Automatic Extraction of Musical Structure Using Pitch Class Distribution Features

Bee Suan Ong[1], Emilia Gómez[1], Sebastian Streich[1]

[1] Music Technology Group
Universitat Pompeu Fabra
Ocata 1-3, 08003 Barcelona, Spain
{beesuan, egomez, sstreich}@iua.upf.edu

Abstract. This paper compares the efficiency of different sets of tonal descriptors in music structural discovery. Herein, we analyze the use of three different pitch-class distribution features, i.e. Constant-Q Profile, Pitch Class Profile (PCP) and Harmonic Pitch Class Profile (HPCP), to perform structural analysis of a piece of music audio. We hypothesize that proper segmentation serves as an important basis to obtain music structure analyses of better quality. Thus, we compare the segmentation results produced by each feature to examine its efficiency. A database of 56 audio files (songs by The Beatles) is used for evaluation. In addition, we also show the validity of the descriptors in our structural description system by comparing its segmentation results with a present approach by Chai [1] using the same database. The experimental results show that the HPCP performs best yielding an average of 82% of accuracy in identifying structural boundaries in music audio signals.

Keywords: automatic music structural analysis, automatic segmentation, pitch class distribution features.

1 Introduction

The generation of high-level metadata to describe audio content contributes to more efficient and better retrieval of digital music. Understanding musical form through analyzing structural transitions can be a primary step going towards generating useful descriptions from music audio data. Melody and harmony are important aspects in music perception and understanding. A substantial part of information about these two elements is contained in the pitch-related perspective of music. Hence, we employ descriptors, which subsume tonal information, to discover musical structure from audio signals.

These descriptors capture most of the tonal information that is present in a song without requiring specific pitch detection or source separation. Here, we compare the efficiency of different low-level tonal descriptors, related to pitch class distributions, which could be useful for automatic structural analysis and discovery. So far, several studies in this area have made comparisons between various description aspects (i.e. tonal-related versus timbre-related features) [1] [2] for application in music structural analysis. However, there still exists no comparison of the performance on segment extraction of different approaches for computing pitch class distribution features. Thus, we evaluate the suitability of these low-level descriptors by examining the segmentation results obtained from our automatic structural analysis system.

Our structural description system presented in this paper is based on Goto's method [3] for detecting chorus sections in music. We have further improved upon the methodology towards accomplishing a more complete music structural description through providing different labelling, together with beginning and ending time information, to mark (dis)similar sections that appear in the music signal (i.e. verse, chorus, bridge, etc.). There are three main steps in our system: (1) feature extraction; (2) structural analysis; (3) repetitive segments compilation for structural description. Figure 1 illustrates the different processing stages in our structural description system.

This paper is organized as follows. Section 2 presents the process of feature extraction from an audio signal. Section 3 gives a detailed description of our system. Section 4 describes the compilation process of the obtained repetitive segments. The evaluation is presented in Section 5. The last section concludes the paper with future research plans.

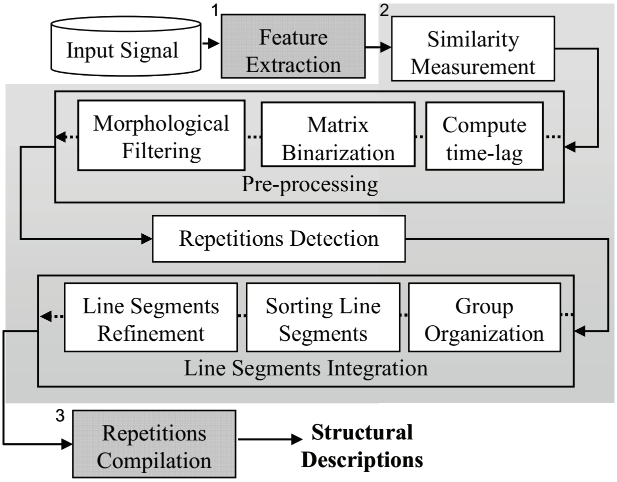

Fig. 1. Overview framework of the automatic structural descriptions system.

2 Feature Extraction

As a first step, our system requires the short-term description of the input audio signal. We segment the input signal into overlapped frames (4096-samples window length) with the hop size of 512 samples. It is then followed by extracting pitch class distribution features of each of these frames. Here, we use one of three different approaches for extracting low-level tonal features from input signals. The general block diagram for computing pitch class distribution features is shown in Figure 2.

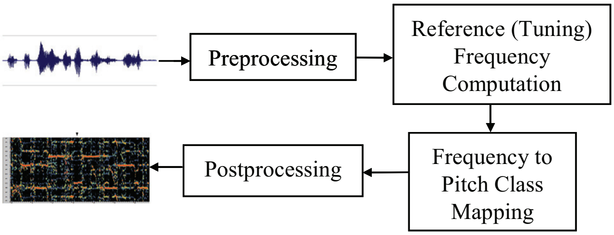
Preprocessing
Reference (Tuning) Frequency Computation
Frequency to Pitch Class Mapping
Postprocessing

Fig. 2. General diagram for computing pitch class distribution features.

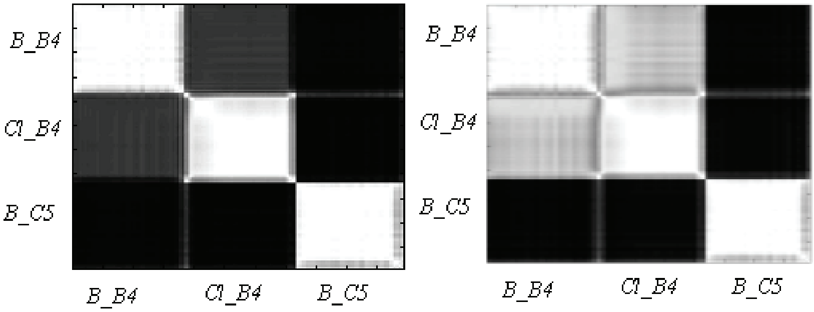

Fig. 3. Self-similarity matrices of three notes, which include B4 played by the bassoon (*B_B4*), B4 by the clarinet (*Cl_B4*), and C5 by the bassoon (*B_C5)*, using different Constant-Q feature vectors: (right) Constant-Q extracted directly from 5 octaves of musical notes (left) Constant-Q extracted from 5 octaves of musical notes and mapped into 1 octave.

Our approach in discovering music structure requires audio features with high sensitivity towards tonal similarities and independence with respect to timbre and instruments played to reveal repeated patterns in music. Thus, different from Lu et al.'s proposed features in music structural analysis [4], we use octave mapping for all our compared features. This is due to the reason that through octave mapping, the CQT features are more sensitive to tonal similarities compared to the non-octave mapping of the features. Figure 3 illustrates self-similarity matrices of three notes based on cosine distances among three notes, which includes B4 played by the bassoon (*B_B4*), B4 by the clarinet (*Cl_B4*), and C5 by the bassoon. The similarity plots are normalized to [0,1], and the brighter points represent high similarity. From the similarity plots, it is noted that the similarity score between B4 played by the bassoon (*B_B4*) and B4 played by the clarinet (*Cl_B4*) is higher for the octave-mapped constant Q transformed features than the non-octave-mapped features. As the act of octave mapping produces audio descriptors with the feature properties very much fulfilling the requirement of our approach, we adopt the octave mapping procedure for all our used features. We focus here in describing the main differences between the three different approaches: Constant-Q profiles (CQP), based on [5], Pitch Class Profiles (PCP), as proposed in [6] and the Harmonic Pitch Class Profiles (HPCP), explained in [7].

2.1 Preprocessing

CQP use the constant-Q transform as a preprocessing step before mapping frequencies to pitch class values, while PCP and HPCP use the DFT. The preprocessing step also includes a frequency filtering after DFT, so that only a frequency band between 100 and 5000 Hz is used. HPCP finally includes a peak detection procedure, so that only the local maxima of the spectrum are considered.

A reference frequency computation procedure is used before computing HPCP, in order to estimate the deviation with respect to 440 Hz of the frequency used to tune the piece. This is done by analyzing the deviation of the peaks frequencies with respect to the perfect tuning. PCP and CQP use a fixed frequency grid with a 440 Hz reference.

2.2 Frequency to pitch class mapping

Once the reference frequency is known and the signal is converted into a spectrogram by means of DFT or constant-Q analysis, there is a procedure for determining the pitch class values from frequency values. In CQPs, the weight of each frequency to its corresponding pitch class is given by the spectral amplitude whereas the PCPs use the squared value. The HPCP introduces a weighting scheme using a cosine function (described in [7]), and considers the presence of harmonic frequencies, taking into account a total of 8 harmonics for each frequency. In the three compared approaches, the interval resolution is set to one-third of a semitone, so that the size of the pitch class distribution vectors is equal to 36.

2.3 Post-processing

Finally, the features are normalized frame-by-frame dividing through the maximum value to eliminate dependency on global loudness.

3 Structural Analysis

3.1 Similarity Measurement

As a first step towards structural analysis, the system computes the average of each 10 extracted feature frames (as described in Section 2) to represent the tonal distributions of the original input signal of every 116 ms, approximately. This is to prevent the system from having high computational load by processing the complete set of feature vectors. The amount of possible reduction is limited by the loss of detail introduced by the averaging operation. With the computed mean feature values, we measure the (dis)similarity distance between each 116 ms of the tonal descriptors using the cosine distance measure [8].

3.2 Pre-processing

To ease the process of identifying repetitive segments in music, we compute the time-lag matrix of the similarity representation, *SD,* computed from the previous processing, by orientating the diagonal of the computed similarity matrix towards the vertical axis. The rotated time-lag matrix, *L(l,t)* between chroma vectors *v(t)* and *v(t-l)* is defined as

$$L(l,t) = SD(v_t, v_{t-l}) \tag{1}$$

For detecting repetitions or vertical lines in the time-lag matrix, we only want to consider line segments that show a sufficiently high degree of similarity. For dealing with broad categories of audio input signals, we perform a binarization process on the time lag matrix based on an adaptive threshold. The implementation of the binarization procedure is based on an iterative procedure. For initialization, our adaptive threshold holds a default value of *Th*. We first binarize the similarity values in the time-lag matrix by setting all values smaller than *Th* to 0 and the rest to 1. Then we computer a *P* value from the binarized matrix to evaluate the sufficiency of information it retains. The *P* value is defined as:

$$P = \frac{\text{total number of 1 in time-lag matrix}}{0.5 \times Area(\text{time-lag matrix})} \tag{2}$$

Based on the computed P value, we consider three cases as listed below:

(1) If $P > P_{max}$, increase the threshold, *Th,* by 0.01 and return to the beginning of the procedure;
(2) Else if $P < P_{min}$, reduce the threshold *Th,* by 0.01 and return to the beginning of the procedure;
(3) Else, quit the iterative process and output the binarized time-lag matrix

where P_{max} and P_{min} denote the empirical upper bound and lower bound of the *P* value. The last operation of the pre-processing section consists of applying the opening operation of a morphological filter [9] to the binarized time-lag matrix. The purpose of applying the opening operation is to remove line segments, which are too short to contain any significant repetition of music (see Figure 4).

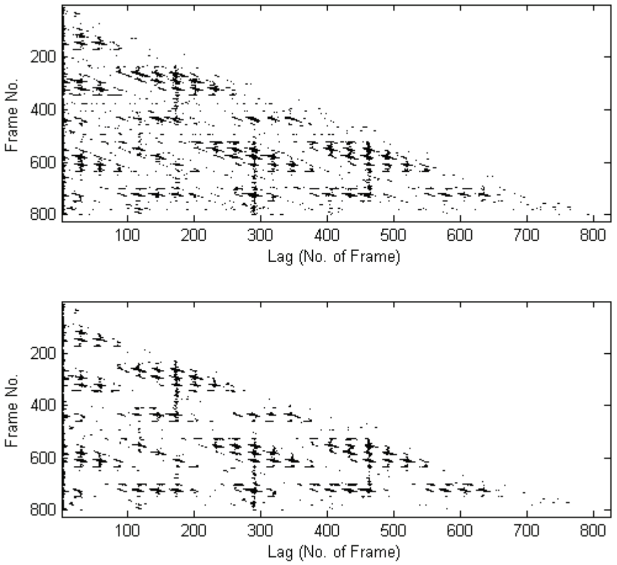

Fig. 4. General diagram for computing pitch class distribution features.

3.3 Repetition Detection

For detecting repetitive segments in music, we adapted Goto's methods from [3]. The main goal of this process is to detect repetitive segments for structure discovery. This process requires the output data, $L_p(l,t)$, from the morphological filtering process as an input signal. As mentioned earlier, vertical line segments in the time-lag matrix represent the occurrence of repetition in music. Thus, for finding the possibility of each lag for containing a line segment, $P_r(l,t)$, we sum up the corresponding column of the time-lag matrix and normalize it with the total number of elements in the lag. The calculation of the posibility of containing line segments, $P_r(l,t)$, of each lag is defined as:

$$P_r(l,t) = \int_l^t \frac{L_p(l,t)}{t-l} dt \tag{3}$$

For finding line segments, we select all peaks in $P_r(l,t)$ and store their lag information in a descending sorted list as $l_{PeakSort.}$ We then evaluate the occurrence of line segments in $L_p(l,t)$ alternately for each element in $l_{PeakSort}$. We comptue $L_p(l_{PeakSort.}\ t)$ for each $l_{PeakSort}$ and search for the occurrence of vertical line segments. Here, we assume that short repetitions (which hold less than 4 seconds) do not carry much significant musical information. Thus, for detecting repetition segments in music, we only consider those line segments with duration longer than 4 seconds. For each detected line segment, we store the beginning and ending time of the original segment together with the repeated segment based on the information from $l_{PeakSort.}$

3.4 Integrating the repeated sections

In this section, we organize the detected repetition pairs obtained from the previous steps into groups. Apparently, line segments that share a common line segment are the repetitions of one another as shown in Figure 5. Thus, if these segments are to be labeled, they should be given the same labeling. Based on this observation, we integrate those line segments, which share a common line, into one group with the same label. From this, we generate a set of repetition groups with different labels marking the different repetition segments that appear in the music. That is

$$Group_{repetitions} = \left\{Group_1, Group_2, ..., Group_n\right\} \tag{4}$$

where *n* is the number of repetition groups. In each repetition group, we sort repeated line segments in an ascending order based on their time information:

$$Group_A = \left\{[Tbegin_1, Tend_1];[Tbegin_2, Tend_2];...;[Tbegin_m, Tend_m]\right\} \tag{5}$$

where *Tbegin* and *Tend* denote the beginning time and ending time of the repetitive segments whereas *m* is the number of repetitive segments in $Group_{\mathrm{A}}$.

For the refinement of line segments, we select the first line segment of each group in $Group_n$, and correlate it with the pre-processed features, *v(n).* This is for the purpose of recovering undetected repetitions that we have missed in the previous detection process. We compute the distance measure, *E(n)*, for the selected line segment and a sliding window of the same length on the pre-processed features, *v(n).* The distance measure, *E(n)*, is defined as

$$E(n) = \sqrt{\frac{\text{å}\ \text{å}\ (Compared_{segment} - v(n)_{len\_compared})^2}{len\_compared^2}} \tag{6}$$

where $Compared_{segment}$ denotes the compared segment features and *len_compared,* its length. $v(n)_{len\_compared}$ represents the nth pre-processed feature sequence with the length *len_compared.*

To detect significant repetition appearing in the music, we use an adaptive threshold based on the computed distances. Excluding the distance of the compared segment to itself (which is always zero), we select the lowest occurring distance value. To obtain the adaptive threshold, we add a fixed tolerance margin to this value. Then, all local minima falling below the threshold are considered to be relevant to the occurrence of repetition. We sort the considered local minima based on the distance measure in descending order. With the length of the compared segment, we estimate and store the corresponding beginning and ending time for each considered local minimum and form a set of candidate segments. Here, we disregard those candidate segments that overlap with any of the line segments in the group that hold the same label as the compared segment based on the assumption that repetitions of a segment do not overlap with each other. The remaining candidates are labeled and included in the correct group as an omitted repetition from the earlier detection process. Finally, we reorganize line segments in the group with an ascending order based on their time information. This procedure is similar to the earlier sorting processes of the line segments of each group in *Group*$_n$.

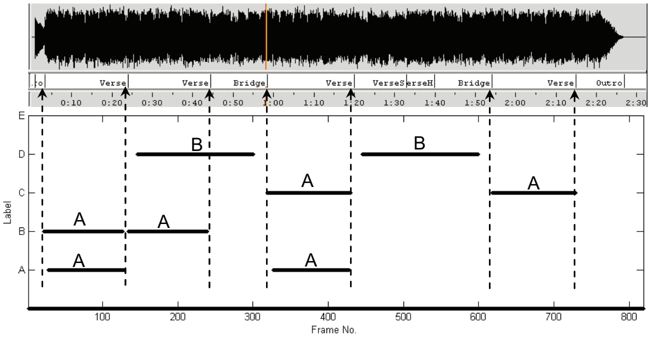

Fig. 5. Detected repetitions correspond to the ground truth annotation of *A Hard Day's Night.*

4 Repetitive Segments Compilation

For generating the music structural description, we select the three most repetitive groups, *Group*$_n$*,* (i.e. with the highest number of elements). We compile the repetitive segments by lining up all the line segments of these repetitive groups according to their labels as shown in Figure 5. If there exists an overlap between two particular labels (e.g. A and B as shown in Figure 6), all the overlapped sections of these two labels will be given a new label (e.g. C), whereas the non-overlapped sections will be given another label (e.g. D). Unlabelled sections between all the labeled segments (e.g. E and F) will be given a new label respectively as a new repetition group by itself. We then select one line segment of each label and perform another repetition detection procedure by correlating it with the pre-processed features, *v(n)*, as described in Section 3.4, this time with the goal of finding all the corresponding repetitions that appear in the music signal. Finally, the repetition detection process terminates when we checked all labels obtained from the previous operation. With the labeled line segments, we combine all the repeated labels, with the length of less than 25 sec *to* become a single label. This is based on the assumption that structural sections in music (i.e. intro, verse, chorus, etc.) are less than 25 sec in length. Figure 7 shows an example of the integration process of repeated labels.

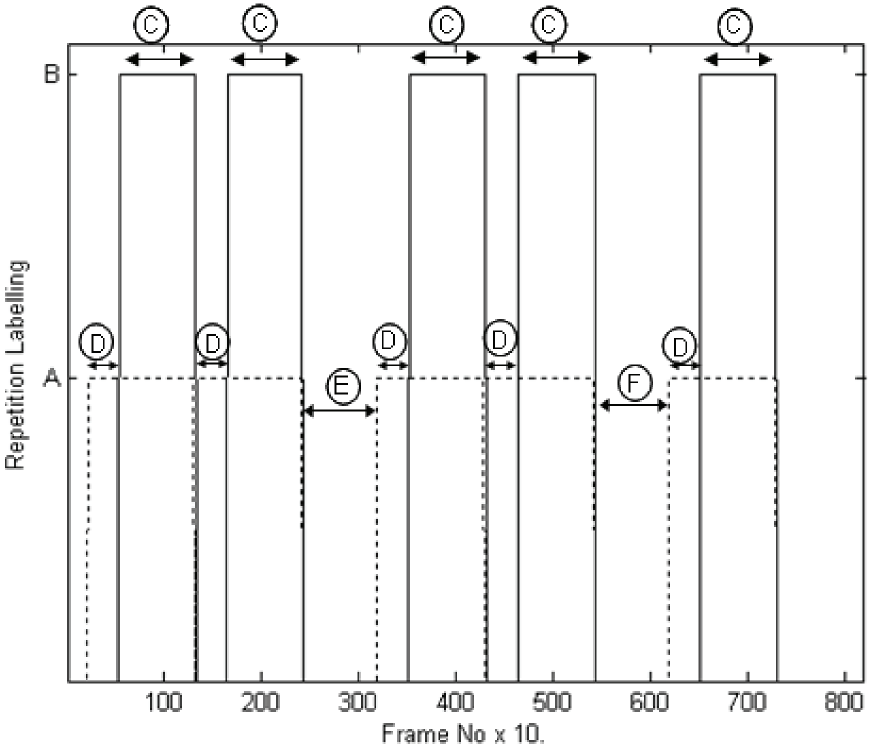

Fig. 6. Repetitive segments compilation process with generated new labels.

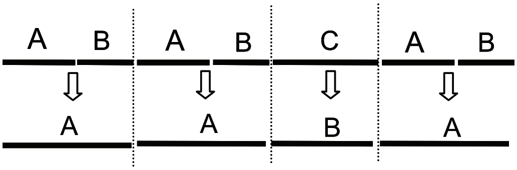

Fig. 7. Labelling integration procedure.

5 Results

5.1 Data Sets

In our experiment, we use 2 test sets. The first test set consists of 56 songs from The Beatles 70s' albums, whereas the second test set uses the same audio database as in [1], 26 songs by The Beatles from the years 1962-1966. Each song is sampled at 44.1 KHz, 16-bit mono. For evaluation purposes, we have generated a ground truth by manually labeling all the sections (i.e. intro, verse, chorus, bridge, outro, etc.) for all songs in both datasets according to the information provided by Allan W. Pollack's "Note On" Series website on song analyses of Beatles' twelve recording project[1]. A music composer supervised the labeling process and results.

5.2 Evaluation Measures

To quantitatively evaluate the segmentation performance, we calculate the precision and recall of the generated structural description. We compare the obtained segment boundaries for each of the three descriptors with manually labeled ground truth results. Recall and precision are computed for various degrees of tolerance deviation (between 0.3 sec and 3.6 sec) in order to obtain a more complete picture about the accuracy and reliability of the segmentation results.

5.3 Experimental Results

Figure 8 and Figure 9 show the evolution of precision and recall scores with respect to the tolerance deviation for the different pitch class distribution descriptors using test set I. From both figures, we have observed a significantly higher performance of HPCP compared to PCP and CQP. With the tolerance deviation of 3.6 sec, HPCP has achieved accuracy higher than 70% and a reliability of 83%. From our segmentation results, it shows that HPCP has outperformed the other tonal descriptors by as much as 10% in both precision and recall scores with 3.6 sec of tolerance deviation. T-test analysis concludes that the differences are statistically significant beyond the 99% confidence level with the p-values<0.01. For the case of PCP and CQP, there is no statistically significant difference in their performance on our used test set.

Figure 10 illustrates both precision and recall scores of the HPCP using test set II with respect to the tolerance deviation. From the segmentation results, we can see that HPCP has achieved a slightly better performance with its precision and recall rate of 82% and 84% respectively compared to results documented in [1]. Overall, we have reached an F-measure of nearly 83%. However it should be noted that the generality of our test sets is quite limited. So far, we have not yet tested our approach on different music genres (e.g. heavy metal, techno, or jazz).

[1] *The Twelve Recording Projects of the Beatles* webpage: http://www.icce.rug.nl/~soundscapes/DATABASES/AWP/awp-beatles_ projects. html

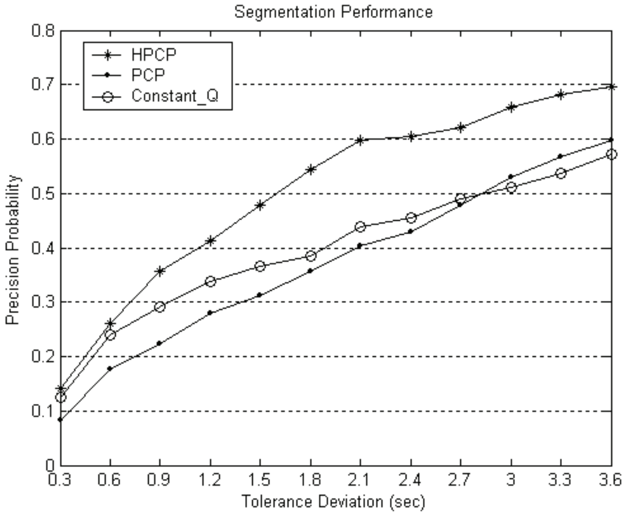

Fig. 8. Evolution of precision score with respect to the tolerance deviation (sec) for the different pitch class distribution features using test set I.

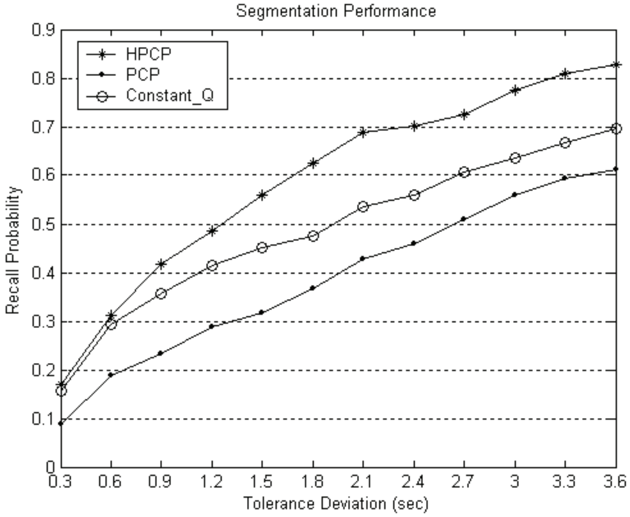

Fig. 9. Evolution of recall score with respect to the tolerance deviation (sec) for the different pitch class distribution features using test set I.

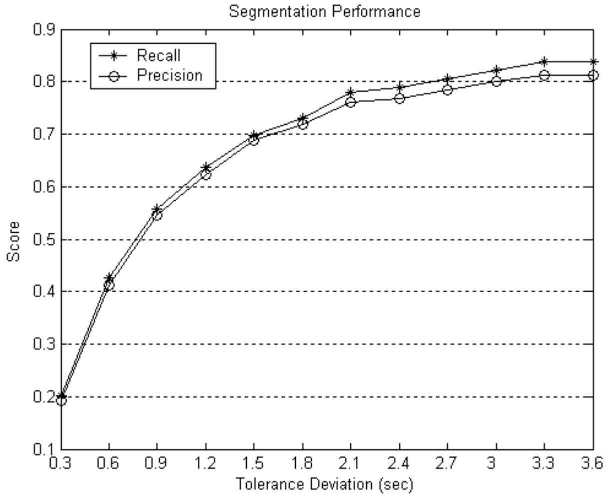

Fig. 10. Evolution of recall and precision rates of HPCP with respect to the tolerance deviation (sec) for the different pitch class distribution features using test set II.

6 Conclusion

In this paper, we have presented an objective comparison between different tonal descriptors for detecting structural changes in music audio. We have also shown the validity of the descriptors by comparing our segmentation results to those recently published by another researcher [1]. We have seen that the approach employed to compute pitch class distribution features has an influence on the performance of structural analysis. With our approach (HPCP), we have been able to achieve as high as 82% accuracy in identifying appropriate structural boundaries in music with a tolerance deviation of 3.6 sec. For ongoing research to further improve the segmentation performance, more attention will be given to the following factors:

- Making use of higher-level analysis techniques (e.g. beat detection or phrase detection) to achieve better segmentation truncation with lower tolerance deviation.
- Testing the performance using an annotated database comprising different music genres different from “60’s pop music” and containing different artists.

7 Acknowledgments

This research has been partially funded by the EU-FP6-IST-507142 project SIMAC (Semantic Interaction with Music Audio Contents; http://www.semanticaudio.org/) and EU-e-Content project HARMOS. The authors would like to thank members of

SIMAC and AUDIOCLAS projects at the Music Technology Group in the UPF for their useful comments and discussions.

References

1. Chai, W.: Segmentation and Summarization of Music. In: IEEE Signal Processing Magazine, Special Issue on Semantic Retrieval of Multimedia, to appear
2. Bartsch, M., and Wakefield, G.: Audio Thumbnailing of Popular Music Using Chroma-Based Representations. In IEEE Transactions on Multimedia, Vol. 7, No. 1 (2005) 96-104
3. Goto, M.: A Chorus-Section Detecting Method for Musical Audio Signals. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (2003) 437-440
4. Lu, L., Wang, M., Zhang, H-J.: Repeating Pattern Discovery and Structure Analysis from Acoustic Music Data. In MIR' 04, New York, USA (2004) 275-282
5. Brown, J. C.: Calculation of a Constant Q Spectral Transform. In: Journal of the Acoustical Society of America 89 (1991) 425-434
6. Fujishima, T.: Realtime Chord Recognition of Musical Sound: a System Using Common Lisp Music. In: Proceedings of the ICMC, Beijin, China (1999)
7. Gómez, E.: Tonal Description of Polyphonic Audio for Music Content Processing. In: INFORMS Journal on Computing, Special Cluster on Computation in Music, E. Chew Ed, Vol. 18, No. 3 (2006)
8. Foote, J.: Automatic Audio Segmentation Using a Measure of Audio Novelty. In: IEEE Int. Conf. Multimedia and Expo, Vol. I (2000) 452-255
9. Filonov, A., Gavrilko, D.Y., Yaminsky, I.V: Scanning Probe Microscopy Image Processing Software Users's Manual "FemtoScan". Version 4.8. Moscow: Advanced Technologies Center (2005).