AUDIO CHORD ESTIMATION USING CHROMA REDUCED SPECTROGRAM AND SELF-SIMILARITY

Nikolay Glazyrin
Ural Federal University, Ekaterinburg
nglazyrin@gmail.com

ABSTRACT

In this paper we describe a method of audio chord estimation than does not rely on any machine learning technique. We calculate a beat-synchronized spectrogram with high time and frequency resolution. The sequence of chroma vectors (CRP features based on constant-Q transform) obtained from spectrogram is smoothed using self-similarity matrix before the actual chord recognition. Binary chord templates with 3 harmonics are used. Two heuristics are applied to the resulting chord sequence to reduce the number of major-minor chord confusions and to remove single-beat chords. The method is evaluated on the *Isophonics* [7] and *RWC Popular Music* [5] datasets (318 tracks in total).

1. INTRODUCTION

Audio chord estimation is one of the most interesting tasks in music information retrieval. Representation of audio as a sequence of chords can be helpful for many other tasks, and itself can provide valuable information to the user. Chord recognition algorithms have been showing great progress during last years. 12 out of 18 algorithms that have participated the MIREX 2011 Audio Chord Description task have achieved chord weighted average overlap ratio greater than 0.70, whereas in 2012 all participants have overcome this barrier on the same dataset (which is named as MIREX in Audio Chord Estimation results in 2012).

The majority of participating algorithms use machine learning techniques to some extent, except one of the variants of the method by T. Cho and J. P. Bello [1], the method proposed by T. Rocher et al. [12] and the method by de Haas et al. [3]. But the usage of machine learning makes the algorithm dependent on training data. Despite the fact that the amount of available data is grownig last years, still the available data cover only a small subset of world music. Therefore, the development of an algorithm that could provide good quality of chord recognition without any training makes sense.

The algorithm presented here is mostly resembles the one submitted to MIREX Audio Chord Estimation 2012 (denoted by NG1), but also has some differences. A simple detector of no playing chord was implemented. One more heuristic was added to correct chords that only last for one interval between 2 successive beats. Binary templates with 3 harmonics are used instead of hand-crafted chord templates. Also all the parameters were tuned more precisely than in our previous submission.

Two versions of this algorithm (NG1 and NG2) have been submitted. NG2 differs from NG1 only in that it also contains templates for maj7, min7 and dominant 7th chords. It gives many confusions with simple major/minor chords though, so it is submitted just for comparison.

2. SYSTEM DESCRIPTION

2.1 Tuning

2.1.1 Tuning frequency estimation

A very simple algorithm is used to estimate tuning frequency of an audio record. It is similar to the one that was described by Y. Zhu et. al. in [13]. This algorithm is by no means the best one, but it has been chosen for the ease of implementation.

The whole record is split into T sequential fragments. Then the constant-Q transform is applied on each fragment $t_i, i = 1, 2, ..., T$ with the following parameters: 240 frequency components per octave (or 20 components per note), minimal component frequency equal to 220 Hz, span of 4 octaves. Due to chosen (rather big) minimal frequency this procedure is very fast for the whole file. On each fragment t_i we determine the position of maximal spectrum value $g(t_i)$. Then a histogram of values of this function is constructed. It has 960 bins. The histogram is then folded to 10 bins: j-th bin of the original histogram contributes to $(j \ mod \ 10)$-th bin of the collapsed histogram for $j = 0, 1, ..., 959$. Then the position of maximal value in the resulting histogram shows the deviation of the tuning frequency from the base 440 Hz frequency in range from -1/2 to +1/2 semitone (427.5 Hz to 452.9 Hz) with step of 1/20 semitone (maximum at 0th position corresponds to no deviation). Calculated deviation is used further to specify the tuning frequency for main constant-Q transform.

2.1.2 Beat positions estimation

To estimate the position of beats in a sound file, the *Beat-Root* library [4] was used. We have also experimented with beat trackers implemented as Vamp plugins by M.

Davies [2] and J. Oliveira [11], but the best results were obtained using *BeatRoot* (and beat tracker by M. Davies for the tracks on which *BeatRoot* fails to provide any result). The sequence of beat positions is then made T times more frequent by inserting $T - 1$ intermediate values evenly between each pair of successive beat positions. Therefore, if the source sequence has n elements, then the resulting sequence will have $T \cdot (n - 1) + 1$ elements. The value of $T = 8$ was chosen.

2.2 Spectrogram calculation

Stereo wave file is converted to mono at first. Then for each time position from the sequence of beats the constant-Q transform on the fragment that is centered at this position is performed. The transform has the following parameters: 36 frequency components per octave, 4 octaves span, minimal component frequency is 33 semitones below tuning frequency. In case of standard tuning frequency 440 Hz the minimal component has frequency 65.41 Hz (corresponds to C2). The whole frequency range in this case spans from 65.41 Hz to 987.77 Hz. Given these parameter values the spectrogram has 144 rows.

Then a median filter with window size w is applied to each row of the spectrogram (each row corresponds to a component frequency of constant-Q transform). Due to very frequent sequence of time positions we can effectively smooth the spectrogram while using only the values that belong to a short time interval. Here the value $w = 15$ is used, which corresponds to only 2 beats.

Then the spectrogram is simply decimated along the time axis: each 8th column is preserved, all the others are removed. High time resolution is redundant, because chords often change along with the beats. But now the decimated spectrogram also includes the information from the whole interval between two successive beats.

2.3 Chroma reduction

The process proposed by Müller in [10] is applied here. Each spectrogram value v is replaced with $log_{10}(50000 \cdot v + 1)$. Then a discrete cosine transform of size 144 is applied to each spectrogram column. The first 15 resulting coefficients are set to zero and the inverse DCT is performed. This value is small compared with the one used in [10], but it gives the best result in our case.

2.4 Smoothing using self-similarity matrix

This step is inspired by the works of T. Cho and J. P. Bello [1] and M. Mauch et al. [9]. Both these approaches look for the repetitive structures in the music to improve the resulting sequence of chords. But in the proposed method a self-similarity matrix is built for the sequence of spectrogram columns $\{p_i\}$, not feature vectors. It is also not required to have only diagonals.

Euclidean distance is used as a measure of similarity, so the self-similarity matrix has zeroes on the main diagonal. It is normalized, so that $0 \leq s_{ij} \leq 1$ for any i, j. Then for each row we preserve only $M \cdot n$ minimal values and set all the others to 1 (here $0 \leq M \leq 1$). The value $M = 0.08$ was chosen.

Then the sequence of spectrogram columns is recalculated using the values from this matrix:

$$\hat{p}_i = \frac{\sum\limits_{j=1}^{n} (1 - s_{ij}) \cdot p_j}{\sum\limits_{j=1}^{n} (1 - s_{ij})}$$

2.5 Calculation of chroma vectors

This is the last step before the actual chord estimation. At first, spectrogram columns are fold from 4 octaves to 1 octave. It is done by summing the rows with indexes $j, j + 36, j + 72, j + 108$ for each $j = 0, ..., 35$ into one row. This results in a sequence of 36-dimensional chroma vectors $\{\hat{p}_i\}_{i=1}^{n}$. Each vector is then projected to 12 dimensions:

$$q_i[j] = \sum_{h=-1}^{1} \hat{p}_i[3j-h] \cdot d^{|h|}, \quad i = 0, ..., n, \quad j = 0, ..., 11$$

The parameter d adjusts the contribution of spectral components that do not correspond to real notes. We choose $d = 0.6$. For the computation of $q_i[0]$ the components $\hat{p}_i[35]$ is substituted.

2.6 Chord estimation

Chord templates are used to determine the corresponding chord for each chroma vector and the distance from a template to a chroma vector is the Euclidean distance.

The templates for major and minor chords are used in the proposed method. We used binary templates with 3 harmonics, where for i-th harmonic an amplitude of 0.6^{i-1} is added. Before distance calculation both template vector and chroma vector are normalized to have unit length in the Euclidean space.

For each spectrogram column the following coefficients are calculated to detect the absence of the chord on a given frame. K_{tonal} is a ratio of the sum of spectral components corresponding to the note frequencies (given the tuning frequency) to the sum of all components for a given spectrogram column. K_{max} is a ratio of maximum component to the sum of all components. The columns where $K_{tonal} \cdot K_{max} < 0.0011$ were treated as corresponding to no chord playing. The limit value of 0.0011 was found empirically. Also the sections of track before the first detected beat and after the last detected beat plus the beat length were marked as having no chord playing.

2.7 Additional correction

This step is introduced to decrease the number of the confusions between chords that have the same root note but different type (e.g. major and minor chords). Such chords very seldom occur one after another. If the sequence of chords has subsequences with this property, these subsequences are suspicious. They are corrected so that each subsequence contains only the chords of one type. It is

Collection	AOR	WAOR	Segmentation
Isophonics dataset	0.7824	0.7686	0.7836
2 datasets together	0.7640	0.7516	0.7907

Table 1. Recognition quality.

done by summing feature vectors from each subsequence into a single vector, which is then matched against chord templates and assigned to the corresponding subsequence.

Another heuristic was introduced to fix chord sequences like (A, B, C), where 3 successive beats are marked with 3 different chords. Each similar sequence is replaced with the one of (A, A, C), (A, C, C), (B, B, C), (A, B, B) for which the sum the of distances from successive feature vectors to corresponding chord labels is minimal. The sequences like (A, B, A) are replaced with (A, A, A).

3. EVALUATION

The algorithm was evaluated on the *Isophonics* [7] (180 songs by *The Beatles*, 20 songs by *Queen* and 18 songs by *Zweieck*) and *RWC Popular Music* [5] datasets (100 tracks), 318 tracks in total.

For each track the Mirex2010 metric (as described in [6]) was calculated. Then the chord average overlap ratio (AOR) and chord weighted average overlap ratio (WAOR) were calculated for the whole collection using following formulae:

$$AOR = \frac{1}{C}\sum_{i=1}^{C} m_i, \quad WAOR = \frac{\sum\limits_{i=1}^{C} l_i \cdot m_i}{\sum\limits_{i=1}^{C} l_i}$$

Here C is the number of tracks in the collection, m_i and l_i are the value of Mirex2010 metric and length for track m correspondingly. We employ these 2 overlap ratios as the measures of chord recognition quality. The segmentation value was also calculated for each track (as it is defined in [8]), as well as its average value for the whole collection.

Table 1 sums up the recognition quality shown by the proposed method for the two collections mentioned above.

Surprisingly, smoothing using self similarity matrix was more effective when dealing with the whole spectrogram columns. Actually, they play the role of feature vectors throughout this work. They are folded from 144 to 12 dimensions only at very end to make the matching with the templates easier. Probably the introduction of chord templates of greater dimensionality can improve the result.

Even though the proposed algorithm has no learning stage, its has some parameters, such as frequency range or the number of CRP coefficients which are set to 0. These parameters were set to obtain the best result on the whole collection, but locally optimal values can be different for each track. We believe that some procedure of automatical adjustment of the parameters for each track before chord recognition can have positve impact on the result. This is another subject of future work.

4. REFERENCES

[1] Taemin Cho and Juan Pablo Bello. A feature smoothing method for chord recognition using recurrence plots. In Anssi Klapuri and Colby Leider, editors, *ISMIR*, pages 651–656. University of Miami, 2011.

[2] Matthew E. P. Davies and Mark D. Plumbley. Context-dependent beat tracking of musical audio. *Trans. Audio, Speech and Lang. Proc.*, 15(3):1009–1020, March 2007.

[3] W. Bas De Haas, Jos Pedro Magalhes, and Frans Wiering. Improving audio chord transcription by exploiting harmonic and metric knowledge. In *Proceedings of the 13th International Society for Music Information Retrieval Conference*, Porto, Portugal, October 8-12 2012.

[4] Simon Dixon. Evaluation of the Audio Beat Tracking System BeatRoot. *Journal of New Music Research*, 36(1):39–50, March 2007.

[5] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. Rwc music database: Popular, classical and jazz music databases. In *ISMIR*, 2002.

[6] Geoffroy Peeters Johan Pauwels. Evaluating automatically estimated chord sequences. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-13)*, Vancouver, Canada, May 26-31 2013.

[7] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. OMRAS2 metadata project 2009. In *Late-breaking session at the 10th International Conference on Music Information Retrieval, Kobe, Japan*, 2009.

[8] Matthias Mauch. *Automatic Chord Transcription from Audio Using Computational Models of Musical Context*. PhD thesis, Queen Mary University of London, 2010.

[9] Matthias Mauch and Simon Dixon. Approximate note transcription for the improved identification of difficult chords. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, 2010.

[10] M. Müller and S. Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):649–662, 2010.

[11] João Lobato Oliveira, Matthew E. P. Davies, Fabien Gouyon, and Luís Paulo Reis. Beat tracking for multiple applications: A multi-agent system architecture with state recovery. *IEEE Transactions on Audio, Speech & Language Processing*, 20(10):2696–2706, 2012.

[12] Thomas Rocher, Matthias Robine, Pierre Hanna, and Laurent Oudre. Concurrent estimation of chords and keys from audio. In J. Stephen Downie and Remco C. Veltkamp, editors, *ISMIR*, pages 141–146. International Society for Music Information Retrieval, 2010.

[13] Yongwei Zhu, Mohan S. Kankanhalli, and Sheng Gao. Music key detection for musical audio. In *MMM*, pages 30–37, 2005.