Signal-based Music Structure Discovery for Music Audio Summary Generation

Geoffroy Peeters and Xavier Rodet

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

Abstract

Deriving directly music structure from signal analysis, without going to symbolic information, is a new subject of interest. Two approaches are studied here in order to derive the structure: - the "sequence" approach, which considers the audio signal as a repetition of sequences of events – the "state" approach, which considers the audio signal as a succession of "states". Both approaches are derived from dynamic features observations extracted from the audio signal. The obtained structures are then used for the creation of audio summary.

1 Introduction

Automatic music structure discovery from signal analysis has become a major field of interest in the field of digital media (digital music) content analysis. Since a symbolic information is most of the time unavailable, and therefore deriving the structure from score or pitch (Walmsley, Godsill et al. 1999) (Dannenberg 2002) is not possible, people started thinking of deriving directly the structure from the audio signal. Among the applications of music structure discovery are: - browsing music catalogs/items by content (browsing by verse/ chorus/ solo), - quick pre-listening (automatic audio summary generation) - musicology (analysis of performances). 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. Music structure discovery from signal analysis methods are based on a search for repetitions of motives or melodies. The repetitions are detected by 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 flow of observations into segments. Various distances can be used in order to measure the similarity: Euclidean distance, cosine distance, …

In order to visualize the structural information of a piece of music, (Foote 1999) proposed to combine the similarity between each pair of times into a matrix, the so-called *similarity matrix*. If we note $s(t_i, t_j)$ the similarity between the observations at two instants t_i and t_j, the similarity of the feature vectors over the whole piece of music is defined as a similarity matrix $\underline{\underline{S}} = \left| s(t_i, t_j) \right| \quad i, j = 1,...I$. Since the distance is symmetric, the similarity matrix is also symmetric.

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 specific segment of music ranging from times t_i to t_{i+I} (sequence of events at time $t_i, t_{i+1}, t_{i+2}, \ldots$) is repeated later in the music from t_j to t_{j+J} (sequence of events at time $t_j, t_{j+1}, t_{j+2}, \ldots$), the succession of feature vectors in $[t_i, t_{i+I}]$ is supposed to be identical (close to) the ones in $[t_j, t_{j+J}]$. This is represented visually by a *lower (upper) diagonal* in the similarity matrix. The lag between the repetition (starting at t_i) and the original (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* $\underline{\underline{L}}$:

$$\underline{\underline{L}}(t_i, lag_{i,j}) = \underline{\underline{S}}(t_i, t_i - t_j) .$$

The *diagonal-sequences* in the similarity-matrix become *line-sequences* in the lag-matrix.

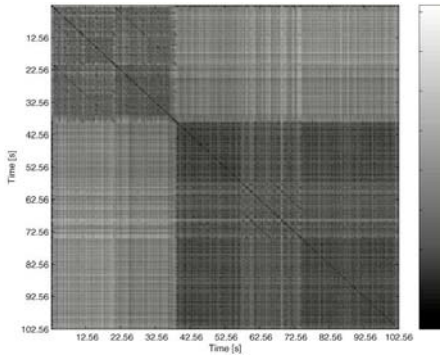

Figure 1 Similarity matrix using MFCC features on the title "Natural Blues" by "Moby"

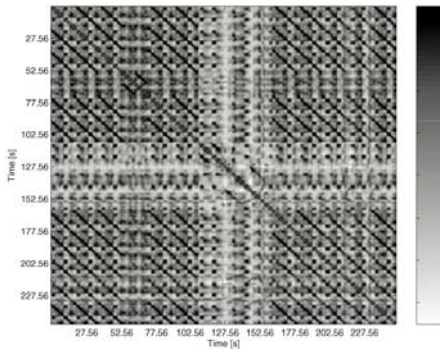

Figure 2 Similarity matrix using Dynamic features with short duration modeling on the title "Natural Blues" by "Moby"

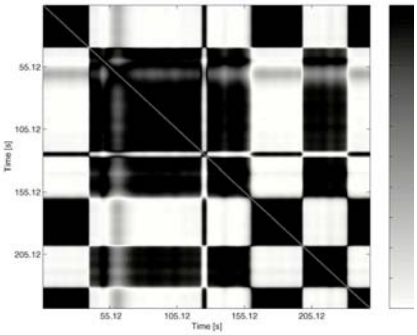

Figure 3 Similarity matrix using Dynamic features with long duration on the title "Natural Blues" by "Moby"

2 Dynamic audio features

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

- Mel Frequency Cepstral Coefficients (MFCCs) (Foote 1999) (Logan and Chu 2000) (Aucouturier and Sandler 2002)
- Mean and standard deviation of MFCCs (VanSteelant, DeBaets et al. 2002)
- Chromagram (harmonic content of the spectrum) (Bartsch and Wakefield)
- Scalar features (such as the spectral centroid, spectral rolloff, spectral flux, zero-crossing rate, …) (Tzanetakis and Cook 1999) (VanSteelant, DeBaets et al. 2002)

Each of the previously mentioned signal features (MFCC, chromagram, ...) represents a specific description (description of the spectral shape, of the harmonic content, ...) of the signal at (around) a given time. For this reason, we call them ``*static*'' features. The succession of this feature along time (succession of MFCC, chromagram along time) gives the evolution along time. ``*Dynamic*'' features aims at representing directly the evolution of the features along time. This evolution is modeled with a Short Time Fourier Transform applied to the values of the feature along time: 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, the same process is applied to each dimension.

The best results were obtained when 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 Figure 4.

In the case of dynamic features, a specific combination of several frequencies *n* in several speeds of variation ω can be chosen for a specific application. The feature vector $\underline{f}(t)$ contains then only this features. 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 $\rightarrow$ sequence approach,
- long duration of the model $\rightarrow$ 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).

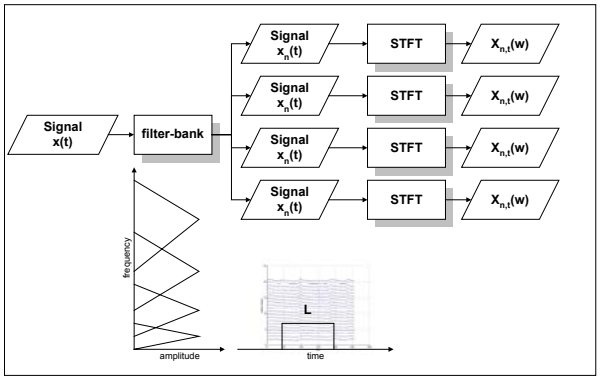

Figure 4 Dynamic features extraction from signal. From left to right: signal, filter bank, output of filter bank, output of each of filter, STFT of the outputs signal

In Figure 1, 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 Figure 2, 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 Figure 1, 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 Figure 3 the similarity matrix using dynamic features with a long duration (L=10.54s) is represented for the whole title duration (252 s). It 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]*.

3 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.

3.1 Related works

Foote showed in (Foote 1999) that a similarity matrix applied to well-chosen features (MFCC in (Foote 1999)) 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 the determination of the *direct location of the key sequence* of a piece of music used then as the audio summary (Bartsch and Wakefield) (Cooper and Foote 2002) or can be used for *discovering the underlying structure* of a piece of music. In order to do that, (Aucouturier and Sandler 2002) propose a method combining Gaussian distribution filter or the ``Hough Transform" for diagonal detection and pattern matching techniques for structure derivation.

3.2 Proposed approach

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)
2. from the set of lines we then form a set of *segments* (a segment is defined here as a set of continuous times).
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).

The global flowchart of the proposed algorithm for sequence representation is represented in Figure 5.

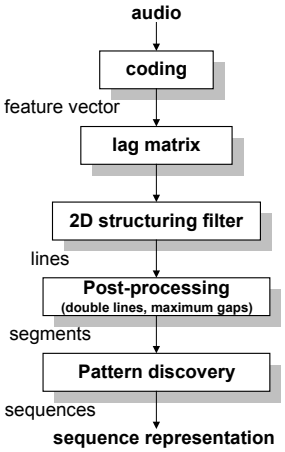

Figure 5 Sequence presentation flowchart

3.3 Search for diagonals/lines 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 (Aucouturier and Sandler 2002) 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 usually used neighboring values *[y-lagy, y+lagy]* to decide on keeping the value *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

The 2D structuring filter method we propose for vertical lines detection (see Figure 6) 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

The first threshold, *t1*, allows one to get rid off the low values in the similarity matrix. The second threshold, *t2*, defines the number of values before and after y which must be above *t1*. This threshold is proportional to the size of the considered interval: *t2=k*(2*lagy+1)*, where k range 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 had the condition that the main contribution to the counter must come from the main column *x*.

The results of the application of the proposed 2D structuring filter is represented on Figure 7 for the title "Love me do" by The Beatles

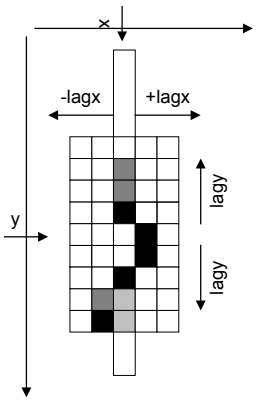

Figure 6 Two dimensional structuring filter for vertical lines detection

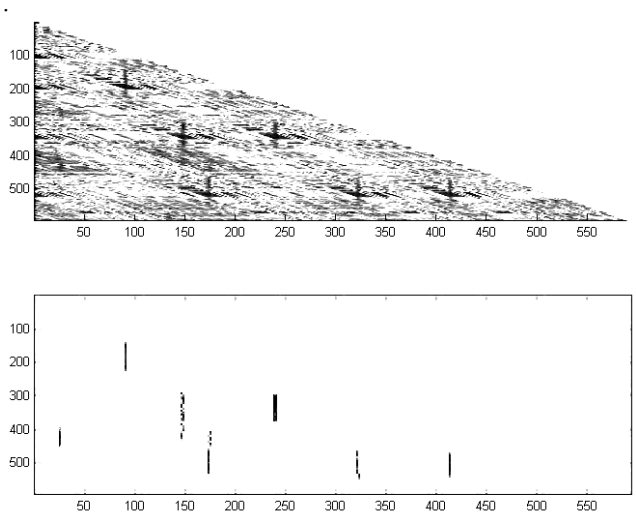

Figure 7 [Upper part] Lag-matrix for the title "Love me do" by "The Beatles" [Lower part] 2D structuring filter applied to the lag matrix

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, doubled lines detection is possible. 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.

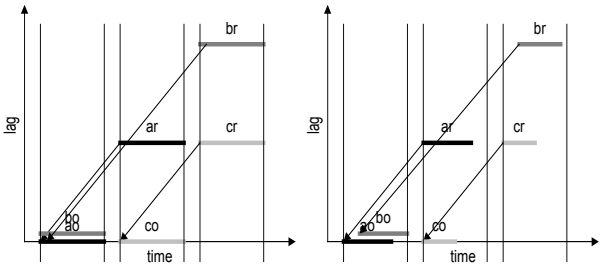

Figure 8 Sequences connections in the lag-matrix

3.4 From diagonals/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 gap tolerated inside a segment. If the observed gap is larger, then the line is divided into two separated segments. We also define the minimum accepted length for a segment.

A sequence *i* is now define by its start time *s(i)*, its end time *e(i)* and - since segment the detected lines are only the repetitions of some original segments - its lag from the original sequence *lag(i)*.

3.5 From segments to sequence: interpreting the detected segments

The connection between the various segments (finding which segment is a repetition of which other one, finding which segment can be considered as the reference for the sequence) is not an easy task.

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

Figure 8, 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*. And logically since *ao* and *bo* are the same and *ar* and *ao* are the same, there is a segment *c* which original is *co* and which share the same period of time as *ar*. However what we observe in practice is closer to the right part of

Figure 8: *bo* and *ao* only share a portion of time (which one is the reference then ?); the sequence *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 usual transition rule: *cr->co->ar->bo*.

Proposed algorithm

In order to perform the connections 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 duration*[1]*, 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)
- iO the original of a segment already present in the container.

init define S=minimum shared time
init add first jO and jR to the container
_while there is non-processed segment j
__take a new j (original jO of length jOL)
__if new segment jO shares time with a iO in the container
___for each of these i,
define iOL the length of iO,
define jOL the length of jO,
define ijOL the shared time length of iO and jO,
define the ratio c1=ijOL/jOL,
define the ratio c2(i)=ijOL/iOL
____select the i with the smallest c1+c2
____if c1 > S | c2 > lag **then** repetition
_____if c1 > S & c2 < lag **then** jO is in iO
_____if c1 < S & c2 < lag **then** iO is in jO
_____add jR to the sequence container with the same tag as iO
____else
_____add jO and jR to the container with a new tag
____end
__else
___add add jO and jR with a new tag
__end
_end

3.6 Results

In Figure 9, we illustrate our sequence representation method on the same signal as for Figure 7 (title "Love me do" from the artist "The Beatles"). The upper part represent the similarity matrix, the lower part represent 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" played by the harmonica. The only false detection was the sequence 1 at time 450.

1. [1] we've chosen a value of 70%

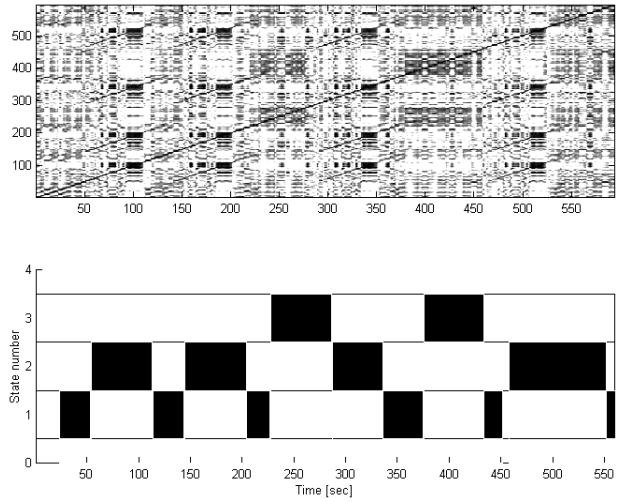

Figure 9 Sequence approach applied to the title "Love me do" by "The Beatles" [top] similarity matrix [bottom] sequences along time

4 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.

4.1 Related works

A study from Compaq (Logan and Chu 2000) uses the MFCC parameterization in order to create "key-phrases". 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 (Logan and Chu 2000), close to the method proposed by (Aucouturier and Sandler 2001), 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.

4.2 Proposed approach

The states we are looking for are 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 (Zhang, Kankanhalli et al. 1993). As for video, human segmentation and grouping performs better when listening (watching in video) to something for the second time (Deliege 1990). 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 previously mentally created templates.

In (Peeters, Laburthe et al. 2002) we proposed a multi-pass approach for music state representation, we review it here briefly. The global flowchart of this multi-pass approach is depicted Figure 10.

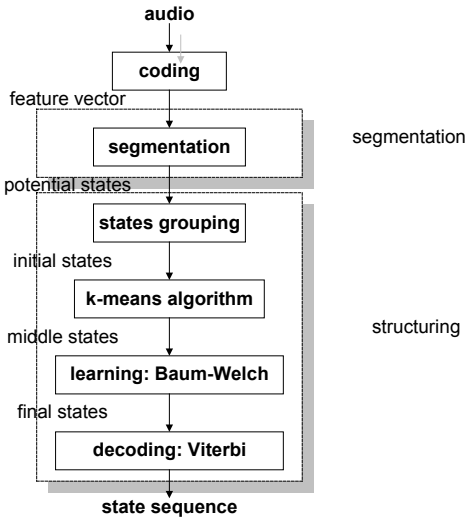

Figure 10 States representation flowchart

4.3 Proposed method: a multi-pass approach

First pass: The first pass of the algorithm performs a signal segmentation which 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 $\underline{\underline{S}}$ of $\underline{f}(t)$ (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. The mean values of $\underline{f}(t)$ inside each segment is used 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 stage:

1) Nearly identical "potential"' states are grouped.
2) The reduced set of states is used as initialization for a Fuzzy K-means (K-means with probabilistic belonging to class) algorithm (knowing the number of states and having a good initialization).
3) The output states of the Fuzzy K-means algorithm are used for the initialization of a hidden Markov model (HMM) learning (Rabiner 1989). Since no priori training on a labeled database is possible we are in the case of ergodic HMM. 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 as a HMM state sequence is obtained by *decoding* using Viterbi algorithm given the hidden Markov model and the signal feature vectors $\underline{f}(t)$.

4.4 Results

In Figure 11, we illustrate the proposed multi-pass approach for the title ``Smells Like Teen Spirit'' by `'Nirvana''. The upper part of the figure represents the similarity matrix while the lower part represents various detected states along time. Seven different states were found. State 1 represents the [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.

5 Audio summary construction

5.1 Related works

In the recent MPEG-7 standard (Multimedia Content Description Interface) (MPEG-7 2002) a set of meta-data has been normalized 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. 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.

5.2 Proposed approach

In our approach, the audio summary is generated by choosing specific excerpts of the signal derived from the So far, a sequence number or a state number has been assigned to each time frame of the audio signal. From this representation several possibilities can be taken in order to create an audio summary.

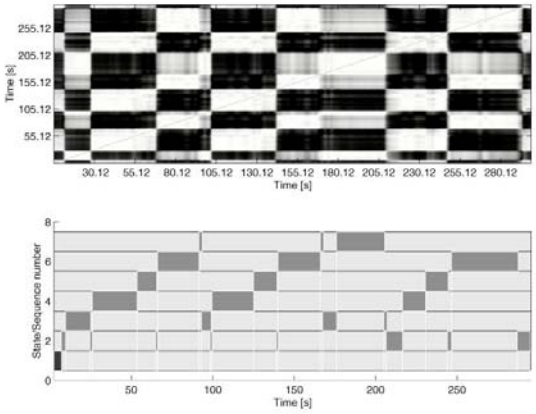

Figure 11 State approach applied to the title " Smells like teen spirit " by “Nirvana” [top] similarity matrix [bottom] state number along time.

Let us take as example the following structure: AA B A B C AA B. The generation of the audio summary from this sequence/state representation can be done in several ways: - [Each]: providing a unique audio example of each sequence/state (A, B, C) – [All]: reproducing the sequence/class successions by providing an audio example for each sequence/state apparition (A, B, A, B, C, A, B) – [Longest]: providing only an audio example of the most important sequence/state (in terms of global time extend or in term of number of occurrences of the sequence/state) (A) – [Transition]: in the case of state representation: providing audio example of class transitions (A -> B, B -> A, B -> C, C -> A).

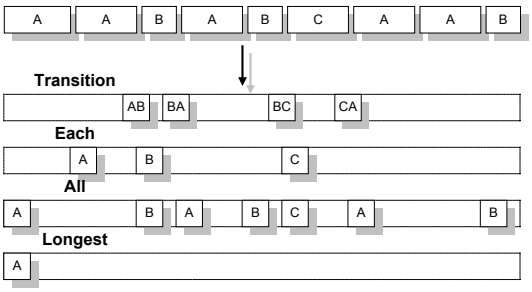

Figure 12 Various possibilities for Audio Summary construction from state representation

The audio summary is generated by taking short fragments of the segment/state's signal. The quality of the audio signal of the summary can be further improved by applying an overlap-add technique of the audio fragment. 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. The flowchart of the audio

summary construction of our algorithm is represented on Figure 13.

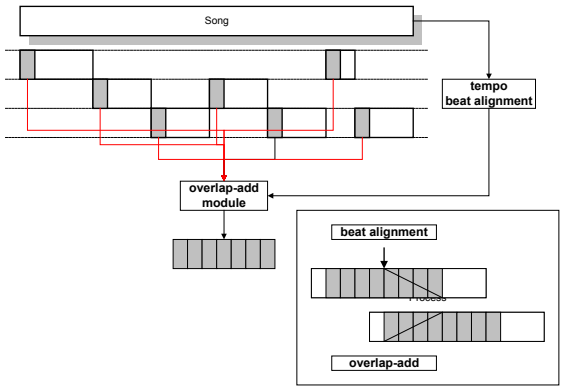

Figure 13 Audio summary construction from class structure representation; details of fragments alignment and overlap-add based on tempo detection/ beat alignment

While the choice between the various proposed methods should rely on user preferences and on time constraints on the summary duration, the "each" summary method was found to be informative (all states are represented in the summary), compact (only one occurrence of a state is represented) and reliable at the same time.

6 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 seems to allow deriving powerful information from the signal. 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 the generation of audio summaries generation from both sequential and state representations.

Combining both segment and state approach: Further work will concentrate on combining both sequence and state approaches. It is clear that both approach can help each other since the probability of observing a given sequence at a given time is not independent of the probability of observing a given state at the same time.

Acknowledgment

Part of this work was conducted in the context of the European I.S.T. project CUIDADO (Vinet, Herrera et al. 2002) (http:www.cuidado.mu).

Part of this work was conducted with Amaury La Burthe (amaury@csl.sony.fr) during its stay at Ircam.

References

Aucouturier, J.-J. and M. Sandler (2001). Segmentation of Musical Signals Using Hidden Markov Models. AES 110th Convention, Amsterdam, The Netherlands.

Aucouturier, J.-J. and M. Sandler (2002). Finding repeating patterns in acoustic musical signals: applications for audio thumbnailing. AES 22nd International Conference on Virtual, Synthetic and Entertainment Audio, Espoo, Finland.

Bartsch, M. and G. Wakefield To catch a chorus: Using chroma-based representations for audio thumbnailing. WASPAA.

Cooper, M. and J. Foote (2002). Automatic Music Summarization via Similarity Analysis. ISMIR, Paris, France.

Dannenberg, R. (2002). Pattern Discovery Techniques for Music Audio. ISMIR, Paris.

Deliege, I. (1990). A perceptual approach to contemporary musical forms. Music and the cognitive sciences. N. Osborne, Harwood Academic publishers. **4:** 213-230.

Foote, J. (1999). Visualizing Music and Audio using Self-Similarity. ACM Multimedia, Orlando, Florida, USA.

Logan, B. and S. Chu (2000). Music Summarization using Key Phrases. ICASSP, Istanbul, Turkey.

MPEG-7 (2002). Information Technology - Multimedia Content Description Interface - Part 5: Multimedia Description Scheme. ISO/IEC JTC 1/SC 29. ISO/IEC FDIS 15938-5:2002.

Peeters, G., A. Laburthe, et al. (2002). Toward Automatic Music Audio Summary Generation from Signal Analysis. ISMIR, Paris, France.

Rabiner, L. (1989). "A Tutorial on Hidden Markov Model and Selected Applications in Speech." Proccedings of the IEEE **77**(2): 257-285.

Tzanetakis, G. and P. Cook (1999). Multifeature Audio Segmentation for Browsing and Annotation. WASPAA, New Paltz, New York, USA.

VanSteelant, D., B. DeBaets, et al. (2002). Discovering Structure and Repetition in Musical Audio. Eurofuse, Varanna, Italy.

Vinet, H., P. Herrera, et al. (2002). The CUIDADO Project. ISMIR, Paris, France.

Walmsley, P., S. Godsill, et al. (1999). Bayesian Graphical Models for Polyphonic Pitch Tracking. Diderot Forum, Vienna, Austria.

Zhang, H., A. Kankanhalli, et al. (1993). "Automatic Partitioning of Full-motion Video." ACM Multimedia System **1**(1): 10-28.