Deriving Musical Structures from Signal Analysis for Music Audio Summary Generation: "Sequence" and "State" approach

Geoffroy Peeters

Ircam,
1, pl. Igor Stravinsky,
75004 Paris, France
peeters@ircam.fr
http://www.ircam.fr

Abstract. In this paper, we investigate the derivation of musical structures directly from signal analysis with the aim of generating visual and audio summaries. From the audio signal, we first derive features - static features (MFCC, chromagram) or proposed dynamic features. Two approaches are then studied in order to derive automatically the structure of a piece of music. The sequence approach considers the audio signal as a repetition of sequences of events. Sequences are derived from the similarity matrix of the features by a proposed algorithm based on a 2D structuring filter and pattern matching. The state approach considers the audio signal as a succession of states. Since human segmentation and grouping performs better upon subsequent hearings, this natural approach is followed here using a proposed multi-pass approach combining time segmentation and unsupervised learning methods. Both sequence and state representations are used for the creation of an audio summary using various techniques.

1 Introduction

Analysis of digital media content, such as digital music, has become one of the major research fields in the past years, given the increasing amount and the increasing facilities for accessing large sets of them. Music identification (Audio-ID), music search by acoustic similarities and music structure discovery are among the new research areas of what is called Music Information Retrieval. Automatic music structure discovery from signal analysis aims specifically at satisfying demands for accessing and browsing quickly through digital music catalogs by content. Among its potential applications are:

- *Browsing/accessing music catalogs* by specific inner-keys (browsing through the chorus or theme of a music catalog),
- *Browsing through music items* (browsing a music title forward or backward by verse/ chorus/ solo),

- *Audio summary (or audio thumbnails)*: creating a short audio file representing the content of a piece of music[1],
- *Educational*: giving a quick and easily understandable (without requiring musical education) insight into temporal organization of a piece of music,
- *Musicology*: helping musicologists analyze pieces of music through the analysis of performances (rather than scores) and allowing their comparison,
- *Creativity*: re-mixing already existing pieces of music using structure[2].

Discovering the musical structure of a piece of music has been based for a long time on the analysis of symbolic representations (such as notes, chords, rhythm) [6] [17] [8]. It was therefore obvious to attempt deriving this symbolic representation directly from the signal [9] (pitch estimation, multiple pitch-estimation, beat tracking). Considering the weak results currently obtained for the task of deriving symbolic representation from the signal, other approaches have been envisioned. Approaches, like score alignment [22], attempt to link a given symbolic representation (MIDI file) to a digital signal. The signal therefore benefits from all the analysis achievable on a symbolic representation. However, since a symbolic representation is most of the time unavailable and since its derivation from the signal is still difficult to achieve, people started thinking of deriving directly the structure from lower-level signal features.

Music structure discovery from signal analysis takes its sources back from the works on signal segmentation first developed for speech applications and later used for musical applications. The question was then "what does the actual segments of the music represent ?".

1.1 Music Structure Discovery

Similarity definition: All the techniques proposed so far for music structure discovery from signal analysis are based on a search for repetitions of motives or of melodies. This kind of approach is, of course, only applicable to certain kinds of musical genres based on some kind of repetition. The search of repetitions is based on measuring the similarity between signal observations or groups of observations. This similarity is then used in order to group the observations two by two (or into clusters) or oppositely to segment the temporal flows of observations into segments. The similarity measure is based on the definition of a distance between two observations (or two groups of observations): Euclidean distance, scalar product, cosine distance, normalized correlation, symmetric Kullback-Leibler distance or the Mahalanobis distance.

[1] Note that not all audio summary techniques use the music structure in order to create an audio summary

[2] This remix process is comparable to the re-arrangement of drum-loops allowed in Steinberg© Recycle software which allows people to recompose new drum loop patterns from small slides resulting from the automatic segmentation of the drum loops into rhythmical elements.

Similarity matrix: The similarity matrix was first proposed by [11] under the name "Recurrence Plots". It was latter used in [13] for music structures discovery. If we note $s(t_1, t_2)$ the similarity between the observations at two instants t_1 and t_2, the similarity of the feature vectors over the whole piece of music is defined as a similarity matrix $\underline{\underline{S}} = |s(t_i, t_j)|\ i, j = 1, \ldots, I$. Since the distance is symmetric, the similarity matrix is also symmetric. If a specific segment of music ranging from times t_1 to t_2 is repeated later in the music from t_3 to t_4, the succession of feature vectors in $[t_1, t_2]$ is supposed to be identical (close to) the ones in $[t_3, t_4]$. This is represented visually by a lower (upper) *diagonal* in the similarity matrix.

Signal observations: The observations derived from the signal, used to compute the similarity, play an essential role in the obtained results. Various types of features have been proposed for the task of music structure discovery:

- *Mel Frequency Cepstral Coefficients (MFCCs):* the MFCCs have been first proposed in the Automatic Speech Recognition (ASR) community [15]. The MFCCs are derived from the DCT of the logarithmic spectrum previously grouped into Mel bands (logarithmically spaced frequency bands). The MFCCs can be viewed as a sinusoidal decomposition of the Mel spectrum and allows representing its global shape with only a few coefficients (usually 12-13 coefficients).
- *Chromagram:* the chromagram was proposed by [3] in order to represent the harmonic content of the spectrum. The chromagram is estimated by grouping the logarithmic spectrum bins according to their pitch-class, whatever their octave-class is. For a western musical scale, 12 coefficients are used.
- *Scalar features:* scalar features derived from the waveform or from the spectrum such as the spectral centroid, spectral rolloff, spectral flux, zero-crossing rate, RMS or Energy are often used for the task of signal segmentation [28] [14] [29] [25]
- *Mean and standard deviation of MFCCs:* in order to reduce the amount of observations needed to represent a whole piece of music, [29] propose to summarize the observation content over a short period of time by taking the statistical moment of the observations.

In part 2, we present new types of features called dynamic features

1.2 Sequence and State Approach

Whatever distance and features used for observing the signal, music structure discovery techniques can be mainly divided into two types of approaches:

- the "sequence" approach: which consider the music audio signal as a repetition of sequences of events. These methods rely mainly on the analysis of the similarity matrix.
- the "state" approach: which consider the music audio signal as a succession of states. These methods rely mainly on clustering algorithms.

1.3 Related Works

"Sequence" approach: Most of the sequence approaches start from Foote's works on the *similarity matrix*. Foote showed in [13] that a similarity matrix applied to well-chosen features allows a visual representation of the structural information of a piece of music, especially the detection of a sequence of repetitions through the lower (upper) diagonals of the matrix. The signal's features used in his study are the MFCCs.

The similarity matrix can be used for the determination of the *direct location of the key sequence* of a piece of music used then as the audio summary. In [3], a similarity matrix is computed using the chromagram parameterization. A uniform average filter is then used in order to smooth the matrix. The maximum element of the matrix, supposed to be the most representative, is then chosen as the audio summary. In [7], a summary score is defined using the "average of the MFCCs' similarity matrix rows" over a specific interval. For a specific interval size, the starting time of the interval having the highest summary score is chosen as the starting time of the audio summary.

The similarity matrix can also be used for *discovering the underlying structure* of a piece of music. In [2], two methods are proposed for diagonal line detection in the MFCCs' similarity matrix: a "Gaussian distribution extruded over the diagonal with a reinforced diagonal" filter, and a computationally more expensive process: the "Hough Transform". Pattern matching techniques (including deletion, insertion and substitution processes) are then used in order to derive the structure from the detected diagonals. In [9], several algorithms based on dynamic programming are proposed in order to derive the structure using either monophonic pitch estimation, polyphonic transcription or chromagram features.

In part 3, we present a novel method for line detection in the lag-matrix based on a 2D structuring filter. It is combined with a pattern matching algorithm allowing the description of the music in terms of sequence repetitions.

"State" approach: The similarity matrix can be used in order to perform simply (i.e. without going into structure description) the *segmentation* of a piece of music. In [12], a measure of novelty is proposed in order to perform music segmentation into long structures (such as chorus/verse), or into short structures (such as rhythmic structures). This measure of novelty is computed by applying a "checkerboard" kernel to the MFCCs' similarity matrix. The novelty is defined by the values of the convoluted similarity matrix over the main diagonal.

Unsupervised algorithms are most of the time used in order to obtain the *state representation*. A study from Compaq [18] uses the MFCC parameterization in order to create "key-phrases". In this study, the search is not for lower (upper) diagonal (sequence of events) but for states (collection of similar and contiguous states). The song is first divided into fixed length segments, which are then grouped according to a cross-entropy measure. The longest example of the most frequent episode constitutes the "key-phrase" used for the audio summary. Another method proposed by [18] is based on the direct use of a hidden Markov model applied to the MFCCs. While temporal and contiguity notions

are present in this last method, poor results are reported by the authors. In [1], a Hidden Markov Model (with gaussian mixture observation probabilities) is also used for the task of music segmentation into states. Several features are tested: MFCC, Linear Prediction and Discrete Cepstrum. The authors conclude on better results obtained using the MFCCs and reports the limitation of the method due to the necessity to fix a priori the number of states of the HMM.

In part 4, we present a novel multi-pass algorithm combining segmentation and HMM which does not require the a priori fixing of the number of states.

1.4 Audio Summary or Audio Thumbnailing

Music summary generation is a recent topic of interest. As a significant factor resulting from this interest, the recent MPEG-7 standard (Multimedia Content Description Interface) [20], proposes a set of meta-data in order to store multimedia summaries: the Summary Description Scheme (DS). This Summary DS provides a complete set of tools allowing the storage of either sequential or hierarchical summaries.

However, while the storage of audio summaries has been normalized, few techniques exist allowing their automatic generation. This is in contrast with video and text where numerous methods and approaches exist for the automatic summary generation. Without any knowledge of the audio content, the usual strategy is to take a random excerpt from the music signal, or an excerpt in the middle of it. In speech, time-compressed signals, or time-skipped signals are preferred in order to preserve the message. A similar strategy can be applied in music by providing excerpts from meaningful parts of the music derived from its structure.

In part 5, we present a novel approach for audio summary generation based on choosing specific excerpt of the signal derived from the sequence/state representation.

2 Signal Observation: Dynamic Audio Features

Signal features, such as MFCCs, chromagram, ... are computed on a frame by frame basis, usually every 10 ms. Each of these features represents a specific description (description of the spectral shape, of the harmonic content, ...) of the signal *at (around) a given time*. Since the amount of features obtained can be very large (for a 4 minutes piece of music, 4*60*100= 24000 feature vectors), and thus hard to process by computer (the corresponding similarity matrix is 24000*24000) and hard to understand, the features are often "down-sampled" by use of mean (low-pass filter), standard deviation or derivative values (high-pass filter) over a short period of time. However this kind of sampling is not adequate to represent the temporal behavior of the features (for example to represent a regular modulation). Modeling the temporal evolution along time is the goal of the *"dynamic"* features which are proposed here. On the opposite,

we call features, such as MFCCs, chromagram, ... *"static"* features since they don't represent any temporal evolution of a description.

In dynamic features, the evolution of a feature along time is modeled with a Short Time Fourier Transform (STFT) applied to the signal formed by the values of the feature along time (in usual signal processing, the STFT is applied to the audio signal): around each instant t, the time evolution of the feature on a specific duration L is modeled by its Fourier Transform. If the feature is multi-dimensional (as it is the case of the MFCCs), the Fourier Transform is applied to each dimension of the feature.

Among the various types of possible dynamic features, the best results were obtained by modeling the time evolution of the energy output of an *auditory filterbank*. The audio signal $x(t)$ is first passed through a bank of N Mel filters. The slow evolution ([0-50] Hz) of the energy of each output signal $x_n(t)$ of the $n \in N$ filters is then analyzed by Short Time Fourier Transform (STFT). Since the frequencies we are interested in are low, the hop size of the STFT can be large (up to 1s). The output of this is, at each instant t, a matrix $X_{n,t}(\omega)$ representing the amplitude of the variations at several speeds ω of several frequency bands n observed with a window of size L. The feature extraction process is represented in Fig. 1.

Dynamic features can represent slow variations (small ω) at low frequencies (small n) as well as quick variations (large ω) at high frequencies (large n) or any other combination between speed of variation and frequency band.

Another important parameter of dynamic features is the window size L on which the modeling is performed (the length of the analysis window used for the STFT analysis). Using large window tends to favor a "state" representation of the music. Since the modeling is performed on a long duration, the resulting spectrum of the FFT represents properties which are common to all features evolution on a long duration. Therefore the modeling tends to catch the arrangement part of the music. On the opposite, using short window tends to favor the "sequence" representation.

When using dynamic features, only the most appropriate combination of several frequency bands n in several speeds of variation ω for a specific application is kept. This can be done for example by using the first Principal Components of a PCA as has been proposed by [27].

In Fig. 2, Fig. 3, Fig. 4 and Fig. 5, we compare the similarity matrices obtained using MFCCs, dynamic features with short, middle and long duration of the model for the title "Natural Blues" by "Moby" [19].
The title "Natural Blues" by "Moby" is used several times in this paper in order to illustrate our studies. This title has been chosen for several reasons: 1) the structure of the arrangement of the music (the instruments playing in the background) does not follow the structure of the melodic part (usually in popular music both arrangement and melody are correlated). Therefore it provides a good illustration of the difference between the sequence and the state representation; 2) the various melodic parts are all based on sampled vocals. Therefore there is an exact sequence repetitions (usually in popular music, when a chorus is repeated

it can vary from one repetition to the other, therefore the question appears "is it the same melody, a variation of it or a different one ?").
In Fig. 2, the similarity matrix using MFCCs is represented for the first 100 s of the music. We see the repetition of the sequence 1 "oh lordy" $t = [0, 18]$ *at* $t = [18, 36]$*, the same is true for the sequence 2 "went down"* $t = [53, 62]$ *which is repeated at* $t = [62, 71]$*. In Fig. 3, the similarity matrix using dynamic features with a short duration (L=2.56s) is represented for the whole title duration (252 s). Compared to the results of Fig. 2, we see that the sequence 1* $t = [0, 18]$ *is in fact not only repeated at* $t = [18, 36]$ *but also at* $t = [36, 54]$*,* $t = [72, 89]$*,* $[89, 107]$*,* $[160, 178]$*, ... This was not visible using MFCC parameterization because the arrangement of the music changes at time* $t = 36$ *masking the sequence repetition. Note that the features' sampling rate used here is only 4 Hz (compared to 100 Hz for the MFCC). In Fig. 4 and Fig. 5 we compare middle and long duration of the model (L=5.12s and L=10.24s) for the representation of the structure of the arrangement of the song. Fig. 4 and Fig. 5 shows the introduction at* $t = [0, 36]$*, the entrance of the first rhythm at* $t = [36, 72]$*, the main rhythm at* $t = [72, 160]$*, the repetition of the introduction at* $t = [160, 196]$*, the repetition of the main rhythm at* $t = [196, 235]$*, and ending with a third repetition of the introduction at* $t = [235, 252]$*. Fig. 4 shows a more detailed description of the internal structure of each part. Note that the features sampling rate used here is only 2 Hz for Fig. 4 and only 1 Hz for Fig. 5.*

3 Sequence Approach

A high value in the similarity matrix $\underline{\underline{S}}(t_i, t_j)$ represents a high similarity of the observations at times t_i and t_j. If a sequence of events at time $t_i, t_{i+1}, t_{i+2}, \ldots$ is similar to another sequence of events at time $t_j, t_{j+1}, t_{j+2}, \ldots$ we observe a lower (upper) diagonal in the matrix.

The lag between the repetition (starting at t_i) and the original sequences (starting at t_j) is given by projecting t_i on the diagonal of the matrix and is therefore given by $t_i - t_j$. This is represented in the lag-matrix $\underline{\underline{L}}$: $\underline{\underline{L}}(t_i, lag_{ij}) = \underline{\underline{S}}(t_i, t_i - t_j)$. The diagonal-sequences in the similarity-matrix become line-sequences in the lag-matrix. This representation is used here since processing on lines is easier.

As an example lag matrix of the title "Natural Blues" by "Moby" is represented in Fig. 6.

There exist many articles about the similarity-matrix or lag-matrix but few of them address the problem of deriving from this visual representation the actual time pointers to the sequence's repetitions. This is the goal of this section.

Our approach for deriving a sequence representation of a piece of music works in three stages:

1. from the feature similarity/lag matrix we first derive a set of *lines* (a line is defined here as a possibly discontinuous set of points in the matrix) (part 3.1)

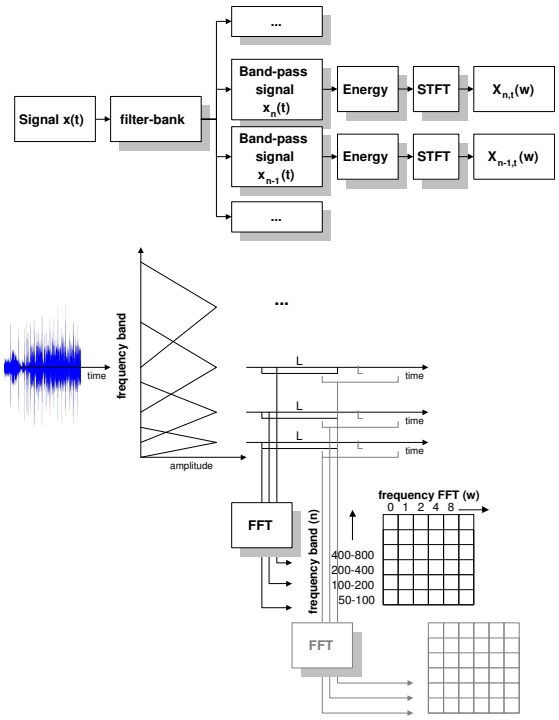

Fig. 1. Dynamic features extraction process. From left to right: the signal is passed through a filterbank; the energy over time of the output signals of each filter (band-pass signal) is estimated; each signals is modeled with a STFT of size L; at each time, the resulting FFT are grouped in a matrix representing the amplitude at each frequency of the FFT (w) for each frequency band (n).

2. from the set of lines we then form a set of *segments* (a segment is defined here as a set of continuous times). Since we only observe repetitions in the similarity/lag matrix, a segment derived from a line is a repetition of some original segment. A segment is therefore defined by the staring time, the ending time of its repetition and a lag to its original segment (part 3.2).
3. from the set of segments (original and repetition segments) we finally derive a *sequence* representation (a sequence is defined by a number and a set of time intervals where the sequence occurs; a sequence representation is defined by a set of sequences) (part 3.3).

The global flowchart of the proposed algorithm for sequence representation is represented in Fig. 18.

3.1 Search for Lines in the Matrix

In order to facilitate the detection of line (or diagonal) in the matrix, usually people [3] [2] first apply 2D filtering to the matrix in order to increase the

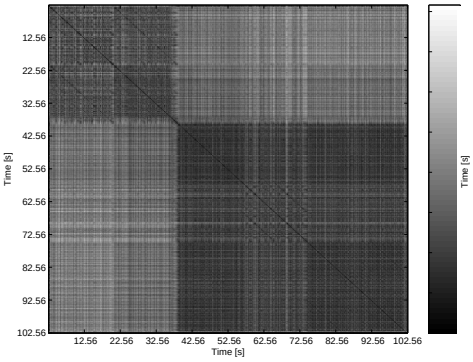

Fig. 2. Similarity matrix using MFCC features for the title "Natural Blues" by "Moby"

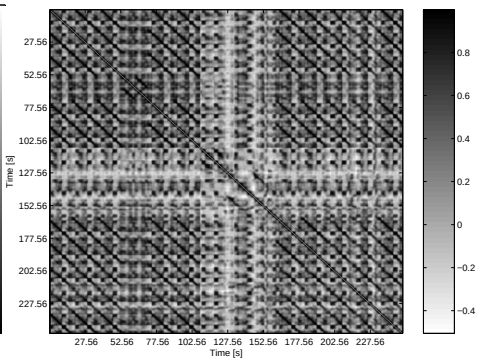

Fig. 3. Similarity matrix using dynamic features with short duration modeling (L=2.56s)

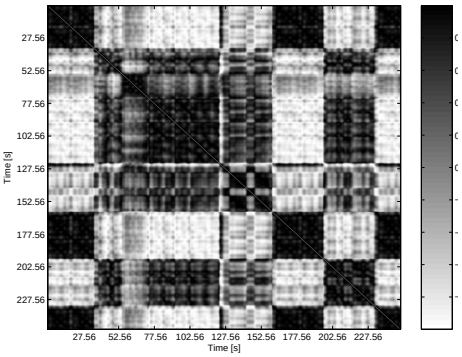

Fig. 4. Similarity matrix using dynamic features with middle duration modeling (L=5.12s)

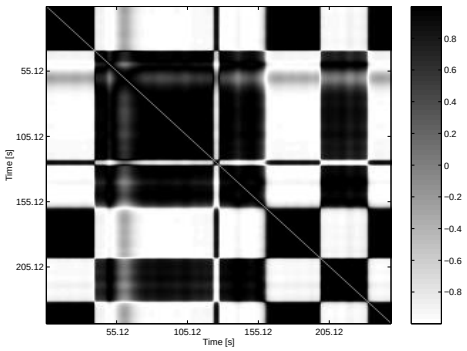

Fig. 5. Similarity matrix using dynamic features with long duration modeling (L=10.24s)

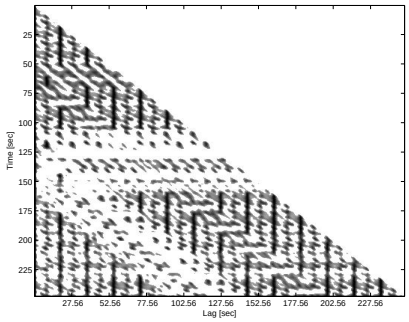

Fig. 6. Lag-matrix (X=lag, Y=time) for the title "Natural Blues" by "Moby"

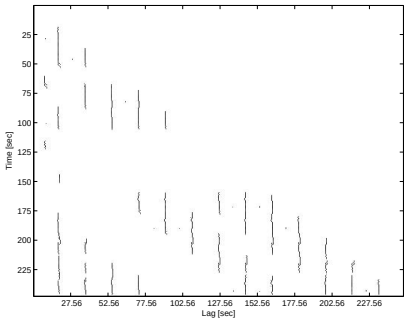

Fig. 7. Results of applying the 2D structuring filter to the lag matrix (X=lag, Y=time)

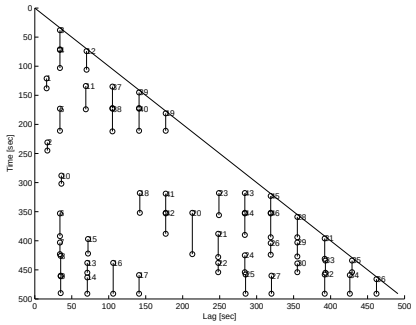

Fig. 8. Detected segments along time (X=lag, Y=time)

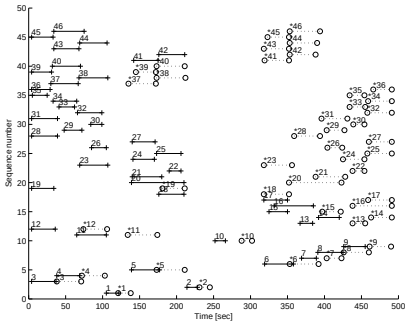

Fig. 9. Original segments (−) and repeated segments (...) along time (X=time, Y=segment number)

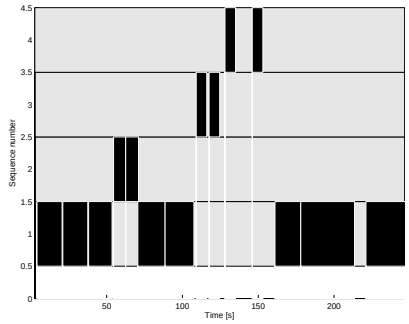

Fig. 10. Detected sequence repetitions along time (X=time, Y=sequence number)

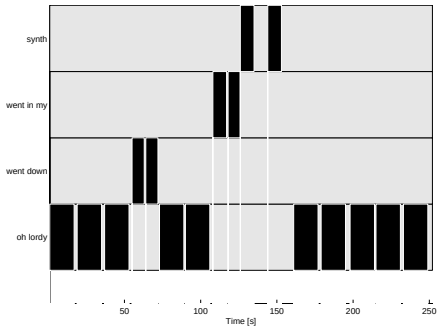

Fig. 11. Real sequence repetitions along time (X=time, Y=sequence number)

contrast between sequences and the so-called "noisy" similarity. For the detection of vertical lines in a lag-matrix, a vertical low-pass and horizontal high-pass filter can be applied. However, use of kernel based filtering techniques, if it allows one to get rid of most of the "noisy" similarity, blurs the values and therefore prevents the detection of the exact start and end positions of a segment. Moreover, it constitutes a step toward the detection of start and end positions of a segment but does not give them (one still needs to find where the sequence starts in the continuous set of values of the matrix). For this reason, we studied the applicability of a 2D structuring filter.

Structuring Filters: For a specific data point y, structuring filters use neighboring values $[y - layy, y + lagy]$ to decide on keeping the value of y or canceling it. This choice is based on the local mean around y. This can be expressed in a MATLAB© way as:

```
if y < mean([y-lagy:y+lagy])
then y=0
else y=y
```

The 2D structuring filter method we propose for vertical lines detection (see Fig. 12) is based on counting the number of values in the neighboring interval $[y-lagy, y+lagy]$ which are above a specific threshold $t1$. If this number is below another threshold $t2$ then y is canceled. This can be expressed in a MATLAB© way as:

```
if y < length( find([y-lagy:y+lagy]) >t1 ) < t2
then y=0
else y=y
```

The first threshold, $t1$, allows one to get rid off the low values in the similarity matrix. The second threshold, $t2$, is proportional to the size of the considered interval: $t2 = k * (2 * lagy + 1)$, where k ranges from 0 (no values need to be above $t1$) to 1 (all values must be above $t1$).

Since a sequence can be repeated at a slower or quicker rate (resulting in a departure of the line-sequence from the column x to its neighboring column $x - lagx$ or $x + lagx$), we extend the 2D structuring filter in order to take also into account the contribution of the neighboring columns $[x - lagx, x + lagx]$: if, for a specific y, at least one of the values on the interval $([x - lagx, x + lagx], y)$ is above a specific threshold $t1$ then a new hit is counted (there is no cumulative total across y).

In order to avoid that all the contribution to the counter would come from a neighboring column, we add the condition that the main contribution to the counter must come from the main column x.

The result of the application of the proposed 2D structuring filter on the lag-matrix of Fig. 6 is represented on Fig. 7.

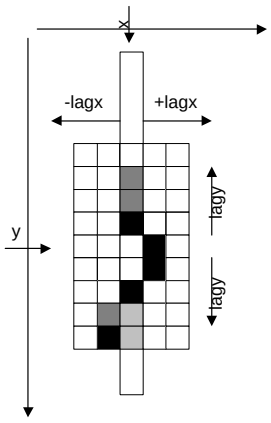

Fig. 12. Two dimensional structuring filter for vertical lines detection

Avoiding doubled lines: Resulting from the previous stage is a subset of lines detected in the lag-matrix. However, because of the extension to the consideration of the neighboring columns x in our algorithm [3]. For an analysis hop size of 0.5 s, and by the definition of a lag-matrix, two neighboring lines represent two repetitions of the same sequence separated by 0.5 s. We remove doubled lines by defining the minimum delay between two sequences' repetition (fixed to a value of 5 s in our experiment - which means that we won't be able to detect a sequence repeated with a period less than 5 s). When several sequences are neighboring, only the longest one is kept.

3.2 From Lines to Segments

Detected lines can be discontinuous (the line suddenly disappears during some values of y). In order to be able to define segments, we need to define the maximum length of a gap, $G_{\max}$, tolerated inside a segment. If the observed gap is larger than $G_{\max}$, then the line is divided into two separated segments. We also define the minimum accepted length of a segment. A detected segment i is now defined by

- its start time $s(i)$,
- its end time $e(i)$,
- its lag from the original segment $lag(i)$

3.3 Interpreting the Detected Segments

Connection between the various segments involves 1) finding which segment is a repetition of which other one 2) finding which segment can be considered as the reference sequence.

[3] Depending on the value of $lagx$, a specific value (x, y) can be shared by several columns), doubled lines detection is possible.

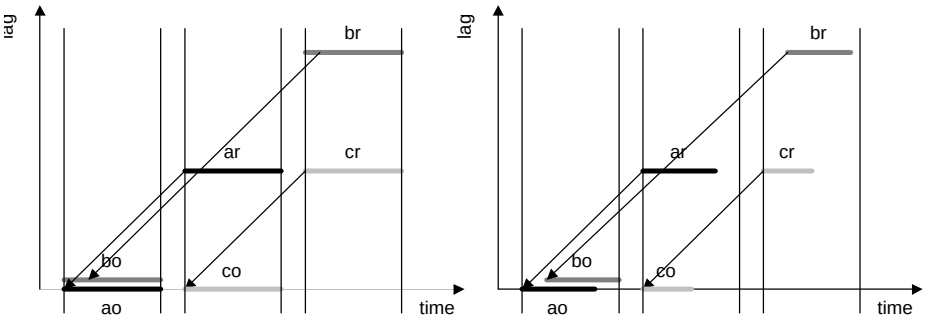

Fig. 13. Sequences connections in the lag-matrix

Each segment i is in fact defined by its repetition segment ir (the detected line) and its original segment io (the detected line translated by lag). In the left part of Fig. 13, we represent the ideal case. The segment a is defined by its repetition ar and its original segment ao. The same is true for b. We see that bo shares the same period of time as ao. a and b are therefore supposed to be identical sequences. This is verified by the presence of the segment c: since ao and bo are the same and ar and ao are the same, there must be a segment c which repetition cr shares the same period of time as br and which original co shares the same period of time as ar. However what we observe in practice is closer to the right part of Fig. 13: bo and ao only share a portion of time (hence the question "which one is the reference ?"); the segment c can be very short, co is in ar but, since it shares a too short period of time with br, it is not in br. This is in contradiction with the transition rule: $cr \rightarrow co \rightarrow ar \rightarrow ao \rightarrow bo \rightarrow br$.

*A real case example is represented in Fig. 8 and Fig. 9 for the same signal as Fig. 6. Fig. 8 represents the detected segments in the lag/time space. Starts and ends of segments are indicated by circles (o). Fig. 9 represents each segments original (−) and repetition (...) along time. X-axis represents the time, Y-axis represents the segment's number. In this case, we need to connect 92 (2*46) segments with each other.*

Proposed algorithm for segments connection: In order to perform the connection between segments, and form sequences, the following algorithm is proposed. Two segments are said to belong to the same sequence if the period of time shared by the two segments is larger than a specific amount of their own duration (we have chosen a value of 70%). If they share less than this amount they are said to be different. The algorithm works by processing segments one by one and adding them to a sequence container. We note

- jO the original of a new segment and jR the repetition of a new segment (the detected line)
- I the sequence container
- iO the original of a segment already present in the sequence container.

```
Init: define min_shared_time=0.7
Init: add first j0 and jR to I
_while there is non-processed segment j
__take a new segment j (original j0 of length j0L)
__if new segment j0 shares time with a i0 in the
container
___for each of these i,
        define i0L = length of i0,
        define j0L = length of j0,
        define ij0L= the shared time length of i0 and j0,
        define c1  = ratios c1=ij0L/j0L,
        define c2  = ratios c2=ij0L/i0L
____select the i with the largest c1+c2
____if c1 > lag | c2 > min_shared_time then repetition
_____if c1 > lag & c2 < min_shared_time then j0 is in i0
_____if c1 < lag & c2 < min_shared_time then i0 is in j0
_____add jR to I with the same sequence tag as i0
____else
_____add j0 and jR to I with a new sequence tag
____end
__else
___add j0 and jR to I with a new sequence tag
__end
_end
```

At this stage, I contains all the segments with an associated "sequence number" tag. The second stage of the algorithm decides, for each sequence number, which of its segments can be considered as the reference segment[4] and which of the segments initially supposed to be part of the sequence are in fact poorly explained by the other segments of this sequence. In order to do that, for each sequence number, each of its segments is in turn considered as the reference segment of the sequence. For each candidate reference segment, we estimate how many and how much of the other segments can be explained by using this reference segment. This is done by computing a score defined as the amount of time shared by the candidate reference segment - original and repetition - and all the other segments[5]. The reference segment is the candidate segment with the highest score. The remaining segments that were highly explained by the reference segment are attributed to this sequence. Finally, the reference segment and the attributed segments are removed from the container I and the process is repeated as long as there are still non attributed segments in I.

[4] The reference segment of a sequence is the one that best explains the other segments of the sequence

[5] The score used is the same as in the previous code; it was noted $c1 + c2$.

3.4 Results

The result of the application of our algorithm of segment connection to the detected segments of Fig. 8 and Fig. 9 is illustrated in Fig. 10 (title "Natural blues" by "Moby"). For comparison, the real structure (manually indexed) is represented in Fig. 11. Three different sequences were detected. Sequence 1 is the "oh lordy, trouble so hard" melody. Sequence 2 is the "went down the hill" melody. Sequence 3 is the "went in the room" melody. Sequence 4 is a melody played by the synthesizers appearing twice. Compared to the real sequence structure of the song, we can see that two occurrences of sequence 1 have not been detected in the last part of the song.

We also illustrate our sequence representation method in Fig. 14, Fig. 15, Fig. 16 and Fig. 17 for title "Love me do" by "The Beatles" [4]. Fig. 14 represents the lag matrix. The segments detected in the matrix using the 2D structuring filters are represented in Fig. 15. The resulting sequence representation is represented in Fig. 16. For comparison, the real structure (manually indexed) is represented in Fig. 17. Three different sequences were detected. Sequence 1 is the harmonica melody played several times across the song, sequence 2 is the "love me do" melody and sequence 3 is the "someone to love" melody. Note that the second occurrence of sequence 3 is in fact the same melody "someone to love" but played by the harmonica. The only false detection is the sequence 1 at time 450.

4 State Approach

The goal of the state representation is to represent a piece of music as a succession of states (possibly at different temporal scales) so that each state represents (somehow) similar information found in different parts of the piece.

The states we are looking for are of course specific for each piece of music. Therefore no supervised learning is possible. We therefore employ unsupervised learning algorithms to find out the states as classes. Several drawbacks of unsupervised learning algorithms must be considered:

- usually a previous knowledge of the *number of classes* is required for these algorithms,
- these algorithms depend on a good *initialization of the classes*,
- most of the time, these algorithms do not take into account contiguity (spatial or temporal) of the observations.

A new trend in video summary is the *"multi-pass"* approach [31]. As for video, human segmentation and grouping performs better when listening (watching in video) to something for the second time [10]. The *first listening* allows the detection of variations in the music without knowing if a specific part will be repeated later. The *second listening* allows one to find the structure of the piece by using the previous mentally created templates. In [23] we proposed a multi-pass approach for music state representation, we review it here briefly. This multi-pass approach allows solving most of the unsupervised algorithm's problems.

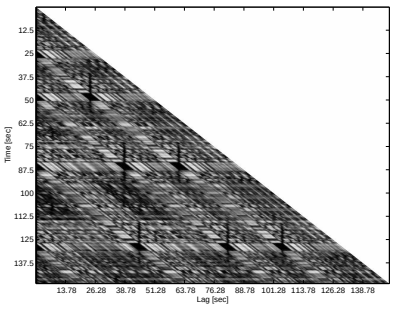

Fig. 14. Lag-matrix (X=lag, Y=time) for the title "Love me do" by "The Beatles"

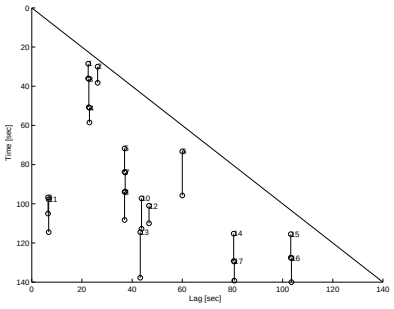

Fig. 15. Detected segments along time (X=lag, Y=time)

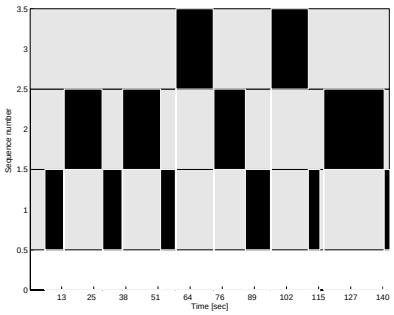

Fig. 16. Detected sequence repetitions along time (X=time, Y=segment number)

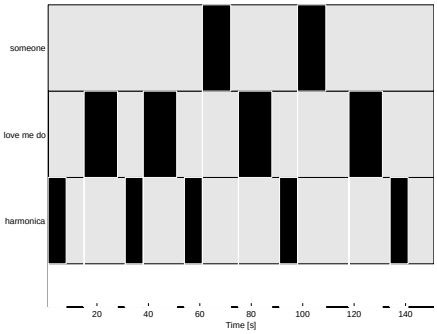

Fig. 17. Real sequence repetitions along time (X=time, Y=sequence number)

The global flowchart of the multi-pass approach for state representation is represented in Fig. 19.

4.1 Multi-pass Approach

First pass: The first pass of the algorithm performs a signal segmentation that allows the definition of a set of templates (classes) of the music.

This segmentation can be either based on a "frame to frame" dissimilarity or a "region to region" dissimilarity. In [23], we used a "frame to frame" dissimilarity criterion. The upper and lower diagonals of the similarity matrix $\underline{\underline{S}}(t)$ of the features $\underline{f}(t)$ (which represent the frame to frame similarity of the features vector) are used to detect large and fast changes in the signal content and segment it accordingly. We found that using a "region to region" dissimilarity, such that provides by the measure of novelty [6] proposed by [12] also gives good results.

[6] The measure of novelty is defined by the values over the main diagonal of the similarity matrix after convolution by a "checkerboard" kernel

In both case, a high threshold[7] is used for the segmentation in order to reduce the "slow variation" effect and to ensure that all times inside a segment are highly similar.

We use the mean values of $\underline{f}(t)$ inside each segment to define "potential" states $\underline{s}_k$. "Potential" states.

Second pass: The second pass uses the templates (classes) in order to define the music structure. The second pass operates in three stages:

1. Nearly identical (similarity ≥ 0.99) "potential" states are grouped. After grouping, the number of states is now K and are called "initial" states. "Potential" and "initial" states are computed in order to facilitate the initialization of the unsupervised learning algorithm since it provides 1) an estimation of the number of states and 2) a "better than random" initialization of it.
2. The reduced set of states (the "initial" states) is used as initialization for a Fuzzy K-means (K-means with probabilistic belonging to classes) algorithm (knowing the number of states and having a good initialization). We note $\underline{s}'_k$ the states' definition obtained at the end of the algorithm and call them "middle" states.
3. In order to take music specific nature into account (not just a set of events but a specific temporal succession of events), the output states of the Fuzzy K-means algorithm are used for the initialization of the learning of a Markov model. Since we only observe $\underline{f}(t)$ and not directly the states of the network, we are in the case of a hidden Markov model (HMM) [24]. A state k produces observations $\underline{f}(t)$ represented by a state observation probability $p(\underline{f}|k)$. The state observation probability $p(\underline{f}|k)$ is chosen as a gaussian pdf $g(\mu_k, \sigma_k)$. A state k is connected to other states j by state transition probabilities $p(k, j)$. Since no priori training on a labeled database is possible we are in the case of ergodic HMM. The *training* is initialized using the Fuzzy K-means "middle" states $\underline{s}'(k)$. The Baum-Welch algorithm is used in order to train the model. The outputs of the training are the state observation probabilities, the state transition probabilities and the initial state distribution.
4. Finally, the optimal representation of the piece of music as a HMM state sequence is obtained by *decoding* the model using the Viterbi algorithm given the signal feature vectors.

4.2 Results

The result of the proposed multi-pass approach is represented in Fig. 20, Fig. 21 and Fig. 24 for three different titles. The left parts of the figures shows the similarity matrix, the right parts of the figures shows the various states detected along time.

[7] The threshold is automatically determined by exhaustively trying all possible values of the threshold and comparing them to the number of segments obtained in each case.

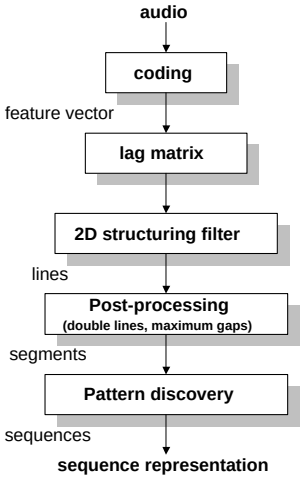

Fig. 18. Sequences representation flowchart

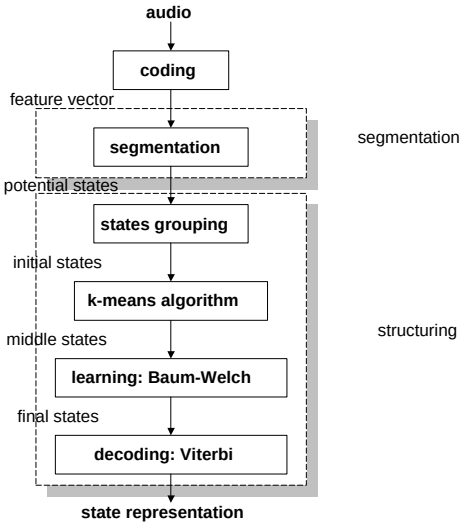

Fig. 19. States representation flowchart

In Fig. 20, for the title "Natural Blues" by "Moby", five different states were found. Fig. 22 represents the "true" structure (manually indexed) of the title. Let us observe the correspondence between the detected states (right part of Fig. 20) and the label of the "true" structure (Fig. 22). State 1 represents the label [intro], state 2 is a mix between [drum1] and [drum2], state 3 is also a mix between [drum1] and [drum2]. Apparently, the algorithm didn't succeeded in catching the difference in the arrangements between [drum1] and [drum2] and rather focused on the variation of the melody (state 3 corresponds to the arrangements on "oh lordy", state 2 to the arrangements on the two other melodies). State 4 is the [synth] part which is correctly identified. State 5 is the [drum2] part which was detected as a separate state.

Fig. 21 represents the title "Smells Like Teen Spirit" by ''Nirvana" [21]. Seven different states were found. Fig. 23 represents the "true" structure (manually indexed) of the title. Let us observe the correspondence between the detected states (right part of Fig. 21) and the label of the "true" structure [(Fig. 23). State 1 represents the label [guitar intro], state 2 seems to be a garbage state containing most drum rolls, state 3 contains the [intro], state 4 is the [verse], state 5 the [transition], state 6 is the [chorus], state 7 represents both the [break] and the [guitar solo]. Considering that the most important part of the title is the (chorus/ transition/ verse/ solo), the detected representation is successful.

In Fig. 24 represents the title "Oh so quiet" by "Bjork" [5]. In this case the "true" structure of the title is hard to derive since most parts when repeated are repeated with large variations of the arrangement. Therefore, we show the obtained structure here only to check whether it makes sense. The characteristic

verse/chorus repetition is very clear. State 3 represents the verse, state 1 the transition to the chorus, state 2 the chorus, state 6 the break, ...

5 Audio Summary Construction

So far, from the signal analysis we have derived features vectors used to assign a sequence number (through line detection in the similarity matrix) or a state number (through unsupervised learning) to each time frame. From this representation several possibilities can be taken in order to create an audio summary. Let us take as example the following structure: $AABABCAAB$. The generation of the audio summary from this sequence/state representation can be done in several ways (see Fig. 25):

- Each: providing a unique audio example of each sequence/state (A, B, C)
- All: reproducing the sequence/state successions by providing an audio example for each sequence/state apparition (A, B, A, B, C, A, B)
- Longest/most frequent: providing only an audio example of the most important sequence/state (in terms of global time extension or in term of number of occurrences of the sequence/state) (A)
- Transition: in the case of state representation: providing audio examples of state transitions $(A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow A)$
- etc ...

This choice relies of course on user preferences but also on time constraints on the audio summary duration. In each case, the audio summary is generated by taking short fragments of the segment/state's signal. For the summary construction, it is obvious that "coherent" or "intelligent" reconstruction is essential.

Information continuity will help listeners to get a good feeling and a good idea of a piece of music when hearing its summary:

- *Overlap-add:* The quality of the audio signal can be further improved by applying an overlap-add technique of the audio fragment.
- *Tempo/Beat:* For highly structured music, beat synchronized reconstruction allows improving largely the quality of the audio summary. This can be done 1) by choosing the size of the fragments as integer multiple of 4 or 3 bars, 2) by synchronizing the fragments according to the beat position in the signal. In order to do that, we have used the tempo detection and beat alignment proposed by [26].

The flowchart of the audio summary construction of our algorithm is represented in Fig. 26.

6 Discussion

6.1 Comparing "sequence" and "state" approach

The "sequence" approach aims at detecting the repetition of sequences in the music, i.e. detecting two identical succession of events in the music such that

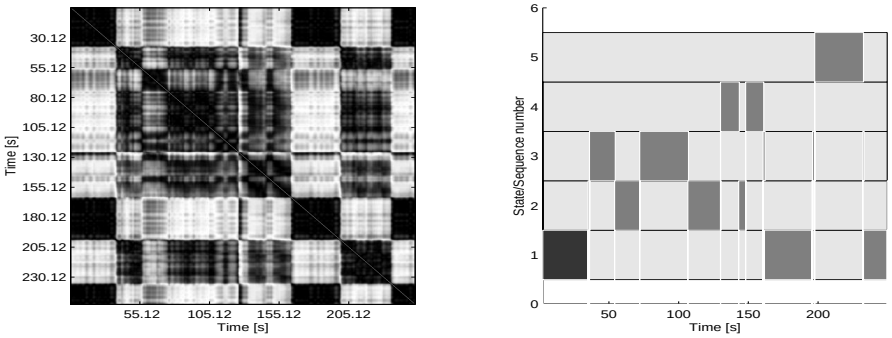

Fig. 20. State approach applied to the title "Natural Blues" by "Moby", [left] similarity matrix, [right] detected states along time

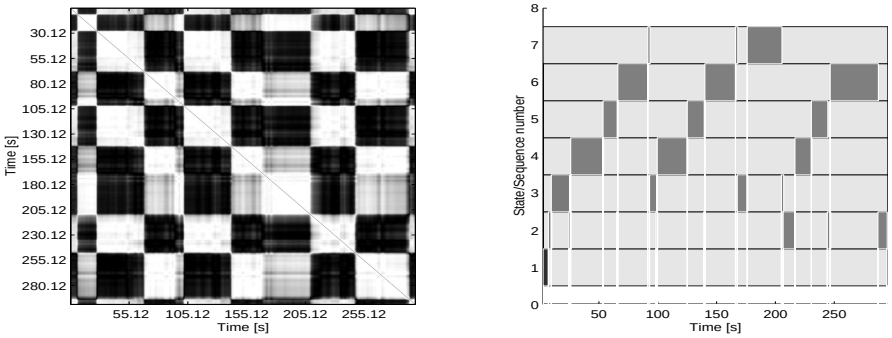

Fig. 21. State approach applied to the title "Smells like teen spirit" by "Nirvana", [left] similarity matrix, [right] detected states along time

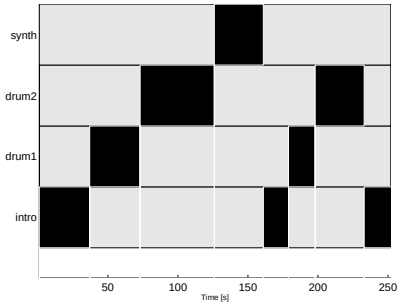

Fig. 22. "True" states along time of the title "Natural Blues" by "Moby"

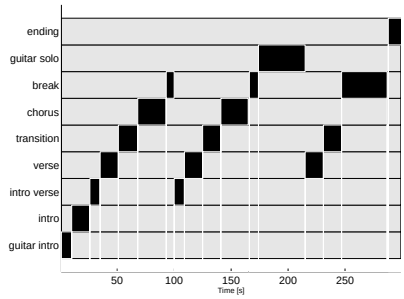

Fig. 23. "True" states along time of the title "Smells like teen spirit" by "Nirvana"

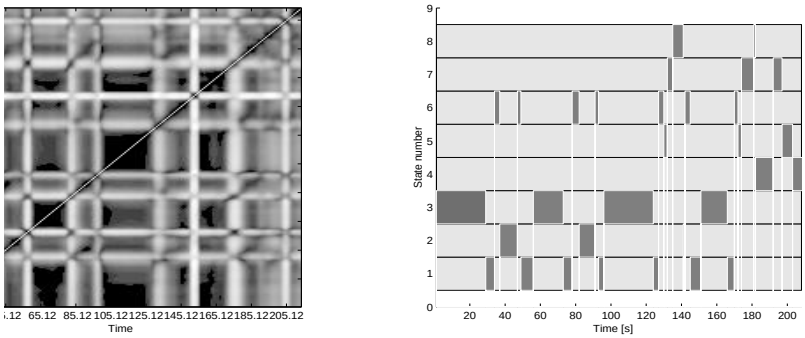

Fig. 24. State approach applied to the title "Oh so quiet" by "Bjork", [left] similarity matrix, [right] detected states along time

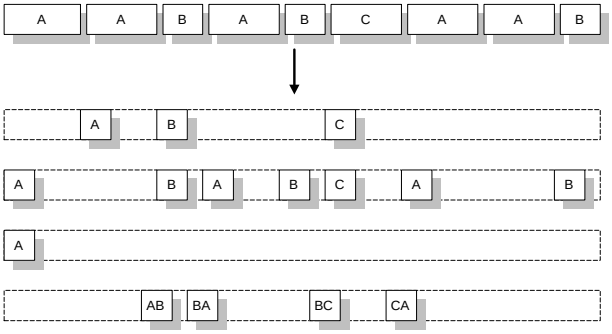

Fig. 25. Various possibilities for Audio Summary construction from state representation

each event of the first sequence is similar to its equivalent event in the second sequence but not to the other events in the sequence[8]. An example of sequence repetition in the case of popular music is the repetition of the successive notes of a melody.

The "state" approach aims at representing the music as a succession of states, such that a state is composed by a set of contiguous times with (somehow) similar events, and that two set of contiguous times of the music belonging to the same state represent (somehow) similar events.

In the state approach, all analysis times are attached to a state and it is possible that a state appears only once. In the sequence approach, only times belonging to a line detected in the similarity matrix belong to a sequence. A sequence appears at least twice (original and repetition). Because of this required lines detection step in the sequence approach, the sequence approach is less

[8] Therefore a sequence representation cannot be derived from a state representation since all events inside a state are supposed to be similar.

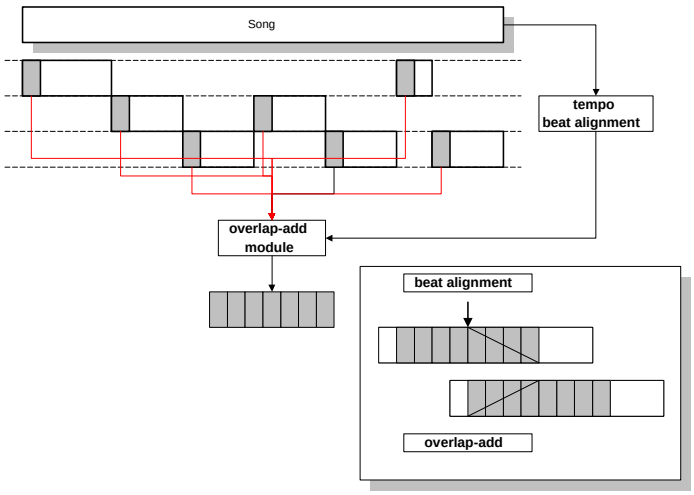

Fig. 26. Audio summary construction from sequence/state representation; details of fragments alignment and overlap-add based on tempo detection/ beat alignment

robust than the state approach. It is also computationally more expensive (need for a highest temporal resolution, need to compute a similarity matrix, need to perform the computationally expensive line detection process).

6.2 Evaluation

Evaluating the quality of the results obtained by the two approaches is a difficult task. The comparison of the structure obtained with the proposed algorithms with the "true" structure involves first deciding on what is the "true" structure. This last point is often subject of controversy among people (when is a melody repeated exactly, when is it a variation, when does this variation make it a different one ?). An example of this occurs in the title "Oh so quiet" from "Bjork". However the results obtained so far were found good by users.

Among the various types of audio summary, the most reliable one was found to be the "each" method. This is due to the fact that the longest/most-frequent sequence/state in a piece of music is not necessarily the most important music key for listeners. Moreover, deciding on a unique sequence/state makes the summary more sensitive to algorithm errors. The summary provided by the "all" method can be long and is redundant; however it is the only one that can remind to listeners the overall temporal structure of the piece of music. The "each" method, since it provides all important music key, rarely fails to provide the listener's preferred music key.

6.3 Limitations of music structure discovery from signal analysis:

Deriving content information directly from the signal implies a severe limitation: we do not observe the intention of the piece of music (the score) but (one of) its realization(s). Moreover, we only observe it through what we can actually automatically extract from it (considering the current limitations of digital signal processing). This implies especially limitations concerning multiple pitch estimation (chords estimation) and mixed sources (several instruments playing at the same time). Because of that, we will also be limited in deriving a single structure for the whole set of instruments (considering current source separation algorithm development, it is still difficult to separate the various instruments hence to obtain the structure for each instrument independently). Therefore, caution must be taken when concluding in the ability of a specific method to derive the actual musical structures. Indeed, except for music for which the musical arrangements (instruments playing in the background) are correlated with the melody part, or when the melody part is mixed in the foreground, there is little chance to be able to derive the actual melody repetitions. Moreover, since most music structure discovery methods are based on a search for repetitions, evolution of motives or of melodies are unlikely to be discovered.

7 Conclusion

In this paper we studied a "sequence" and "state" representation for music structure detection with the aim of generating visual and audio summaries. We introduced dynamic features, which seem to allow deriving powerful information from the signal for both 1) detection of sequences repetition in the music (lower/upper diagonals in a similarity matrix) and 2) representation of the music in terms of "states".

We proposed a 2D structuring filter algorithm for lines detection in the lag matrix and an algorithm to derive a sequence representation from these lines. We proposed a multi-pass algorithm based on segmentation and unsupervised learning (fuzzy-kmeans and HMM) for state representation of the music. We finally investigated sequential summaries generation from both sequential and state representations.

Perspectives

Combining both segment and state approach: Further work will concentrate on combining both sequence and state approaches (by using for example two different window length, it is possible to obtain both representation at the same time). It is clear that both approaches can help each other since the probability of observing a given sequence at a given time is not independent from the probability of observing a given state at the same time.

Hierarchical summaries: Further works will also concentrate on the development of hierarchical summary [16]. Depending on the type of information desired,

the user should be able to select the "level" in a tree structure representing the piece of music (for example the various sub-melodies composing a melody or the various parts of a chorus/ verse).

Acknowledgments

Part of this work was conducted in the context of the European I.S.T. project CUIDADO [30] (http://www.cuidado.mu). Thanks to Amaury La Burthe (during its stay at Ircam), Kasper Souren and Xavier Rodet for the fruitful discussions .

References

1. J.-J. Aucouturier and M. Sandler. Segmentation of musical signals using hidden markov models. In *AES 110th Convention*, Amsterdam, The Netherlands, 2001.
2. J.-J. Aucouturier and M. Sandler. Finding repeating patterns in acoustic musical signals: applications for audio thumbnailing. In *AES 22nd International Conference on Virtual, Synthetic and Entertainment Audio*, Espoo, Finland, 2002.
3. M. Bartsch and G. Wakefield. To catch a chorus: Using chroma-based representations for audio thumbnailing. In *WASPAA*.
4. T. Beatles. Love me do (one, the best of album). Apple, Capitol Records, 2001.
5. Bjork. It's oh so quiet (post album). Mother records, 1995.
6. E. Cambouropoulos, M. Crochemore, C. Iliopoulos, L. Mouchard, and Y. Pinzon. Algorithms for computing approximate repetitions in musical sequences. In R. R. Simpson and J., editors, *10th Australasian Workshop On Combinatorial Algorithms*, pages 129–144, Perth, WA, Australia, 1999.
7. M. Cooper and J. Foote. Automatic music summarization via similarity analysis. In *ISMIR*, Paris, France, 2002.
8. T. Crawford, C. Iliopoulos, and R. Raman. String matching techniques for musical similarity and melodic recognition. In *Computing in Musicology*, volume 11, pages 73–100. MIT Press, 1998.
9. R. Dannenberg. Pattern discovery techniques for music audio. In *ISMIR*, Paris, 2002.
10. I. Deliege. A perceptual approach to contemporary musical forms. In N. Osborne, editor, *Music and the cognitive sciences*, volume 4, pages 213–230. Harwood Academic publishers, 1990.
11. J. Eckman, S. Kamphorts, and R. Ruelle. Recurrence plots of dynamical systems. *Europhys Lett*, (4):973–977, 1987.
12. J. Foote. Automatic audio segmentation using a measure of audio novelty. In *ICME (IEEE Int. Conf. Multimedia and Expo)*, page 452, New York City, NY, USA, 1999.
13. J. Foote. Visualizing music and audio using self-similarity. In *ACM Multimedia*, pages 77–84, Orlando, Florida, USA, 1999.
14. J. Foote. Arthur: Retrieving orchestral music by long-term structure. In *ISMIR*, Pymouth, Massachusetts, USA, 2000.
15. M. Hunt, M. Lennig, and P. Mermelstein. Experiments in syllable-based recognition of continuous speech. In *ICASSP*, pages 880–883, Denver, Colorado, USA, 1980.

16. A. Laburthe. *Resume sonore*. Master, Universite Joseph Fourier, Grenoble, France, 2002.
17. K. Lemstrom and J. Tarhio. Searching monophonic patterns within polyphonic sources. In *RIAO*, pages 1261–1278, College of France, Paris, 2000.
18. B. Logan and S. Chu. Music summarization using key phrases. In *ICASSP*, Istanbul, Turkey, 2000.
19. Moby. Natural blues (play album). Labels, 2001.
20. MPEG-7. Information technology - multimedia content description interface - part 5: Multimedia description scheme, 2002.
21. Nirvana. Smells like teen spirit (nevermind album). Polygram, 1991.
22. N. Orio and D. Schwarz. Alignment of monophonic and polyphonic music to a score. In *ICMC*, La Habana, Cuba, 2001.
23. G. Peeters, A. Laburthe, and X. Rodet. Toward automatic music audio summary generation from signal analysis. In *ISMIR*, Paris, France, 2002.
24. L. Rabiner. A tutorial on hidden markov model and selected applications in speech. *Proccedings of the IEEE*, 77(2):257–285, 1989.
25. S. Rossignol. *Segmentation et indexation des signaux sonores musicaux*. Phd thesis, Universite Paris VI, Paris, France, 2000.
26. E. Scheirer. Tempo and beat analysis of acoustic musical signals. *JASA*, 103(1):588–601, 1998.
27. K. Souren. Extraction of structure of a musical piece starting from audio descriptors. Technical report, Ircam, 2003.
28. G. Tzanetakis and P. Cook. Multifeature audio segmentation for browsing and annotation. In *WASPAA*, New Paltz, New York, USA, 1999.
29. D. VanSteelant, B. DeBaets, H. DeMeyer, M. Leman, S.-P. Martens, L. Clarisse, and M. Lesaffre. Discovering structure and repetition in musical audio. In *Eurofuse*, Varanna, Italy, 2002.
30. H. Vinet, P. Herrera, and F. Pachet. The cuidado project. In *ISMIR*, Paris, France, 2002.
31. H. Zhang, A. Kankanhalli, and S. Smoliar. Automatic partitioning of full-motion video. *ACM Multimedia System*, 1(1):10–28, 1993.