MUSIC STRUCTURE DISCOVERING USING DYNAMIC AUDIO FEATURES FOR AUDIO SUMMARY GENERATION: "SEQUENCE" AND "STATE" APPROACH

Geoffroy Peeters

Ircam
Analysis/Synthesis Team
1, pl. Igor Stravinsky - 75004 Paris- France
peeters@ircam.fr

Xavier Rodet

Ircam
Analysis/Synthesis Team
1, pl. Igor Stravinsky - 75004 Paris- France
rod@ircam.fr

ABSTRACT

Music structure discovery from signal analysis is a recent topic of interest. In this paper, we study two approaches - the "sequence" and the "state" structure representations with the aim of creating automatically music visual and audio summary. We introduce new type of audio features, the dynamic audio features in order to represents the temporal evolution of musical content. The "sequence" approach aims at representing the music as repetitions of sequences of events (repetitions of the melody in popular music). It is derived from the similarity matrix by a proposed algorithm based on 2D structuring filter and segments connection. The "state" approach aims at representing the music as a succession of "states" (various arrangements of chorus/verse in popular music). It is derived using a proposed two-pass approach using unsupervised learning methods (K-means and hidden Markov model). Both "sequence" and "state" representation are used for the creation of audio summary. Various techniques are proposed in order to achieve this.

1. INTRODUCTION

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, and rhythm) [4]. 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.

1.1. Similarity

Music structure discovery from signal analysis methods 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.

The similarity measure requires the choice of signal observations (often-called signal features) and the definition of a distance between the observations (or groups of observations). Various types of *signal features* have been proposed for the task of music structure discovery: Mel Frequency Cepstral Coefficients (MFCCs) [10], Mean and standard deviation of MFCCs [17], Chromagram (bins of a logarithmic spectrum grouped according to their pitch-class) [3] , Scalar features (such as the spectral centroid, spectral rolloff, ...) [16] [9] [17]. Various types of *distances* are used such as the Euclidean distance, scalar product, cosine distance, normalized correlation, symmetric Kullback-Leibler distance or the Mahalanobis distance.

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.

1.2. Visual representation

The most common tools for visually representing music structures is the similarity matrix. It was first proposed by [7] under the name "Recurrence Plots" for visually representing non-stationarity of series of data and latter used in [8] for music structures representation. 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.

1.3. Sequence and State approach

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

1.3.1. "Sequence" approach

The "sequence" approach considers the music audio signal as a repetition of sequences of events. These methods rely mainly on the analysis of the similarity matrix.

Foote showed in [8] that a similarity matrix applied to well-chosen features (MFCC in [8]) allows a visual representation of the structural information of a piece of music, especially the detection of repetitions of sequences through the lower (upper) diagonals of the matrix. The similarity matrix can be used for *discovering the underlying structure* of a piece of music. [2] propose a method combining Gaussian distribution filter or the "Hough Transform" for diagonal detection and pattern matching techniques for structure derivation. [5] propose several algorithms based on dynamic programming for structure derivation using either monophonic pitch estimation, polyphonic transcription or chromagram features.

1.3.2. "State" approach

The "state" approach considers the music audio signal as a succession of states so that each state represents (somehow) similar information found in different parts of the piece. These methods rely mainly on clustering algorithms.

In order to obtain the *state representation*, unsupervised algorithms are most of the time used. A study from Compaq [11] 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 a summary. Another method proposed by [11], close to the method proposed by [1], is based on the direct use of a hidden Markov model applied to the MFCC. While temporal and contiguity notions are present in this last method, poor results are reported by the authors.

1.4. Audio Summary

Music audio summary is a recent topic of interest driven by numerous applications: quick preview of online music catalog items, accessing music database by specific inner-keys, browsing through music items, ... While the storage of audio summaries has now been normalized by the recent MPEG-7 standard (Multimedia Content Description Interface) [12] (the `xml` Summary Description Scheme), 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.

1.5. Organization of the paper

In part 2, we present new types of audio features called dynamic features. In part 3, we study the sequence approach with a novel method for diagonal detection in the matrix and sequences repetition detection. In part 4, we study the state approach with a novel multi-pass algorithm combining segmentation and HMM. 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. DYNAMIC AUDIO FEATURES

All previously mentioned signal features (MFCC, chromagram, ...) are called *"static"* features because 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*. In order to get the evolution along time of this description, we need the succession of this feature along time (succession of MFCC, chromagram along time).

In the opposite, *"dynamic"* features represent directly the *evolution along time* of the features. The evolution of a feature along time is modeled with a Short Time Fourier Transform applied to the values of the feature along time (rather than to the values of the signal along time in usual STFT): around each time instant t, the time evolution of the feature on a specific duration L is modeled by a Fourier Transform. If the feature is multi-dimensional (as it is the case of the MFCCs), the same process is applied to each dimension.

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). The output of this is, at each instant t, a matrix $X_{n,t}(\omega)$ representing the amplitude of the variations at several speed ω of several frequency band 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 more sophisticated combinations between speed of variation and frequency.

The window size L used for the STFT analysis of $x_n(t)$ determines the kind of structure (short term or long term) that we will be able to derive from signal analysis favoring

one of the two approaches: • short duration of the model → sequence approach, • long duration of the model → state approach.

Several advantages come from the use of dynamic features: 1) for an appropriate choice of ω, n and L, the search for repeated patterns in the music can be far easier, 2) the amount of data (and therefore also the size of the similarity matrix) can be greatly reduced: for a 4 minute long excerpt, the size of the similarity matrix is around 24000*24000 in the case of the MFCCs (analysis hop size of 10ms), it can be only 240*240 in the case of the "dynamic" features (analysis hop size of 1s) .

In Fig. 2 , Fig. 3 and Fig. 4 , we compare the similarity matrix obtained using MFCCs, Dynamic features with short and long duration of the model on the title "Natural Blues" by Moby. In Fig. 2 , the similarity matrix using static MFCCs is represented for the first 100 s of the music. We see the repetition of the sequence $t = [0, 18]$ *at* $t = [18, 36]$*, the same is true for* $t = [53, 62]$ *which is repeated at* $t = [62, 71]$*. In Fig. 3 , dynamic features with a short duration (*L*=2.56) of the model is performed. Compared to the previous results, we see that the sequence* $t = [0, 18]$ *is in fact not only repeated at* $t = [18, 36]$ *but also at* $t = [36, 54]$*,* $t = [71, 89]$*, ... 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' sample rate used here is only 4 Hz (compared to 100 Hz for the MFCC). In Fig. 4 , dynamic features with a long duration (*L*=10.24) of the model is performed. The structure in terms of arrangements shows the introduction at* $t = [0, 36]$*, the entrance of the first rhythm* $t = [36, 72]$*, the main rhythm* $t = [72, 160]$*, the break* $t = [160, 196]$*, the main rhythm* $t = [196, 235]$*, and ending with a repetition of the introduction* $t = [235, 252]$*. Note that the features sample rate used here is only 1 Hz.*

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.

An example of the similarity matrix and the corresponding lag matrix is represented in Fig. 5 and Fig. 6 for the title "Love me do" by "The Beatles".

There exist many articles about the similarity-matrix or lag-matrix but few of them address the problem of deriving

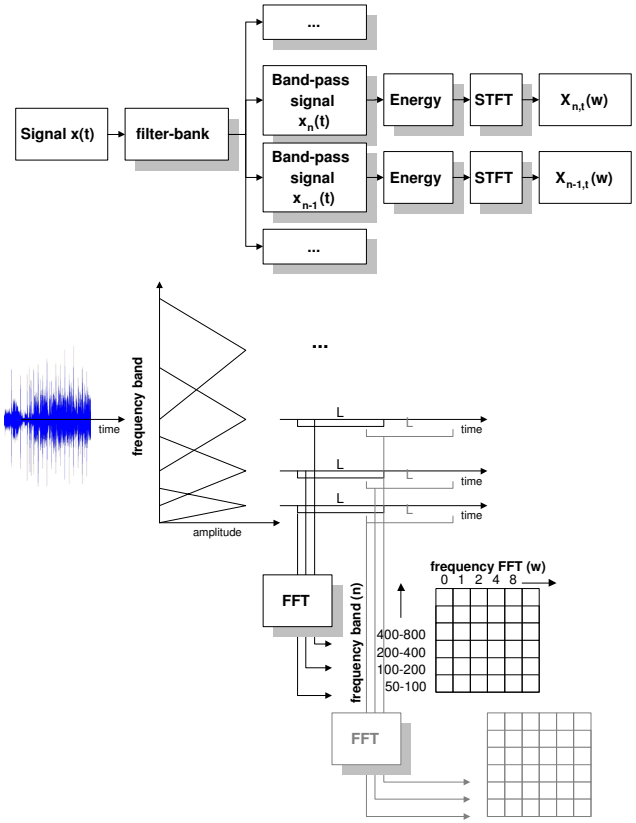

Fig. 1. Dynamic features extraction process from signal. From left to right: signal, filter bank, output of filter bank, output of each of filter, STFT of the outputs signal

from this visual representation the actual time pointers to the sequence's repetitions. This is the goal of this section.

The global flowchart of the proposed algorithm for sequence representation is represented in Fig. 12 .

3.1. Diagonals/lines detection in the matrix

In order to facilitate the detection of line-sequences (or diagonal-sequences) in the matrix, usually people first apply 2D kernel filters to the matrix in order to increase the contrast between sequences and the so-called "noisy" similarity. 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 sequence. For this reason, we studied the applicability of 2D structuring filters.

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. The 2D structuring filter method we propose for vertical lines detection (see Fig. 7) 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 $t2$

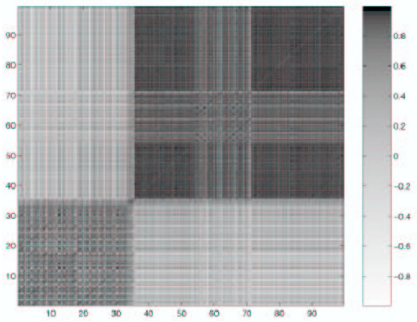

Fig. 2. Similarity matrix using MFCC features on the title “Natural Blues” by “Moby”

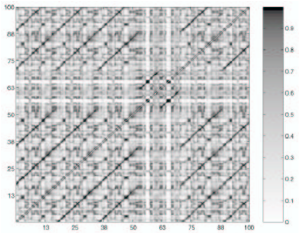

Fig. 3. Similarity matrix using Dynamic features with short duration modeling on the title “Natural Blues” by “Moby”

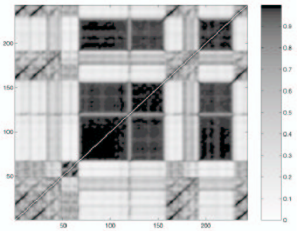

Fig. 4. Similarity matrix using Dynamic features with long duration modeling on the title “Natural Blues” by “Moby”

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 $[x - lagx, x + lagx]$ but we require 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. 8 .

Post-processing: Doubled lines detected in the matrix are then removed by defining the minimum delay between two sequences’ repetition (we’ve chosen a value of 5 s). The

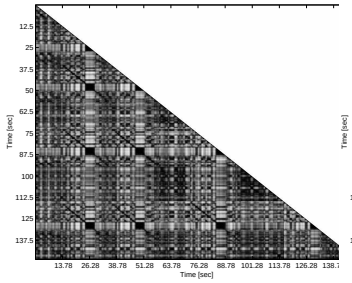

Fig. 5. Similarity-matrix (X=time, Y=time) for the title “Love me do” by “The Beatles”

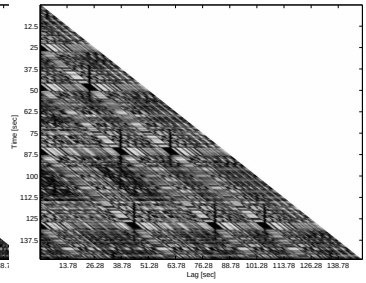

Fig. 6. Lag-matrix (X=lag, Y=time)

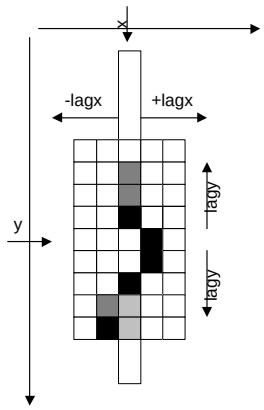

Fig. 7. Two dimensional structuring filter for vertical lines detection

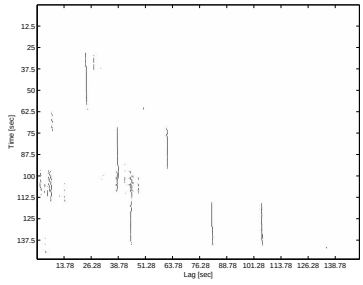

Fig. 8. Applying 2D structuring filter to the lag matrix (X=lag, Y=time)

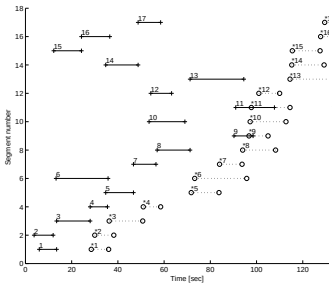

Fig. 9. Original segments (−) and repeated segments (. . .) along time (X=time, Y=segment number)

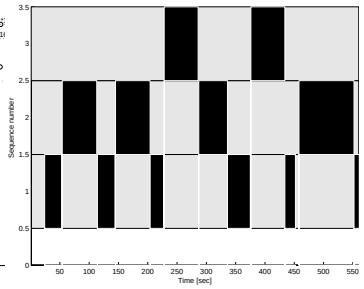

Fig. 10. Sequences repetition along time (X=time, Y=segment number)

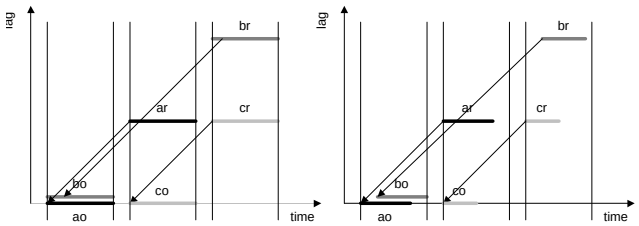

Fig. 11. Sequences connections in the lag-matrix

remaining detected lines are then processed in order to avoid "gaps" inside a segment and avoid too short segments.. A maximum tolerated gaps is defined. Lines without or with a smaller gaps form segments. Lines with a larger gaps are splitted into two separated segments.

A detected segment i is now defined by: • its start time $s(i)$, • its end time $e(i)$, • its lag from the original segment $lag(i)$

3.2. Interpreting the detected diagonals/lines

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.

Each segment i is in fact defined by its repetition segment ir (the line detected in the matrix) and its original segment io (the line detected translated by lag). In the left part of Fig. 11 , 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. 11 : 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. 9 for the same signal as Fig. 6 . 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 34 (2*17) segments with each other.*

Proposed algorithm for connecting segments between each other and forming sequences: 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 I.

```
Init: define T=0.7
Init: add first jO and jR to I
_while there is non-processed segment j
__take a new segment j
__if jO shares time with a iO in I
___for each of these i,
     define
        iOL= length of iO,
        jOL= length of jO,
        ijOL= the shared time of iO and jO,
        c1= ijOL/jOL,
        c2= ijOL/iOL
____select the i with the largest c1+c2
____if c1 > lag | c2 > T then repetition
_____if c1 > lag & c2 < T then jO is in iO
_____if c1 < lag & c2 < T then iO is in jO
_____add jR to I with the same sequence tag as iO
____else
_____add jO and jR to I with a new sequence tag
____end
__else
___add jO and jR to I with a new sequence tag
__end
_end
```

3.3. Results

The result of applying our algorithm for segments' connection to the detected segments of Fig. 9 is illustrated in Fig. 10 (title "Love me do" by "The Beatles". Fig. 10 represents the detected sequences along time. 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 was 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 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.

A new trend in video summary is the *"multi-pass"* approach [19]. As for video, human segmentation and grouping performs better when listening (watching in video) to

something for the second time [6]. 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 [13] 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: • knowledge of the *number of classes* required, • good *initialization of the classes* required, • difficulty to take spatial or temporal contiguity of the observations into account.

The global flowchart of the multi-pass approach for state representation is represented in Fig. 13 .

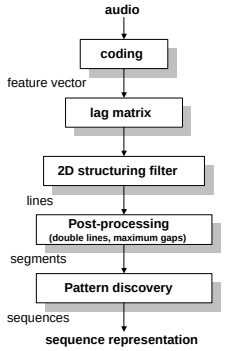

Fig. 12. Sequences representation flowchart

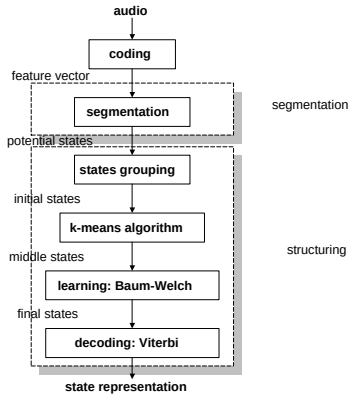

Fig. 13. States representation flowchart

4.1. A 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. In order to do that, 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. A high threshold (similarity ≤ 0.99) is used for the segmentation in order to reduce the "slow variation" effect. We use the mean values of $\underline{f}(t)$ inside each segment to define "potential" states $\underline{s}_k$.

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) [14]. 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. 14 and Fig. 15 . The left parts of the figures represent the similarity matrix, the right parts of the figures represent the various states detected along time. In Fig. 14 , the states represent the title "Oh so quiet" by "Bjork". The characteristic chorus/verse repetition is very clear. State 3 represents the verse, state 1 the transition to the chorus, state 2 the chorus, state 6 the break, ... In Fig. 15 , the states represent the title "Zombie" by "The Cranberries". State 1 represents the guitar introduction, state 6/3 the verse, state 3 the verse, state 5 the transition to the chorus, state 4 the chorus, ...

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.

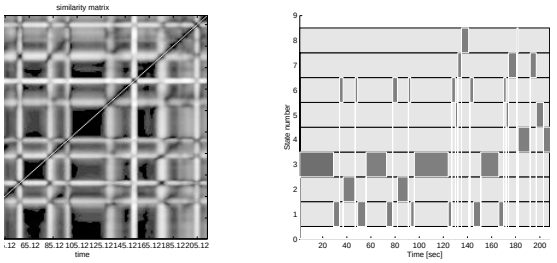

Fig. 14. State approach applied to the title "Oh so quiet" by "Bjork" [top] similarity matrix [bottom] state number along time

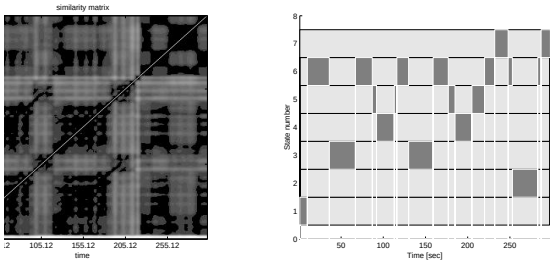

Fig. 15. State approach applied to the title "Zombie" by "The Cranberries" [top] similarity matrix [bottom] state number along time

Let us take as example the following structure: $A\ A\ B\ A\ B\ C\ A\ A\ B$. The generation of the audio summary from this sequence/state representation can be done in several ways: • providing a unique audio example of each sequence/state (A, B, C) • reproducing the sequence/state successions by providing an audio example for each sequence/state apparition (A, B, A, B, C, A, B) • 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) • 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

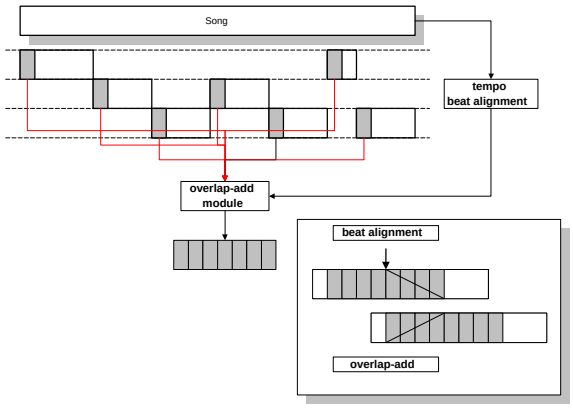

Fig. 16. Audio summary construction from sequence/state representation; details of fragments alignment and overlap-add based on tempo detection/ beat alignment

used the tempo detection and beat alignment proposed by [15].

The flowchart of the audio summary construction of our algorithm is represented in Fig. 16 .

6. DISCUSSION AND CONCLUSION

Pro and cons of "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 as two identical repetitions of the successive notes of a melody). The "state" approach aims at representing the music as a succession of states, such that two segments of the music belonging to the same state represent a (somehow) similar information (all the analysis times inside the first segment are supposed to be similar to each other and similar to all the analysis times inside the second segment). Of course a sequence representation could be derived from a state representation (in the state approach, all the analysis times inside the first segment are at least similar to all the analysis times inside the second segment). But in this case, the obtained representation would have another meaning (interpreting this representation as a sequence would mean that we consider that all events in a sequence can be similar, or all notes in a melody can be similar). The sequence approach we studied is based on the detection of lines (not blocks) in the similarity matrix. In this case, the successive analysis times of a sequence are not similar between each other but similar one by one to the successive analysis times of another sequence. In the state approach, all analysis times are attached to a state. The state algorithm in fact simply attaches the analysis times to a state (although the algorithm also needs to define what are the states). In the sequence approach, only analysis times belonging to a line detected in the similarity matrix belong to a sequence. Because of this required lines

detection step, the sequence approach is less robust than the state approach. The sequence approach is also computationally more expensive: the temporal resolution required is larger (a 0.25 ms resolution was used for the sequence approach while we used only a 1 s resolution for the state representation), the need to compute the whole similarity matrix since each time can be a priori similar to any other time (the computation of the whole similarity matrix is not necessary in the state representation since clustering algorithms can be used), the computationally expensive process for line detection in the matrix.

Combining both segment and state approach: Because the sequence approach is less robust than the state approach, further works will concentrate on combining both sequence and state approaches (by using two different set of parameters for the dynamic audio features, such as various STFT 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.

Evaluation: Further works will also concentrate on evaluating the results obtained by both approaches. While the results obtained so far were found good by users, remains the comparison of the obtained structure with the real structure of a piece of music which is not an easy task since the real structure of a piece of music is often subject to discussion (when is a melody repeated exactly, when is it a variation of it, when this variation makes it a different one ?).

Acknowledgment

Part of this work was conducted in the context of the European I.S.T. project CUIDADO [18] (http:www.cuidado.mu) Part of this work was conducted with Amaury La Burthe (amaury@csl.sony.fr) during its stay at Ircam.

7. 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. 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.

[5] R. Dannenberg. Pattern discovery techniques for music audio. In *ISMIR*, Paris, 2002.

[6] 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.

[7] JP. Eckman, SO. Kamphorts, and R. Ruelle. Recurrence plots of dynamical systems. *Europhys Lett*, (4):973–977, 1987.

[8] J. Foote. Visualizing music and audio using self-similarity. In *ACM Multimedia*, pages 77–84, Orlando, Florida, USA, 1999.

[9] J. Foote. Arthur: Retrieving orchestral music by long-term structure. In *ISMIR*, Pymouth, Massachusetts, USA, 2000.

[10] M. Hunt, M. Lennig, and P. Mermelstein. Experiments in syllable-based recognition of continuous speech. In *ICASSP*, pages 880–883, Denver, Colorado, USA, 1980.

[11] B. Logan and S. Chu. Music summarization using key phrases. In *ICASSP*, Istanbul, Turkey, 2000.

[12] MPEG-7. Information technology - multimedia content description interface - part 5: Multimedia description scheme, 2002.

[13] G. Peeters, A. Laburthe, and X. Rodet. Toward automatic music audio summary generation from signal analysis. In *ISMIR*, Paris, France, 2002.

[14] L. Rabiner. A tutorial on hidden markov model and selected applications in speech. *Proccedings of the IEEE*, 77(2):257–285, 1989.

[15] E. Scheirer. Tempo and beat analysis of acoustic musical signals. *JASA*, 103(1):588–601, 1998.

[16] G. Tzanetakis and P. Cook. Multifeature audio segmentation for browsing and annotation. In *WASPAA*, New Paltz, New York, USA, 1999.

[17] 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.

[18] H. Vinet, P. Herrera, and F. Pachet. The cuidado project. In *ISMIR*, Paris, France, 2002.

[19] H. Zhang, A. Kankanhalli, and S. Smoliar. Automatic partitioning of full-motion video. *ACM Multimedia System*, 1(1):10–28, 1993.