [2]. This allowed multiple versions to be matched. However, the techniques were tested on collections of less than one hundred pieces. Matching MIDI files with (monophonic) hummed melody queries has been shown to be more successful [4, 9, 10, 15].

In this work we test the feasibility of finding polyphonic audio recordings of music given a symbolic representation. Instead of using low-level audio features such as Mel-frequency cepstral coefficients,[2] we use the high-level music feature called *pitch*.

In brief, our technique makes use of automatic transcription and dynamic programming. It is further described in Section 2. How we evaluate the effectiveness of our approach is discussed in Section 3. In Section 4, we discuss our experimental setup, and the results are presented in Section 5. We finally present our conclusions and suggestions for future work in Section 6.

We discovered that for a collection of about 2 000 pieces, the rank of most queries is quite acceptable

[*]This paper is currently in submission to DASFAA 2007.

[1]See *http://www.shazam.com*.

[2]Audio features are not explained in this paper. They are described elsewhere [1, 12, 16].

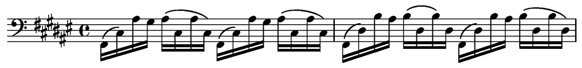

Figure 1. *J. S. Bach's "BWV 1007 Prelude."*

when given a query representing the entire musical work. Effectiveness degrades as the query shortens, however. Nevertheless, we have shown that it is possible to use current transcription technology to retrieve answers to queries.

2 Matching

The matching process is composed of five stages: transcription (of audio to symbolic data), noise removal, bass-part extraction, standardisation, and alignment. These are discussed below.

Transcription

In the transcription stage, the audio files in the collection are transcribed so that the symbolic data are obtained.The transcriber we use is TS-AudioToMidi 3.30.[3] The transcription results are saved in the Standard MIDI file format.[4].

State-of-the-art transcription technology still produces musical symbols with much noise in the form of extraneous notes, particularly when the audio consists of more than one timbre and is polyphonic. As an example, one of the tracks in our collection is J. S. Bach's "BWV 1007 Prelude" performed by Carrai. The first two bars of the track are shown in Figure 1. The transcription result is shown in Figure 2. It contains many extraneous notes.

Noise removal

As mentioned in Section 2, current transcription technology still leaves much noise in its symbolic output. Therefore, we need a noise removal procedure to help us generate a retrievable sequence.

We use a noise removal heuristic that depends on the statistics of a tune, particularly that of pitch.

[3]See *http://audioto.com/eng/aud2midi.htm.*

[4]See *http://www.midi.org/about-midi/abtmidi2.shtml.*

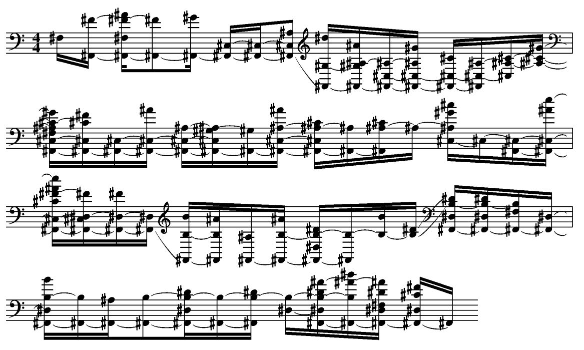

Figure 2. *The first few notes of the transcription result of J. S. Bach's "BWV 1007 Prelude" performance by Carrai that corresponds to the first two bars of the score shown in Figure 1. Notes are quantised to semiquavers/sixteenth notes for reading comfort.*

The result of this process is not intended for listening purposes but for producing "clean" symbolic sequences that can be used for retrieval.

The first step in filtering a tune is removing notes that are perceived as "too soft." A softness threshold V_t, where V_t is a valid MIDI velocity value, is used here. Any notes softer than V_t are discarded. The next step is building a histogram of pitches. As an example, the histogram showing the number of soundings of pitches in the transcription result of J. S. Bach's "BWV 1007 Prelude" performed by Carrai (excerpt shown in Figure 2) is plotted in Figure 3. By building a histogram, the pitch median can be determined. Let us call the pitch median $\tilde{P}$. For each pitch p, if $p > \tilde{P}$, it is removed. The filtered result is shown in Figure 4.

This heuristic was developed through observation of typical transcription results. Harmonics due to instrument timbre and reverbation effects tend to lead to additional notes being transcribed along with the actual melody pitch. The problem of finding melody lines is still only partially solved.[5]

Bass-part extraction

From every transcribed tune, a monophonic sequence that represents the tune is extracted. Past work [19] shows that the ALL-MONO algorithm is

[5]Research on this area is active as described in various work [6, 11, 13] and elsewhere.

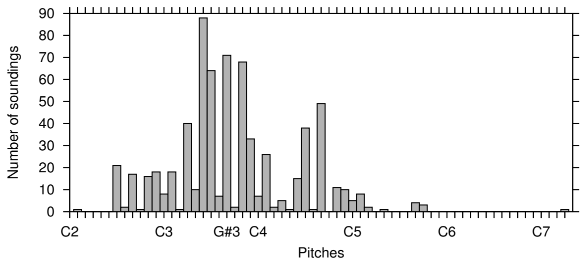

Figure 3. *Pitch distribution of the transcription result of J. S. Bach's "BWV 1007 Prelude" performed by Carrai. G#3 is the median. Only notes with MIDI velocities of equal to or greater than V_t are included.*

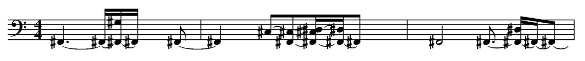

Figure 4. *The filtered result of J. S. Bach's "BWV 1007 Prelude" performance by Carrai. Note that although as shown in Figure 2 the first note is an F#3, it does not appear here because it is softer than V_t.*

the most effective melody extraction technique for retrieval purposes. When there is a note or a chord starting to sound, the note having the highest pitch becomes "the melody note." If there is a note m of length l_m sounding at t_m and another note n sounding at t_n so that $l_m + t_m \geq t_n$, l_m will become $l'_m \leftarrow l_m - (t_n - t_m)$, that is, note overlaps are removed. For our experiments, we modify the ALL-MONO algorithm so that instead of taking the highest pitch, we take the lowest pitch. In Algorithm 1 we show the bass-part extraction algorithm we use (note duration information however is not extracted as this is not used in post-extraction stages). This process is applied to tunes in the collection and also the query tune. The bass part extracted from Figure 4 is shown in Figure 5.

The rationale of modifying the ALL-MONO algorithm is that high pitches are not sufficiently reliable since most extraneous notes are high in pitch. Also, in classical music (particularly from older periods such as Baroque), the lower pitches are quite melodic, for example, J. S. Bach's Inventions (Baroque period), Mozart's piano sonatas (Classical period), and Chopin's fantasias (Romantic period), compared to three-chord blues tunes, where

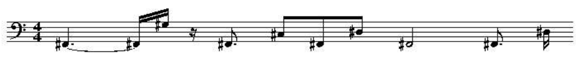

Figure 5. *The bass-line extraction result of the tune shown in Figure 4.*

the chord sequence (and thus the bass line) follows a common pattern.

Standardisation

Once a monophonic "melody" has been acquired, we can generate a string that represents it. The idea is to match the string that represents the query tune with the sequences that represent the tunes in the collection. To facilitate ranking, there must be a measure that reflects how similar two matched strings are. This is achieved using the score returned by an approximate matching function.

A standardisation that has been shown to work well is the *directed modulo-12* standardisation. In this standardisation, a note is represented as a value R which is the interval between the current note and its previous note, scaled to a maximum of 12 semitones (one octave) [17, 18]:

$$R \equiv d(1 + ((\Delta - 1) \bmod 12)) \qquad (1)$$

where Δ is the interval between a note and its previous note (absolute value) and d is $+1$ if the previous note is lower than the current note, -1 if higher, and 0 if otherwise.

Using the directed modulo-12 standardisation, the melody shown in Figure 5 is represented as $\langle +2, -2, +7, -7, +9, -9, 0, +9 \rangle$.

The clear advantage of using this standardisation is transposition-invariance. For example, the melodies C4-E4-G4 and G3-B3-D4 match perfectly, as both are represented as $\langle +4, +3 \rangle$.

Alignment

For the alignment phase of our approach, we apply dynamic programming to the pitch sequences, a technique that has been previously applied to symbolic music matching [4, 17, 19] as well as audio [2, 7]. To investigate the ability of manually constructed symbolic sequences being matched with transcription-of-audio sequences, we use *global alignment* since the query and the target tune(s)

Algorithm 1. *A note is expressed as a tuple $n = \langle p, o \rangle$ where p is the pitch and o is the onset time.* $\mathbf{N}$ *is the array of notes. The base index is* 0. *P is the sequence of the representative bass part. "π_x" is the relational operator for projecting the x attribute.*

```
1. Sort $\mathbf{N}$ by ascending onset time as the first sort key and descending pitch descending as the second sort key.
2. For $i$ in $0 \ldots |\mathbf{N}| - 2$: $(\pi_o n_i \neq \pi_o n_{i+1}) \rightarrow$ append $\pi_p n_i$ to $P$.
3. Append $\pi_p n_{|\mathbf{N}|-1}$ to $P$.
   (Steps 2 and 3 takes the lowest note at any onset time.)
4. Return $P$.
```

are complete albeit different renditions of the same tune. However, we also inspect the possibility of using *local alignment* to find out whether it is possible to issue a query in the form of a short piece of the target tune.

Suppose we have two strings, s and q. Algorithm 2 is used to calculate the global alignment between them whereas Algorithm 3 is used to calculate the local alignment between them. We use both algorithms in our experiment (see Section 4) with sequences of directed modulo-12 representation (see Eq. 1) of the tunes as the strings.

3 Retrieval Performance Evaluation

The scores generated by alignment are used for ranking the similarity of the tunes in the collection for a particular query. From here, we can measure the effectiveness of the approach we propose. The widely-known *precision* information retrieval measure [3] is used and it is defined as $P \equiv |\mathbf{Rel} \cap \mathbf{Ret}|/|\mathbf{Ret}|$ where P is precision, $\mathbf{Rel}$ is the set of relevant tunes, and $\mathbf{Ret}$ is the set of retrieved tunes. In particular, we report $\langle P_N \rangle$, that is the mean precision in the top N answers. To provide a clearer view, we also show R_N, the number of relevant answers in the top N answers.

4 Experiment

Our collection contains tunes from the Magnatune classical music collection[6] (as at 28 April 2005) stored as MP3 (MPEG Layer 3)[7] files. It contains multiple versions of some pieces, such as J. S. Bach's "Suite I for Cello Solo" (BWV 1007), performed by Phoebe Carrai, Antonio Meneses, and Gonzalo X Ruis, respectively.

We used the default setting of TS-AudioToMidi when transcribing the MP3 files. Some files could not be processed, leaving us with 1 808 MIDI files of transcriptions to work with.

For our query set, we gathered the MIDI versions of some of the covers in the Magnatune collection. The sources of the symbolic queries are the Mutopia Project,[8] Kern Scores,[9] and MuseData.[10] In total, we have 34 queries that have relevant answers in the collection. They are specified in Table 1 but we discuss the properties of some queries below.

We have two queries for "BWV 870 Prelude" (`B870_P_MD1` and `B870_P_MDP`) and also two for "BWV 870 Fugue" (`B870_F_MD1` and `B870_F_MDP`). The queries ending with `MD1` are optimised for printing and those ending with `MDP` are optimised for listening. We intend to see how effective they are in retrieving the target answers. "BWV 1042" in the collection is a polyphonic work whereas the

[6]See *http://www.magnatune.com/genres/classical/*.

[7]See *http://www.iis.fraunhofer.de/amm/techinf/layer3/*.

[8]See *http://www.mutopiaproject.org*.

[9]See *http://kern.humdrum.net*.

[10]See *http://www.musedata.org*.

Algorithm 2. *Global alignment between the strings s and q. The base index is 0. M is the match/mismatch function ($M(x,x)$ means a match and $M(x,y)|_{x \neq y}$ means a mismatch) and I is the insertion/deletion score. $M(x,x) \geq M(x,y)|_{x \neq y} \geq I$.*

```
1. Data structure preparation.
    1. Construct a matrix with $(|s|+1)$ rows and $(|q|+1)$ columns. Let us call a cell $D_{i,j}$.
    2. $D_{0,0} \leftarrow 0$.
    3. $D_{i,0} \leftarrow iI; i = \{1 \dots |s|\}$.
    4. $D_{0,j} \leftarrow jI; j = \{1 \dots |q|\}$.
2. Score calculation.
    1. For $i$ in $1 \dots |s|$:
        1. For $j$ in $1 \dots |q|$:
            1. $D_{i,j} \leftarrow \max \begin{cases} D_{i-1,j} + I \\ D_{i,j-1} + I \\ D_{i-1,j-1} + M(s_{i-1}, q_{i-1}) \end{cases}$.
3. Return $D_{|s|,|q|}$.
```

Algorithm 3. *Local alignment between the strings s and q. The base index is 0. M is the match/mismatch function ($M(x,x)$ means a match and $M(x,y)|_{x \neq y}$ means a mismatch) and I is the insertion/deletion score. $M(x,x) \geq M(x,y)|_{x \neq y} \geq I$.*

```
1. Data structure preparation.
    1. Construct a matrix with $(|s|+1)$ rows and $(|q|+1)$ columns. Let us call a cell $D_{i,j}$.
    2. $D_{0,0} \leftarrow 0$.
    3. $D_{i,0} \leftarrow 0; i = \{1 \dots |s|\}$.
    4. $D_{0,j} \leftarrow 0; j = \{1 \dots |q|\}$.
2. Score calculation.
    1. For $i$ in $1 \dots |s|$:
        1. For $j$ in $1 \dots |q|$:
            1. $D_{i,j} \leftarrow \max \begin{cases} 0 \\ D_{i-1,j} + I \\ D_{i,j-1} + I \\ D_{i-1,j-1} + M(s_{i-1}, q_{i-1}) \end{cases}$.
3. Return $\max(D_{i,j}); i \in \{1 \dots |s|\}, j \in \{1 \dots |q|\}$.
```

Table 1. *The queries and the number of relevant covers in the collection (C). (KS: Kern Scores, MD-1: MuseData [optimised for printing], MD-P: MuseData [optimised for listening], MP: the Mutopia Project.)*

Query	Title	Source	C
B1011_C	BWV 1011 Courante	KS	3
B1011_S	BWV 1011 Sarabande	KS	3
COR_O1N7_1	Corelli, Trio Sonata Op. 1 No. 7 in C Maj. (Mvt. 1)	KS	1
COR_O1N7_2	Corelli, Trio Sonata Op. 1 No. 7 in C Maj. (Mvt. 2)	KS	1
COR_O1N7_3	Corelli, Trio Sonata Op. 1 No. 7 in C Maj. (Mvt. 3)	KS	1
DUF_ACB	Dufay, Adieu Ces Bons Vins De Lannoys	KS	1
JOP_STOPTIME	Joplin, Stoptime Rag	KS	1
K545_1	K 545 Movement 1	KS	1
K545_2	K 545 Movement 2	KS	1
K238	K 238	KS	1
B870_F_MD1	BWV 870 Fugue	MD-1	1
B870_P_MD1	BWV 870 Prelude	MD-1	1
B870_F_MDP	BWV 870 Fugue	MD-P	1
B870_P_MDP	BWV 870 Prelude	MD-P	1
B1007_A	BWV 1007 Allemande	MP	3
B1007_C	BWV 1007 Courante	MP	3
B1007_G	BWV 1007 Gigue	MP	4
B1007_M	BWV 1007 Menuets	MP	3
B1007_P	BWV 1007 Prelude	MP	3
B1007_S	BWV 1007 Sarabande	MP	3
B1010_A	BWV 1010 Allemande	MP	3
B1010_B1	BWV 1010 Bouree I	MP	3
B1010_B2	BWV 1010 Bouree II	MP	3
B1010_C	BWV 1010 Courante	MP	3
B1010_G	BWV 1010 Gigue	MP	3
B1010_P	BWV 1010 Prelude	MP	3
B1010_S	BWV 1010 Sarabande	MP	3
B1042_AD	BWV 1042 Adagio	MP	1
B1042_AL	BWV 1042 Allegro	MP	1
B1042_AA	BWV 1042 Allegro Assai	MP	1
B846_F	BWV 846 Fugue	MP	1
B846_P	BWV 846 Prelude	MP	1
B860_F	BWV 860 Fugue	MP	1
B860_P	BWV 860 Prelude	MP	1

queries (B1042_AD, B1042_AL, and B1042_AA) are monophonic.

"BWV 846" and "BWV 860" in the collection are polyphonic piano pieces. The same case is also for the first two movements of Mozart's "K 545" in the collection. The queries (B846_P, B846_F, B846_P, B846_F, K545_1, and K545_2) are also polyphonic.

Joplin's "Stoptime Rag" and Scarlatti's "K 238" in the collection uses the harpsichord. Dufay's "Adieu Ces Bons Vins De Lannoys" in the collection uses a lute. Corelli's "Trio Sonata Opus 1 No. 7 in C Major" in the collection is performed by an orchestra. All are polyphonic and so are the corresponding queries (JOP_STOPTIME, DUF_ACB, COR_01N7_1, COR_01N7_2, and COR_01N7_3 respectively).

"BWV 1007 Menuets" consist of two parts. In our audio collection, there are three covers and all of them play the two parts in a single file. We obtained the query with both parts separated. They were concatenated to form B1007_M. A similar case happened with "BWV 1010 Bourees," which also consist of two parts, and there are also three covers with all of them playing the two parts as a single track. However, for this one, there were two queries, one for each part (B1010_B1 and B1010_B2).

For both the audio covers and the queries, "BWV 1007" is mostly monophonic while "BWV 1010" and "BWV 1011 Courante" are mostly polyphonic ("BWV 1011 Sarabande" is monophonic).

For filtering, we used $V_t = 48$ (see Section 2). This was applied to both the tunes in the collection and the query tunes. We tested four alignment procedures:

- Global alignment (see Algorithm 2) on whole tunes and whole queries. We call this SBDG.
- Local alignment (see Algorithm 3) on whole tunes and whole queries. We call this SBDL.
- The query tunes were truncated to a maximum of 50 symbols and matched using local alignment (see Algorithm 3). We call this SBDL50.
- The query tunes were truncated to a maximum of 100 symbols and matched using local alignment (see Algorithm 3). We call this SBDL100.

The alignment parameters we use are $M(x,x) = 10$, $M(x,y)|_{x\neq y} = -1$, and $I = -2$ (see Section 2).

5 Results and Analysis

The results for our experiments with the methods mentioned in Section 4 are shown in Figure 6. Based on the mean precision values in the figure, the most effective method is SBDG, followed by SBDL, both achieving $P_1 = 0.412$, and then SBDL100 ($P_1 = 0.324$), and finally SBDL50 ($P_1 = 0.176$). SBDG is just marginally more effective than SBDL (P_{10} is equal to 0.097 for SBDG and 0.076 for SBDL), but none is superior in terms of efficiency. In terms of R_N, using SBDG, overall 18 queries successfully retrieved an answer in the top 10 and 20, whereas with SBDL, overall 16 queries successfully retrieved an answer in the top 10 and 20. Using SBDL50, overall 9 queries retrieved an answer in the top 10, and 11 in the top 20 of returned results, whereas with SBDL100, overall 16 queries retrieved an answer in the top 10, and 18 in the top 20. The P_N and R_N values tell that shorter queries tend to sacrifice effectiveness.

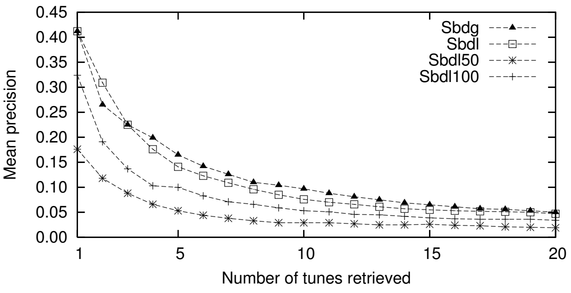

Figure 6. *The mean precision curves for* SBDG, SBDL, SBDL50, *and* SBDL100.

Symbolic data that is optimised for listening serve as better queries than those optimised for printing. Print versions have less details thus are more ambiguous. For example, in listening versions of the MIDI files, a trill (fast alteration between two nearby notes) is encoded as the notes a player would usually play. From our experience as musicians, Western classical music somewhat allows stricter interpretation. In contrast to, say, jazz performance, Western classical performers typically try to perform the repertoires as closely as how they were performed originally; for example, the use of the piano sustain pedal when performing J. S. Bach's harpsichord composition is generally frowned upon, in the spirit of emulating harpsichords. As mentioned in Section 4, in our query set, there were both print (B870_P_MD1 and B870_F_MD1) and listening (B870_P_MDP and B870_F_MDP) versions of "BWV 870." Using SBDG, the queries B870_P_MD1 and B870_F_MD1 retrieve the target tunes at ranks 313 and 63 respectively, whereas the B870_P_MDP and B870_F_MDP at ranks 9 and 30. The same situation also occurs for SBDL. B870_P_MD1 and B870_F_MD1 retrieve the target tunes at ranks 404 and 52, whereas B870_P_MDP and B870_F_MDP at ranks 24 and 36.

The retrieval performance for "BWV 1007" was among the best. Using SBDG, all of the queries produced the first correct answer in the first place, except B1007_M, which produced the first correct answers at rank 3. All of the queries retrieved at least two correct answers in the top 10. They also retrieved all correct answers in the top 20 except B1007_M, which only retrieves two of them (at ranks 3 and 4, the other one at rank 34). Using SBDL, all of the queries produced the first correct answer at the first place with no exception. Only B1007_S failed to retrieve all correct answers in the top 20 (but close enough; it retrieved the last correct answer at rank 22).

The polyphonic piano pieces "BWV 846," "BWV 860," and "K 545" were also retrieved in the first place using SBDG. The retrieval performance SBDL is the same except for B846_F, which produces the correct answer at rank 2. These good results may be partly due to fairly clean transcriptions.

The retrieval performance for "BWV 1010" is only good for B1010_A and B1010_P. For B1010_A, the correct answers were retrieved at ranks 1, 7, and 93 (using SBDG) and 2, 16, and 27 (using SBDL). For B1010_P, all the three covers were retrieved in the top 3. For B1010_C and B1010_S, both failed to retrieve answers in the top 20 using any of the methods. B1010_B2 with SBDL however retrieved a correct answer in the first place.

Retrieval performance was poor for DUF_ACB, JOP_STOPTIME, B1042_AD, B1042_AL, and B1042_AA using all methods. All of them could not retrieve a correct answer in the top 90. B1042_AD could not even retrieve a correct answer in the top 1 000. This is actually not surprising since the query contains the high pitches and lacks the "bass part."

B1011_C performed poorly but B1011_S retrieved the correct answer in the first place using all methods.

Referring to "BWV 1007" and "BWV 1011 Sarabande", it should be noted that even though an audio performance is monophonic, its transcription result is polyphonic (refer again to Figures 1 and 2).

All the mostly or fully polyphonic audio, with the exception of the piano pieces, were hard to retrieve. Our simple filtering heuristic still cannot filter out the noise from the transcription results of these pieces well. However, where actual notes and extraneous notes are separated well by the median of the overall pitch distribution, this makes low pitch audio tend to be retrieved more easily.

The results for SBDL and SBDG show that it is possible to align a whole audio music file with its symbolic equivalent representation by using dynamic programmign techniques that had previously been shown effective for aligning symbolic sequences. However, using the same techniques with short queries does not yield the same level of effectiveness. This still requires further experimentation.

6 Conclusions and Future Work

This paper describes our exploratory work in symbolic-audio music cross-matching using dynamic programming and western classical music. The results of our experiment show that:

- Automatic audio transcription results need to be filtered in order to be useful for matching.
- It is possible to match audio with symbolic music.
- Long queries are more effective than short ones.

Extraneous notes included in transcription results seem to degrade retrieval effectiveness. As they are typically highly pitched, removing those notes using the median of the pitch distribution only works well when they are separated well enough from the correctly transcribed notes (by the median). As such, a different technique to work with musical pieces where the correctly transcribed notes and the extraneous ones are not well separated needs to be devised. Using a heuristic that recognises melody lines accurately may yield better results. This is supported by the evidence in our experiment that fairly clean transcribed audio can be retrieved by its equivalent symbolic versions.

We plan to explore this technique on music of other genres to see the scope of its applicability.

7 Acknowledgements

We thank Justin Zobel for obtaining TS-AudioToMidi and proof-reading the draft version of this paper and Steven Garcia for his input on automating the transcription process.

References

[1] J.-J. Aucouturier and F. Pachet. Music similarity measures: What's the use? In M. Fingerhut, editor, *Proceedings of the Third International Conference on Music Information Retrieval*, pages 157–163, Paris, France, Oct. 2002. IRCAM-Centre Pompidou.

[2] J.-J. Aucouturier and M. Sandler. Using long-term structure to retrieve music: Representation and matching. In Downie and Bainbridge [5], pages 1–2.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* ACM Press, New York, USA, 1999.

[4] R. B. Dannenberg, W. P. Birmingham, G. Tzanetakis, C. Meek, N. Hu, and B. Pardo. The Musart testbed for query-by-humming evaluation. In Hoos and Bainbridge [8], pages 41–47.

[5] J. S. Downie and D. Bainbridge, editors. *Proceedings of the Second International Symposium on Music Information Retrieval*, Bloomington, USA, Oct. 2001.

[6] J. Eggink and G. J. Brown. Extracting melody lines from complex audio. In C. L. Buyoli and R. Loureiro, editors, *Proceedings of the Fifth International Conference on Music Information Retrieval*, pages 84–91, Barcelona, Spain, Oct. 2004. Universitat Pompeu Fabra.

[7] J. Foote. Arthur: Retrieving orchestral music by long-term structure. In D. Byrd, J. S. Downie, T. Crawford, W. B. Croft, and C. Nevill-Manning, editors, *Proceedings of the First International Symposium on Music Information Retrieval*, Plymouth, USA, Oct. 2000.

[8] H. H. Hoos and D. Bainbridge, editors. *Proceedings of the Fourth International Conference on Music Information Retrieval*, Baltimore, USA, Oct. 2003. Johns Hopkins University.

[9] N. Hu and R. B. Dannenberg. A comparison of melodic database retrieval techniques using sung queries. In G. Marchionini and W. Hersh, editors, *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 301–307, Portland, USA, July 2002.

[10] D. Mazzoni and R. B. Dannenberg. Melody matching directly from audio. In Downie and Bainbridge [5], pages 17–18.

[11] R. P. Paiva, T. Mendes, and A. Cardoso. On the detection of melody notes in polyphonic audio. In Reiss and Wiggins [14], pages 175–182.

[12] E. Pampalk, A. Flexer, and G. Widmer. Improvements of audio-based music similarity and genre classification. In Reiss and Wiggins [14], pages 628–633.

[13] G. E. Poliner and D. P. W. Ellis. A classification approach to melody transcription. In Reiss and Wiggins [14], pages 161–166.

[14] J. D. Reiss and G. A. Wiggins, editors. *Proceedings of the Sixth International Conference on Music Information Retrieval*, London, UK, Sept. 2005. Queen Mary, University of London.

[15] J. Shifrin and W. P. Birmingham. Effectiveness of HMM-based retrieval on large databases. In Hoos and Bainbridge [8], pages 33–39.

[16] P. Somerville and A. L. Uitdenbogerd. Classification of music based on musical instrument timbre. In *Proceedings of the Fourth Australasian Conference on Knowledge Discovery and Data Mining*, pages 173–188, Sydney, Australia, Dec. 2005.

[17] I. S. H. Suyoto and A. L. Uitdenbogerd. Effectiveness of note duration information for music retrieval. In L. Zhou, B. C. Ooi, and X. Meng, editors, *Proceedings of the Tenth International*

Conference on Database Systems for Advanced Applications, pages 265–275, Beijing, China, Apr. 2005. Springer-Verlag.

[18] A. L. Uitdenbogerd. *Music Information Retrieval Technology.* PhD thesis, School of Computer Science and Information Technology, RMIT, Melbourne, Australia, 2002.

[19] A. L. Uitdenbogerd and J. Zobel. Melodic matching techniques for large music databases. In D. Bulterman, K. Jeffay, and H. J. Zhang, editors, *Proceedings of the 1999 ACM Multimedia Conference*, pages 57–66, Orlando, USA, Nov. 1999.

[20] A. Wang. An industrial-strength audio search algorithm. In Hoos and Bainbridge [8], pages 7–13. Invited talk.