MUSICAL STRUCTURE RETRIEVAL BY ALIGNING SELF-SIMILARITY MATRICES

Benjamin Martin, Matthias Robine and Pierre Hanna
LaBRI - University of Bordeaux
351, cours de la Libération
33405 TALENCE Cedex - FRANCE
firstname.name@labri.fr

ABSTRACT

We propose a new retrieval system based on musical structure using symbolic structural queries. The aim is to compare musical form in audio files without extracting explicitly the underlying audio structure. From a given or arbitrary segmentation, an audio file is segmented. Irrespective of the audio feature choice, we then compute a self-similarity matrix whose coefficients correspond to the estimation of the similarity between entire parts, obtained by local alignment. Finally, we compute a binary matrix from the symbolic structural query and compare it to the audio segmented matrix, which provides a structural similarity score. We perform experiments using large databases of audio files, and prove robustness to possible imprecisions in the structural query.

1. INTRODUCTION

Content-based search on very large audio files databases is an important issue in music information retrieval. New browsing tools propose to compare audio songs according to music properties such as style, rhythm, melody, timbre, etc. Among all of these properties, taking into account information about structure may be very useful for discriminating songs, since it may be closely linked to music style or music composer. In this paper, we propose to focus on these structural properties.

Musical structure has been of major concern over the last years. This field aims to retrieve and compare human-recognizable musical structure within an audio piece. To this end, Foote proposed in 1999 [1] a self-similarity matrix-shaped representation whose coefficients carry structural information over the musical piece. This matrix is obtained analyzing repeated sections within the piece; it describes both a global structure and different local structures.

Existing works about music structure generally focus only on structural analysis in audio files. Foote *et al.* proposed a music summarization method by summing scores in its self-similarity matrix representation [2]. Dannenberg tested several transcription methods by adapting them to the nature of the given audio file, in order to explicitly retrieve the underlying structure [3]. Müller *et al.* proposed a method to extract relevant paths in a self-similarity matrix, deducing the precise structure of the music piece [9]. Bartsch developed an automatic thumbnailing system that retrieve relevant parts from audio [7], and Goto focused later in chorus extraction over songs taking into account possible modulations or variable durations of their occurrences in the audio musical piece [6]. Peeters used [5] a representation in terms of "states" of music and proposed an algorithm that computes a structural summarization in several passes over the audio file. Bartsch *et al.* worked [4] on a summarization centered on choruses using chroma features. Paulus *et al.* used [8] several audio features to build a probability space in order to analyze the best probable underlying structure.

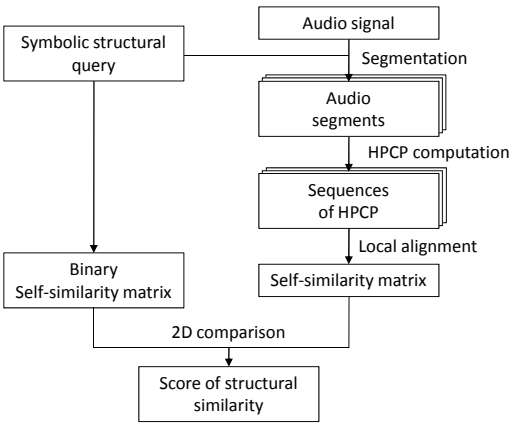

Figure 1. Overview of the query-by-structure method proposed.

Once the structure is correctly analyzed, a retrieval system may be directly developed by comparing the sequences of structure. The problem is the correctness of the structure estimated. Although several systems have been proposed and evaluated, analysis errors significantly limit the accuracy of a retrieval system based on such approaches. Lately, Izumitani and Kashino proposed a method that estimates the structural similarity between two excerpts by directly comparing the self-similarity matrices computed [12]. This method is applied to cover song detection.

The method we propose trails after the same principle: comparing musical form without extracting explicitly the underlying audio structure. In this paper, we bring a new retrieval system based on musical structure using symbolic queries. In Section 2, we describe the system proposed. In Section 3, we present different experiments on real pop music databases. Finally, we conclude and open perspectives in Section 4.

2. METHOD

The proposed method searches the database for the musical piece that best matches a given symbolic query. See Figure 1 for the method global schematic view.

2.1 Audio Self-Similarity matrix computation

First, the system splits an audio file into n audio segments according to a given segmentation. The applied segmentation used in our retrieval system will be described later.

2.1.1 Features

In the proposed method, the comparison is based on musical structures, not directly on audio features. Therefore, it is fundamental to keep in mind that the chosen audio features can be changed, regardless of the further steps of the method.

As a feature set, we use *Harmonic Pitch Class Profiles* (HPCP) [11]. HPCPs, which provide tonal information, are robust to noise, timbre, dynamics, tuning or loudness variations, and ensure then an accurate tonal description. The different tonality vectors are extracted the same way as in [11] or [10], frame by frame.

This analysis transforms an input audio file into a sequence $H = (\overrightarrow{h_i})_{1 \leq i \leq n}$ of N B-dimensional vectors $\overrightarrow{h_i}$, where N denotes the number of frames in the musical piece, and B denotes the chosen chroma bin resolution (generally 12, 24 or 36). Our system settles for a 12 bins resolution.

The method proposed compares the signal to itself in order to retrieve structural information (see Section 2.1.2). That's why an adapted measure that enables the comparison between two HPCP vectors is needed. We choose the binary local alignment technique described in [10] to this purpose. On top of being adapted to HPCPs, this measure computes the optimal transposition index between two chromas to provide a similarity score, which allows our detection to be robust to key changes within the same audio musical piece. This comparison measure is able to compute from two features sequences H_1 and H_2 a similarity score by local alignment.

2.1.2 Segmented self-similarity matrix of an audio file

As explained before, we use self-similarity matrices in order to represent the repeated sections. In our model, self-similarity matrices contain elements, whose range is $[0, 1]$, that stand for the likeliness between two parts of a structure. Horizontal and vertical axis of the matrix represent time, that runs from left to right as well as from top to bottom.

In classic uses of self-similarity matrices, each i, j coefficient is computed by comparing the feature corresponding to the time i of the musical piece with the one corresponding to the time j of the same musical piece. However, in our model, contrary to the general self-similarity computing process, time does not run uniformly on both axis. Indeed, each element i, j of the matrix corresponds to the similarity measure between two entire parts P_i and P_j of the musical piece. Thus, each coefficient corresponds to the evaluation of the similarity between two sets of feature vectors, not directly between two vectors.

Based on the n audio file segments computed from the audio signal, n sequences $H_1, H_2, \ldots, H_n$ of HPCP vectors are computed, each one corresponding to an audio segment. Therefore, comparing two HPCP sequences means comparing two segmented parts of the audio signal. Thus, for each couple of HPCP sequences H_1 and H_2, we compute a similarity score using the binary local alignment by dynamic programming technique inspired from Gomez's work and described in 2.1.1. This provides a self-similarity matrix R, whose coefficients stand for the comparison of two parts within the musical piece:

$$R = (\text{alignment}(H_i, H_j))_{1 \leq i \leq n, 1 \leq j \leq n} \qquad (1)$$

where alignment() denotes the binary local alignment by dynamic programming technique used to compare two HPCP sets. An example of reference segmented self-similarity matrix is shown on Figure 2 (right).

2.2 Query matrix computation

In a first approach, the query used for comparisons is a binary self-simlarity matrix computed from a symbolic structural query, that can be arbitrarily defined or taken from a ground truth.

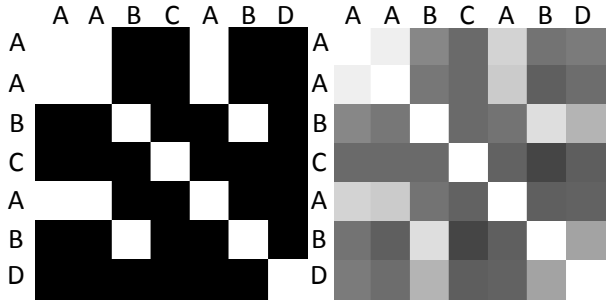

Figure 2. Left: Symbolic query self-similarity matrix of *The Beatles - All My Loving* musical piece. Right: Segmented reference self-similarity matrix of the same musical piece. Letters show the different recognizable patterns; white pixels stand for repeated sections, and black pixels stand for distinct sections. The left reference matrix is segmented according to the ground truth for this musical piece (exact structure).

2.2.1 Symbolic structural query

Our model uses symbolic structural queries, which are a symbolic representation of the underlying structure of a musical piece. A symbolic query can be seen as a sequence of symbols that represent a particular musical form. Each part within a symbolic structural query has its own duration. It can represent a simple note as well as an entire excerpt.

A symbolic structural query can be seen as a sequence of symbols, for instance 'aabca', combined or not to symbols representing duration information. Two identical symbols within the sequence indicate a similarity between the two corresponding parts, and two different symbols indicate a dissimilarity.

2.2.2 Self-similarity matrix of a symbolic query

Symbolic queries are comparable to pattern sequences that impose two kinds of constraints : similarities, *i.e.* remarkable repetitions, and dissimilarities.

The symbolic query self-similarity matrix is created by analyzing the provided patterns sequence. Assuming that $(s_k)_{1 \leq k \leq n}$ represents a symbols sequence of length n (*e.g.* $s =$ '$ababc$'), the query self-similarity matrix Q is defined as follows:

$$\forall (i,j) \in \{1 \ldots n\}^2,\ Q_{i,j} = \begin{cases} 1 & if \ \ s_i = s_j \\ 0 & if \ \ s_i \neq s_j \end{cases} \qquad (2)$$

The resulting matrix is binary, and stands for 2 types of constraints: similarity and dissimilarity. An example of a self-similarity matrix created from a symbolic query can be viewed in Figure 2 (left).

2.3 Matrices comparison

In order to assign a similarity score between the matrices we compute, we use three different algorithms that have different properties. These three algorithms provide a normalized similarity score according to a binary query matrix and a reference matrix. From now on, the two matrices compared are denoted as Q and R.

The first approach consists in computing the similarity between matrices using a pixel-to-pixel algorithm, based on an euclidean distance algorithm. However, we consider more sophisticated algorithms that show different characteristics.

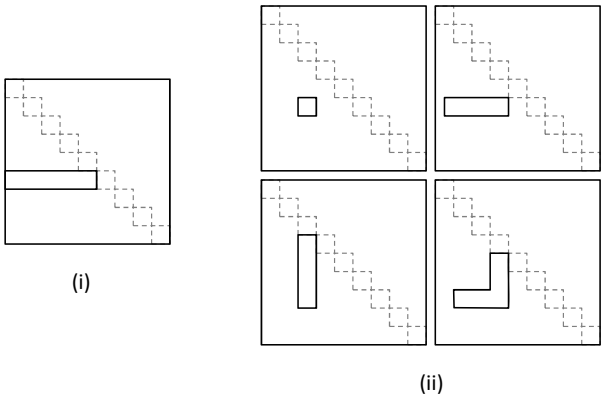

Figure 3. Possible compared patterns taken into account in local alignment with (i) Izumitani's algorithm and (ii) Lecroq's algorithm.

An approach to the comparison problem can be yielded by local alignment algorithms. In our context, this principle is extended to matrices comparison; possible operations in order to transform the first matrix into the second work on symbols that can be seen as clusters of the compared matrices.

In some cases, it can be very relevant to consider pixel-wise deletions or insertions in the matrices comparison. Moreover, if the query binary matrix is smaller than the compared reference matrix (structure excerpts search), local alignment techniques are able to cope with the dimension difference and to provide a local similarity score.

2.3.1 Izumitani and Kashino's algorithm

The first local alignment algorithm tested is the one presented in [12]. This algorithm takes as an input two self-similarity matrices. Since these matrices are symmetrical and have constant maximum diagonal, this algorithm only works on the lower triangles. It is based on a dynamic programming method that searches the diagonal direction of the reference self-similarity matrix. Indeed, the Izumitani and Kashino's algorithm compares each entire line of the reference matrix with each entire line of the query matrix. Thus, for each comparison, it allows the 3 different operations (insertion/deletion/substitution) on a unique pattern: entire lines of the lower triangle of the compared matrices. This pattern can be seen on Figure 3 (left).

Furthermore, the dynamic programming matching method is based on "matched element indices sets" recursively computed, which means that at each step n, matching elements between two compared lines are deduced from the $n-1$ step (see [12], 2.3, p.612). In other words, if an element does not match the compared line at a step, the whole following column elements will not be taken into account in any further comparison. This represents a limitation of this algorithm.

2.3.2 Lecroq et al.'s algorithm

In order to improve the alignment comparison, we chose to consider a different method. Indeed, reducing considered patterns for the comparisons to lines only seemed to be rather limited, which led us to evaluate a new method.

The second local alignment algorithm studied is adapted from [13]. It was developed and used in order to compare symbolic dialog annotations, and is particularly specialized in aligning 2-dimensional patterns. Lecroq *et al.*'s algorithm browses the matrices element by element, and allows the 3 typical operations on different patterns: a single element (pixel), a part of a line or an entire line, a part of a column or an entire column, and a part of a line and a part of a column simultaneously. These different patterns are represented on Figure 3 (right).

Our adaptation consisted in not comparing text entries but patterns included in self-similarity matrices, taking into account their properties and adapting the comparison to a non-binary similarity measure.

2.4 Retrieval system specificities

2.4.1 Query-based segmentation

Symbolic structural queries can be combined or not to informations about the duration of each pattern.
If the query indicates absolute time informations (in frames or seconds), these can be used to split the musical piece in segments.
If the query indicates relative time informations, the global duration of the piece can be used to split the musical piece.
If no time information is provided in the symbolic query, segmentation is arbitrary: for instance, it can be uniformly processed, each part having the same duration than the others.

2.4.2 Pre-processing

Before evaluating a similarity score, the reference self-similarity matrix must be pre-processed. Indeed, since the reference matrix contains the similarity scores between the different parts of the musical piece, the distribution of the values of its coefficients is likely to vary according to the considered musical piece. For some audio files, the tonal distinction between two parts that are supposed to be different, e.g. a chorus and a verse, will be very clear, whereas it will turn out to be vague in some other cases.

Let μ and σ be respectively the average and the standard deviation values of the coefficients' distribution in the reference matrix R. The normalized reference self-similarity matrix $\hat{R}$ is computed as follows:

$$\forall (i,j) \in \{1 \dots n\}^2, \hat{R_{i,j}} = \frac{(R_{i,j} - \mu)}{\sigma} \cdot \hat{\sigma} + \mu \qquad (3)$$

where $\hat{\sigma}$ is a constant corresponding to the new standard deviation to apply. It must be adapted to the pixel-to-pixel comparison constants (see 2.4.3). In our model, we used $\hat{\sigma} = 0,31$.

2.4.3 Adapted euclidean distance

In a first approach, we compare matrices by computing an euclidean score, based on pixel-to-pixel comparisons. As explained before, the binary query matrix contains similarity and dissimilarity constraints. However, our method does not give these two constraints the same importance.

Actually, we can reasonably hypothesize that two parts that are supposed to be similar (*i.e.* that are represented by the same symbol in the query) present two close tonal descriptions, whereas two parts that are supposed to be dissimilar can be either close, or different in their tonal descriptions: there is no gradation on the dissimilarity notion. Therefore, it is necessary to make an distinction between the strong similarity constraints and the weak dissimilarity constraints that imposes the symbolic query.

Our adapted euclidean distance computes then two different scores, which takes into account this distinction: A similarity score $s^{=} \in [0,1]$, that is established only with similarity constraints imposed by the query,
A dissimilarity score $s^{\neq} \in [0,1]$, that is established only with the dissimilarity constraints of the query.
Let Q and R denote the query and reference compared matrices, respectively. We introduce the set

$$R^{=} = \{(i,j) | Q_{i,j} = 1, i < j\}$$

that denotes the reference matrix indices that correspond to a similarity (white pixel, value 1) in the query. In the same way, we introduce the set

$$R^{\neq} = \{(i,j)|Q_{i,j} = 0, i < j\}$$

that denotes the reference matrix indices that correspond to a dissimilarity (black pixel, value 0) in the query. $s^{=}$ is computed comparing each similarity element of the query with the corresponding element in the reference the following way:

$$s^{=} = \frac{1}{|R^{=}|} \sum_{(i,j)\in R^{=}} ||1 - R_{i,j}|| \quad (4)$$

where $||.||$ denotes the classical euclidean norm. $s_{\neq}$ is computed comparing each dissimilarity element of the query with the corresponding element in the reference the following way:

$$s^{\neq} = \frac{1}{|R^{\neq}|} \sum_{(i,j)\in R^{\neq}} f(R_{i,j}) \quad (5)$$

where f denotes an exponential shaped function that was empirically determined :

$$f(x) = \frac{1}{e^8 - 1} \cdot (e^{8 \cdot x} + 1)$$

3. EXPERIMENTS

In order to evaluate our method in the most systematic way, we established a serie of experiments that underline its different characteristics.

3.1 Databases establishment

We based our research on a structural ground truth corpus created by M. Levy, K. Noland and G. Peeters [1]. It includes 60 accurate XML annotations for western pop songs, and numbers for each musical piece every detected section with its duration. It was found to be one of the most used corpora in this field, in many prior studies, such as [14] or [15].

We used two different audio files databases in order to validate our model.
- The ground-truth corresponding audio files database $\mathcal{D}_g$, that includes the 60 musical pieces annotated in the corpus;
- A noise database $\mathcal{D}_m$, that contains 200 audio western pop music audio files from different authors.
Obviously, we respected the following inclusion:
$\mathcal{D}_g \subset \mathcal{D}_m$. Audio files were taken from commercial CD versions.

We analyzed signals using HPCP features (see 2.1.1) with an overlap of 50% and a window size of 744 milliseconds.

3.2 Exact structural queries

The first experiment consists in searching musical pieces through a large database with the prior knowledge of their exact structure and time segmentation. Thus, we generated queries according to the symbolic representation and segmentation provided by our ground truth corpus. To compute the reference matrices, we segmented the $\mathcal{D}_m$ files according to each available ground truth data given in $\mathcal{D}_g$ (60 files). Segmentation was carried out with relative time information provided by the queries.

We then computed similarity scores between each query and the 200 well-segmented references, and checked whether the best matching was made on the file corresponding to the query. To do so, since the applied segmentation and symbolic query were supposed to be exact, we used the adapted euclidian distance described in Section 2.4.3. Because of the high accuracy of the used symbolic queries and segmentations, this simple algorithm was as efficient as local alignment techniques for this experiment.

[1] http://www.elec.qmul.ac.uk/digitalmusic/downloads/index.html#segment

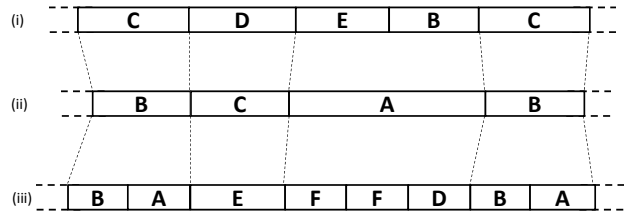

Figure 4. Structural excerpts concordance between the three best results matching with *All My Loving* excerpt. (i) *Stop the rock*, (ii) *All My Loving* - the queried structural excerpt, and (iii) *A Day in the Life*. Rectangles widths are commensurate with pattern durations; Dashed lines show structural concordances.

For all of the 60 cases tested, the experiment was conclusive: the corresponding audio file was best matched. Therefore, the system is able to retrieve exactly a musical piece providing its exact structure and segmentation.

3.3 Structure excerpts queries

The second experiment consists in searching musical pieces through a database with the prior knowledge of a part of their exact structure and time segmentation. In other words, knowing an excerpt of a structure and the segmentation of a musical piece, we now aim at retreiving the entire original musical piece as well as any piece that contains the same given structural excerpt.

As a symbolic query, we chose a structural excerpt from the *All My Loving* by *The Beatles* musical piece:
'VCBV' (Verse - Chorus - Bridge - Verse).
We assume that the time segmentation of this structure is known, *i.e.* we know how long each part lasts.

Since the symbolic query matrix and the reference matrix do not have the same size, the pixel-to-pixel distance is irrelevant here. Thus, we tested the alignment of the subrequest matrix on every musical piece of $\mathcal{D}_g$ segmented according to the *The Beatles* piece with Izumitani et al.'s and Lecroq et al.'s algorithms.

With each of both algorithms, the best matching was obtained on the original Beatles piece. However, the second best matched result differs from one algorithm to the other. Figure 4 shows the matched structure excerpt on the best matched pieces: the original *The Beatles* piece (ii), that was best matched on both of the algorithms, the second best matched piece with Izumitani's algorithm (i), and the second best matched piece with Lecroq's algorithm (iii). The indicated patterns correspond to ground truth data relative to each audio musical piece. Here are the full symbolic structures of these pieces :
All My Loving: 'A A B C B A B D'
A Day in the Life: 'A B B B C D E B C F G'
Stop the Rock: 'A A A A A B C D B A E F F D B A'

By segmenting musical pieces according to the *All My Loving* ground truth, the structures of pieces (i) and (iii) were re-segmented, exhibiting a new structure that highly matched the queried structure excerpt (ii). Dashed lines in Figure 4 show the high pattern matching: pattern similarities and dissimilarities are nearly identical between each of the three best matched pieces.

3.4 Time robustness

In the experiments described above, we processed exact segmentations taken from the ground truth. However, our query-by-structure method aims at getting rid of time constraints, and at being able to retrieve correctly providing exclusively a symbolic query as an entry.

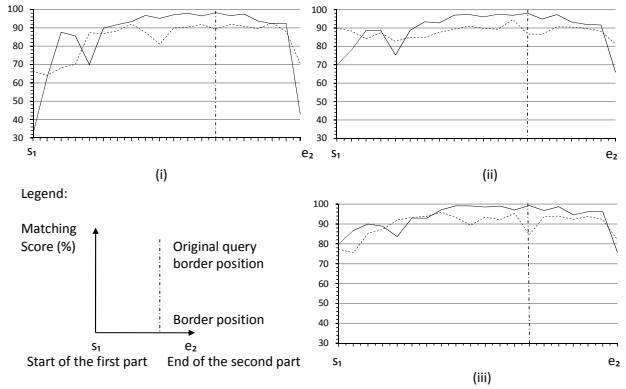

Figure 5. Altering a single time border between two parts of a structure: consequences on similarity scores. (i): Izumitani and Kashino alignment algorithm, (ii): Lecroq alignment algorithm, (iii): Adapted pixel-to-pixel distance. Plain line stand for the *All My Loving, The Beatles* musical piece, while dashed line stands for the second best matching score. Horizontal axis show the border position, from the start of the first part to the end of the second, in seconds. Its length is about 15 seconds.

Therefore, it is necessary to test our algorithms on time constraints, *i.e.* to change more or less the prior exact segmentation in order to evaluate their robustness towards this criterion.

As explained before, a segmentation combined to a symbolic query provides a series of symbolic parts indexed in time. From now on, we will denote as a border of two consecutive parts the exact point in time when ends the first part and starts the second one. The following time robustness experiments consist in changing the position of one or several borders in order to observe the impact on the matching scores.

3.4.1 Changing one segmentation border

The first time robustness test consists in changing one border over a given segmentation. From the ground truth corpus, we chose a musical piece (*All My Loving*), and modified its exact segmentation.

Here is the symbolic structure of this musical piece :
Verse - Verse - Chorus - Bridge - Verse - Chorus - Outro
We chose to work on the first Chorus / Bridge border, for it generally demarcates two distinct parts in their tonal description.

Let s_1, e_1, s_2 and e_2 denote respectively the beginning and the end of the first chorus, and the beginning and the end of the bridge. In order to estimate time robustness on our method, we generated a series of 20 different segmentations changing the border position from s_1 to e_2, keeping the constraint that $e_1 = s_2$. Thus, the only difference between two generated segmentations is the position of this border.

The experiment results are shown on Figure 5. The plain line identifies the scores for the considered musical piece, whereas the dashed line indicates the second best matching score over the database $\mathcal{D}_g$. In the x-axis, the tested position of the border varies from s_1 to e_2, the vertical dashed and dotted line indicating the original position of the border in the ground truth.

For the 3 comparison methods, we can see that the scores obtained for the musical piece (plain lines) vary slightly in an interval that corresponds to ± 6 seconds around the original position of the border. Above this value, the border seems to alter the score in such a way that the tested musical piece does not best match the symbolic query matrix.

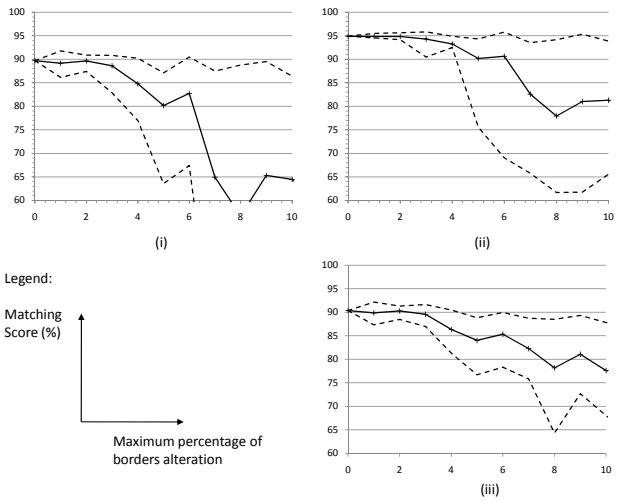

Figure 6. Altering every time borders between parts in a structure: consequences on similarity scores. (i) Izumitani and Kashino alignment algorithm, (ii) Lecroq et al. alignment algorithm, (iii) Adapted pixel-to-pixel distance. Plain line stand for the average over the 10 draws, and dashed lines stand for the greatest and lowest obtained values. The matching scores correspond to the output computed by the different algorithms.

3.4.2 Changing every segmentation borders

After changing a single border in a segmentation, we estimated time robustness on our algorithms by changing every borders in the same segmentation. Let d be the total considered musical piece duration. We introduce p_{max}, expressed as a percentage of the total musical piece duration, that corresponds to the maximum variation of each border in the segmentation (in percent). In other words, introducing $t_{max} = p_{max} \cdot d$, each border in the segmentation may be changed to a new time that does not exceed the old one $\pm t_{max}$. We chose 10 different values for p_{max} from 1% to 10% of the total piece duration, and generated a random segmentation whose borders were changed according to the above principle. This operation was repeated 10 times for each value of p_{max}.

Results can be seen in Figure 6. The figures show the average scores (plain lines) as well as the minimum and maximum scores (dashed lines) obtained over the 10 draws. They show the evolution of similarity scores between the symbolic query and the randomly altered segmentations according to the maximum alteration factor. Considering the second best matching scores obtained on the same symbolic query, the three algorithms seem to be robust to a maximum variation of the borders of 6% of the total piece duration. Above this value, the best matching is realized on a different piece.

3.5 Query-by-Structure

We now consider timeless queries that only impose a symbolic structure. From now on, no information about time segmentation is given: the principle is to search for a musical piece in the database providing exclusively its symbolic structure.

In order to realize this experiment, we focused on a few pieces that make part of the ground truth. This way, we could get the annotated structure of each musical piece and use it as the input symbolic query of our method. However, borders timings were not retrieved from the annotation files, which yielded queries without timing indications. Lacking any indication about time in the segmentation, our system assumes that the different symbols are

#	Musical Piece	Rank		
		Eucl	Izum	Lecroq
1	AllMyLoving	7	6	**4**
2	DevilinHerHeart	2	**1**	2
3	Drive	**1**	**1**	3
4	ItWon'tBeLong	**1**	**1**	**1**
5	Lonestar	84	111	**26**
6	Misery	**1**	**1**	**1**
7	NotaSecondTime	**1**	**1**	**1**
8	TakeOnMe	29	**13**	55
9	Thubthumpning	**3**	15	**3**
10	Wannabe	13	12	**8**
11	WhenI'mSixtyFour	100	90	**54**
12	WithaLittleHelp..	**1**	**1**	**1**
13	Words	**2**	9	10
14	YouReallyGot..	**1**	**1**	**1**
	Average Rank	17.57	18.79	12.14
	Median Rank	2	3.5	3
	MRR	0.544	0.537	0.480

Table 1. Query-by-Structure results with 3 different algorithms: euclidean (Eucl), Izumitani (Izum) and Lecroq.

uniformly distributed over the excerpt. In other words, it assigns the same duration to each symbol in the query.

Results can be viewed in Table 1. We tested 14 different symbolic structural queries with no time informations compared to each file of $\mathcal{D}_m$. The result table shows the retrieval ranks obtained for every musical pieces and for each of the 3 algorithms.

As shown by the two previous experiments, the algorithms used are able to deal with an inexact segmentation over the audio files. However, we saw limitations on the maximum time variation applied on borders. After analysing the ground truth relative to the different tested musical pieces, we could split the test set in two classes:
- Regular structured pieces, whose recognizable patterns have close durations (less than 10% of the duration of the piece). This class contains pieces No. 1, 2, 3, 4, 6, 7, 9, 12, 13, 14 in the table;
- Irregular structured pieces, whose recognizable patterns durations may vary significantly (possibly more than 10% of the duration of the piece). This class contains pieces No. 5, 8, 10, 11.

As we can see, irregular structured excerpts ranks are rather high with the three algorithms, which shows the limitations of our system. However, regular structured pieces are much more precisely retrieved.

At comparing the different algorithms used, we can see that the three algorihms seem to be efficient, euclidian adapted distance providing the best Mean Reciprocal Rank (MRR).

4. CONCLUSION AND FUTURE WORK

We have proposed a music retrieval system based on structural similarity. Considering a symbolic representation of the underlying structure of a musical piece, an audio file can be segmented and compared using self-similarity representation on HPCP features, providing a similarity score that indicates the structural likeliness. We used three different algorithms to compare matrices. We proved that not only the algorithm works exactly on accurate symbolic queries, but it presents also a significant robustness to slight time variations, which even allow searching for structures ignoring timing informations.

An interesting idea as a perspective is to work on a query-by-example oriented system. This time, the query will not be a symbolic structural information but an audio file, arbitrarily segmented. Then, the retrieval will be focused on finding the closest structures to the underlying structure of the input audio file. In our first tests, we managed to get from an input audio file a very structurally similar musical piece.

Finally, our system is based on a tonal representation of the signal. Nevertheless, since the retrieval operation works on structural data, this parameter could be changed to any other one. We should consequently test our system on other features, such as rhythm representations or timbre analysis.

This work is part of the SIMBALS project (JC07-188930), funded by the French National Research Agency.

5. REFERENCES

[1] J. Foote: "Visualizing Audio Segmentation using Self-Similarity," *Proc. of ACM Multimedia*, 77-80, 1999.

[2] J. Foote and M. Cooper: "Automatic Music Summarization via Similarity Analysis," *ISMIR02*, 2002.

[3] R. B. Dannenberg and N. Hu: "Pattern Discovery Techniques for Music Audio," *ISMIR02*, 2002.

[4] M. A. Bartsch and G. H. Wakefield: "Audio Thumbnailing of Popular Music Using Chroma-Based Representations," *IEEE Transactions on multimedia, Vol. 7*, 2005.

[5] G. Peeters, A. L. Burthe and X. Rodet: "Toward Automatic Music Audio Summary Generation From Signal Analysis," *Proc. of ISMIR*, 2002.

[6] M. Goto: "A Chorus-Section Detecting Method for Musical Audio Signals," *Proc. of ICASSP*, 2003.

[7] M. A. Bartsch and G. H. Wakefiels: "To catch a chorus: using chroma-based representations for audio thumbnailing," *Proc. of WASPAA*, 2001.

[8] J. Paulus and A. Klapuri: "Music Structure Analysis Using a Probabilistic Fitness Measure and an Integrated Musicological Model" *Proc. of ISMIR*, 2008.

[9] M. Müller and F. Kurth: "Towards Structural Analysis of Audio Recordings in the Presence of Musical Variations," *EURASIP Journal on Advances in Signal Processing*, Vol. 07, 2007.

[10] J. Serra, E. Gomez, P. Herrera, and X. Serra: "Chroma Binary Similarity and Local Alignment Applied to Cover Song Identification," *IEEE Transactions on Audio, Speech and Language Processing*, 2008.

[11] E. Gomez: "Tonal description of music audio signals," *Ph.D. dissertation,*, MTG, Universitat Pompeu Fabra, Barcelona, Spain, 2006.

[12] T. Izumitani and K. Kashino: "A Robust Musical Audio Search Method Based on Diagonal Dynamic Programming Matching of Self-Similarity Matrices," *Proc. of ISMIR*, 2008.

[13] E. Chanoni, T. Lecroq and A. Pauchet: "Une nouvelle heuristique pour l'alignement de motifs 2D par programmation dynamique (in french)," *JFPDA08 (Planification, Decision and Learning French-Speaking Days)*, Metz, France, pp.83-92, 2008.

[14] E. Peiszer, T. Lidy and A. Rauber: "Automatic Audio Segmentation: Segment Boundary and Structure Detection in Popular Music," *Proc. of LSAS*, Paris, France, 2008.

[15] M. Casey and M. Slaney: "The Importance of Sequences in Musical Similarity," *Proc. of ICASSP06*, Vol. 5, Toulouse, France, 2006.