AUDIO CHORD ESTIMATION USING CHROMA REDUCED SPECTROGRAM AND SELF-SIMILARITY

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

ABSTRACT

In this paper we propose 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. Chord templates used for recognition are binary-like, but have the tonic and the 5th note accented. Additional correction is performed to reduce the number of major-minor chord confusions. The method is evaluated on the 13 *Beatles* albums and RWC Popular Music collection.

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. All of them use machine learning algorithms to some extent, except one of the variants of the method by T. Cho and J. P. Bello [2] and the method proposed by T. Rocher et al. [8]. But the usage of machine learning makes the algorithm dependent on training data. Until recently the only data set available to researchers was composed of 12 albums of *The Beatles*, one 2-CD album of *Queen* and one album of *Zweieck*. Not a long time ago the Music Audio and Research Lab in New York University started to annotate the songs from RWC collection of popular music [5]. 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 has some common steps with our previous work [3], but also has some differences. Here only one median filter is used to smooth the spectrum before conversion to a sequence of chroma vectors. Chroma DCT-reduced log pitch features [7] are calculated instead of PCP-like features. Self-similarity is applied to spectrogram columns instead of feature vectors and is used here in a more straightforward manner. Also, additional correction is applied to reduce the number of major-minor confusions.

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 [9]. 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: 120 frequency components per octave (or 10 components per note), minimal component frequency equal to 440 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 480 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 $j = 0, 1, ..., 479$. 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 1/10 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 *BeatRoot* library [1] was used. 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.

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: 60 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 240 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 $w = 17$ 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 [7] is applied here. Each spectrogram value v is replaced with $log_{10}(1000 \cdot v + 1)$. Then a discrete cosine transform of size 240 is applied to each spectrogram column. The first 10 resulting coefficients are set to zero and the inverse DCT is performed. This value is small compared with the one used in [7], 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 [2] and M. Mauch et al. [6]. 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 not required to have only diagonals.

Euclidean distance is used as a measure of similarity. The self-similarity matrix has zeroes on the main diagonal. It is normalized, so that $0 \le s_{ij} \le 1$ for any i, j. Then for each row we preserve only $M \cdot n$ minimal values and set all the others to 1 (here $0 \le M \le 1$). The value $M = 0.1$ 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 $j, j + 60, j + 120, j + 180$ for each $j = 0, ..., 59$ into one row. This results in a sequence of 60-dimensional chroma vectors $\{\hat{p}_i\}_{i=1}^{n}$. Each vector is then projected to 12 dimensions:

$$q_i[j] = \sum_{h=-2}^{2} \hat{p}_i[5j-h] \cdot d^{|h|}, \quad i = 0, ..., n, \quad j = 0, ..., 11$$

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 $\hat{p}_i[58]$ and $\hat{p}_i[59]$ are substituted.

2.6 Chord estimation

Chord templates are used to determine the corresponding chord for each chroma vector and Kullback-Leibler divergence to measure the distance from a template to a chroma vector.

The templates for major, minor, augmented, and diminished chords are used in the proposed method. A template is a 12-dimensional vector. For augmented and diminished chords this vector is binary. It has 1 on the positions corresponding to notes which compose the chord and 0 on other positions. But the templates for major and minor chords are not really binary. Template components that correspond to tonic and 5th note were additionally emphasized, so that the template for C:maj looks like (1.3, 0, 0, 0, 1, 0, 0, 1.3, 0, 0, 0, 0). Before distance calculation both template vector and chroma vector are normalized to have unit length in Euclidean space. We only mark as "no chord" the fragments that go before the 1st beat and after the last beat detected by *Beatroot* in each audio record.

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 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.

3. EVALUATION

The algorithm was evaluated on a collection of 13 Beatles albums annotated by C. Harte [4] (180 songs) and on RWC Popular Music & Royalty-Free Music collection [5] (100 songs) annotated by Music and Audio Research Lab at NYU.

For each track the frame-based recall has been calculated with the evaluations on the triad level. It means that

Collection	AOR	WAOR
The Beatles albums	79.21	78.17
RWC Popular Music	64.99	65.48
2 collecions together	74.13	73.33

Table 1. Recognition quality.

only the chords from this list have been checked: N, X:maj, X:min, X:aug, X:dim, X:sus2, X:sus4. All the other chords (including all 7th chords) have been excluded from recall calculation. For those tracks that contain other chords the effective length (that was used for recall calculation) is therefore less than the whole track length. Then the chord average overlap ratio (AOR) and chord weighted average overlap ratio (WAOR) were calculated for the whole collection using following formulae:

$$AOR = \frac{1}{C}\sum_{m=1}^{C} r_m, \quad WAOR = \frac{\sum_{m=1}^{C} l_m \cdot r_m}{\sum_{m=1}^{C} l_m}$$

Here C is the number of tracks in the collection, r_m and l_m are the frame-based recall and effective length for track m correspondingly. We employ these 2 overlap ratios as the measures of chord recognition quality.

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 240 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.

It also can be noted that the proposed method works better for The Beatles songs than for the songs from RWC collection. 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] BeatRoot. http://www.eecs.qmul.ac.uk/ simond/beatroot/, 2012.

[2] 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.

[3] N. Glazyrin and A. Klepinin. Chort Recognition using Prewitt Filter and Self-Similarity. In *Proceedings of the 9th Sound and Music Computing Conference*, pages 480–485, Copenhagen, Denmark, July 2012.

[4] Christopher Harte, Mark B. Sandler, Samer A. Abdallah, and Emilia Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *ISMIR*, pages 66–71, 2005.

[5] Takuichi Nishimura Masataka Goto, Hiroki Hashiguchi and Ryuichi Oka. RWC Music Database: Popular, Classical, and Jazz Music Databases. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, pages 287–288, October 2002.

[6] 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.

[7] 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.

[8] 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.

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