CHROMA FEATURE: ESTIMATION AND APPLICATIONS OF THE HARMONIC PITCH CLASS PROFILE

Junhao Wang
McGill University
MUMT 605 Final Project

1. OVERVIEW

Chroma features, also referred to as pitch class profiles [3], can be obtained by mapping the frequency bins obtained through Fourier transform to the twelve bins corresponding to the twelve pitch classes in a chromatic scale. By placing the chroma vector computed from each frame along the temporal axis, we obtain a 2-dimensional graph representation, which is called a chromagram. As a widely used representation in signal processing, the spectrogram displays the energy distribution of the signal over frequency and time. Similarly, the chromagram reflects the energy distribution over pitch classes (or chroma) and time. As pitches with the same chroma often play similar harmonic roles, the chroma features are often more useful than the spectrum-based features in representing the harmonic aspects of music.

1.1 Objectives

The main objective of this project is to explore the chroma feature and its applications in computational music processing and analysis. Specifically, an algorithm for extracting Harmonic Pitch Class Profile (HPCP) from polyphonic audio signals is implemented. As an application and validation of the HPCP feature, a key estimation experiment is performed. The implemented algorithm and experiment are both adapted from [4] with slight modifications. By verifying related theory and analyzing the results, it is shown that HPCP closely correlates to the tonal aspects of polyphonic music, and can thus be a powerful tool in estimating the key of a musical piece or measuring the similarity in terms of key between pieces and genres.

1.2 Report Structure

This report is organized as follows. Section 2 gives a brief description of concepts of HPCP and other related features. Section 3 presents the methodology for extracting HPCP from polyphonic audio signals. Section 4 reports on the design of the key estimation experiment and evaluation of the results. The conclusion, limitation, and potential improvements are discussed in section 5. Finally, a detailed description of the MATLAB implementation is given in section 6.

2. THE CONCEPTS

Chroma feature is related to our perception of music. Human perception of a musical pitch can be represented in two dimensions: *tone height* and *chroma* [7]. The tone height is related to the rise in perceived pitch when the frequency increases while the chroma relates to the perceived similarity in "color" between notes that differ by one or several octaves. This makes the chroma feature an ideal musically-informed feature for analyzing music audio signals.

2.1 HPCP, PCP, and Chromagram

In western music notations, chroma is denoted by the pitch class and tone height is denoted by the octave number. As proposed in [4], HPCP can be considered a variant of the pitch class profile (PCP) proposed by [3] and the intensity map in the Simple Auditory Model (SAM) proposed by [5]. Essentially, it is a 12-dimensional (or 24, 36... depending on bin resolution) vector that represents the energy distribution over the twelve different pitch classes in an equal-tempered scale. An instantaneous HPCP vector can be considered a column taken from a chromagram without logarithmic compression.

As proposed in [4], the computation of HPCP is based on PCP, with the following modifications: first, only a subset of spectral peaks contribute to HPCP bins; second, 36 instead of 12 bins are used in an HPCP vector, increasing the resolution to 1/3 of a semitone; third, each eligible spectral peak contributes to more than 1 bins in HPCP, and the contributions are scaled by a weighting function. This is further explained in the HPCP Computation section.

2.2 Feature Hierarchy

Before presenting the procedure for calculating instantaneous and global HPCP, we briefly describe the different temporal scales and levels of abstraction of the features as proposed in [4]. In the feature design process of this project, we distinguish between features on two abstraction levels. Features directly computed from the signal are considered low-level features. By processing and analyzing low-level features, we can obtain descriptors that reflect the musical content of the signal, which are defined as high-level features. We also distinguish between two temporal scales. Features related to one analysis frame are defined as instantaneous features while features related to a

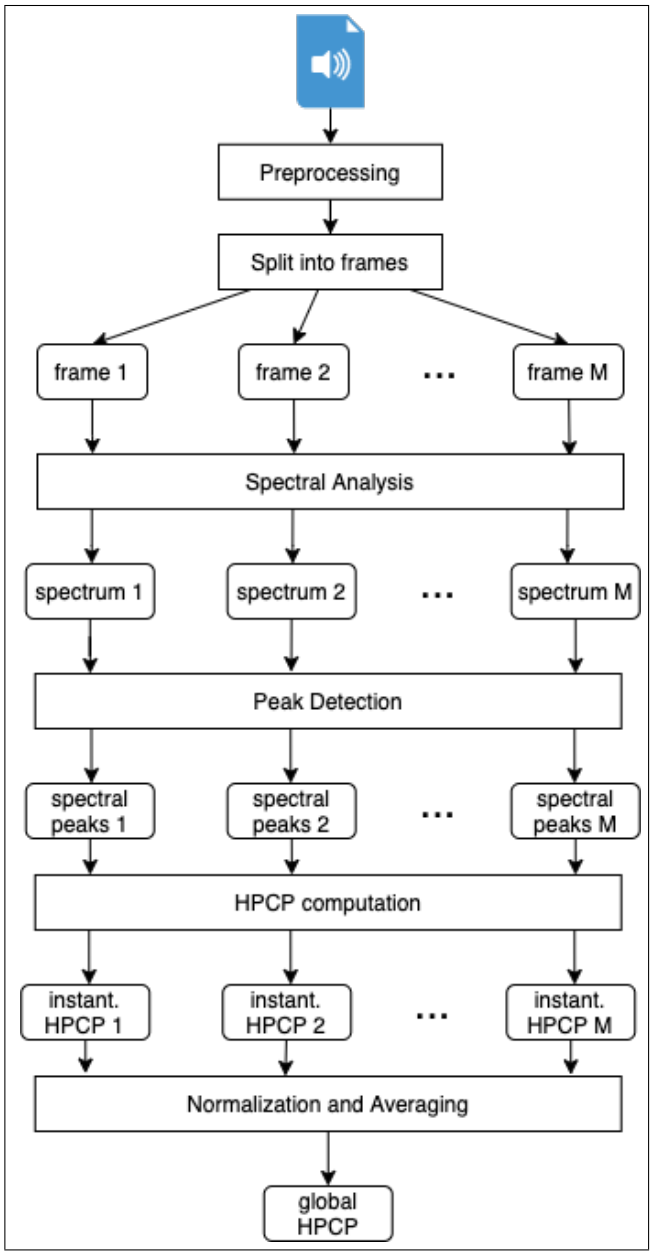

Figure 1. HPCP Computation

whole piece of music is considered global features. For example, the HPCP vector obtained by analyzing one frame of audio data is an instantaneous low-level feature. By averaging the instantaneous HPCPs vectors over all frames, we get one HPCP vector for the whole piece, which is a global low-level feature. The global HPCP vector is then correlated with the key profile matrix to produce the key estimation result for the piece. This includes the estimated tonic, mode, and key strength, which are all global high-level features.

3. EXTRACTING HPCP

An overview of the algorithm for computing the instantaneous and global HPCP vectors is shown in Figure 1. The algorithm is adapted from [4] with slight modifications. Each component of the algorithm will be described in more detail in the following subsections.

3.1 Preprocessing

The preprocessing scheme used in this project is different from what is proposed in [4]. In [4], the preprocessing is performed on a frame-by-frame basis, where a transient detection algorithm is used to eliminate the transient regions in each frame, so that the harmonically-noisy areas within the frame are not analyzed. This can reduce the computational cost, but some information is lost. Considering the transient regions only make up a small part of the whole frame, they should not make much difference in the overall result. Therefore, we choose to keep the transient regions in our analysis. Besides, we add one preprocessing step on the piece level, which is to remove the leading and trailing zeros in each audio excerpt.

3.2 Spectral Analysis

After the preprocessing step, the input audio signal is split into frames. Then spectral analysis is performed on each frame. Each analysis frame is multiplied by a window function. In development for this project, different window functions are tested. In the end, Blackman-Harris window is used in the implementation. In order to achieve adequate frequency resolution, we use a frame size of 4096 samples and a hop size of 512 samples. Fast Fourier Transform (FFT) is then performed on each windowed frame.

For a sampling rate of 44.1KHz, one frame lasts for about 92.9 ms. This further disproves the usage of the transient elimination algorithm used in [4], where the areas located 50ms before and after the transients are eliminated and ignored for analysis. Removing regions near the transients this way could result in whole frames of data being eliminated, which may jeopardize the integrity of harmonic information captured by the HPCP.

3.3 Peak Detection

Similar to [4], we use peak detection to capture the local maxima in each magnitude spectrum, i.e. spectral peaks. In [4], two constraints are set for the peaks. For a spectral peak to be considered in HPCP computation, its magnitude has to be higher than a threshold (-100dB), and its frequency must fall within a frequency range of [100, 5000] Hz. In development for this project, it is found that adding this magnitude threshold does not produce significant change in the result. Therefore the magnitude constraint is discarded. On the other hand, the frequency constraint is kept in order to eliminate percussion and instrumental noise that are often present in music audio recordings.

3.4 HPCP Computation

Using the spectral peaks extracted from each frame, we compute the instantaneous HPCP vector. In traditional

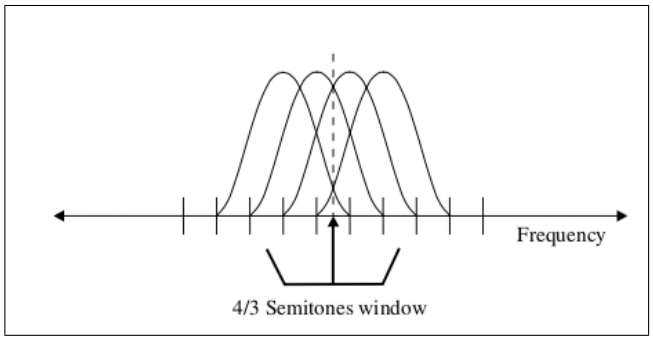

Figure 2. Weighting Function [4]

chroma feature computation, each peak in the spectrum only contributes to one chroma bin, i.e. the bin whose center frequency is the closest to the peak frequency. In this project, a higher chroma resolution is used. Instead of 12 bins corresponding to the 12 semitones in an equal-tempered scale, 36 bins are used, which increases the bin resolution to 1/3 semitone. Also, each spectral peak contributes to a weighting window of 4 bins, i.e. 4/3 semitones. The weight for each spectral peak is determined by its distance from the center frequency of the bin. The center frequency of the *n*th chroma bin is obtained by

$$f_n = f_{ref} 2^{n/size} \tag{1}$$

where n=1,...,$size$, and in our case, $size$=36. We choose the concert pitch $A4$=440Hz as the reference frequency f_{ref} here, but other notes can also be used without altering the result.

The absolute distance in semitones between the frequency of the ith peak f_i and the nth bin center frequency is given by

$$d_s = 12 \log_2 \left(\frac{f_i}{f_n} \right) \tag{2}$$

Considering octave equivalence, this absolute distance in semitones can be converted to pitch class distance by

$$d_p = d_s + 12m \tag{3}$$

where m is an integer that minimizes the absolute value of distance. Then the weight function is given by

$$w\left(n, f_i\right) = \begin{cases} \cos^2\left(\frac{\pi}{2} \frac{d}{0.5 \cdot l}\right) & |d| \leq 0.5l \\ 0 & |d| > 0.5l \end{cases} \tag{4}$$

where l equals the weighting window length in semitones. In this case, l=4/3. This gives the weight of the ith peak frequency in computing the nth bin of the instantaneous HPCP vector. A plot of the weighting function is shown in Figure 2, which is taken from [4].

Finally, the instantaneous HPCP values are computed by

$$\mathrm{HPCP}(n) = \sum_{i}^{n_{peaks}} w\left(n, f_i\right) a_i^2 \tag{5}$$

where n=1,2,...,$size$ denotes the HPCP bin index. n_{peaks} stands for the number of spectral peaks in the current frame, and the amplitude of the ith peak is denoted by a_i. In the key estimation context, the weighting scheme and higher chroma resolution make the model more robust against slight tuning differences that could be present between different music pieces, and thus reduce the estimation errors.

For each frame, the instantaneous HPCP vector is normalized with respect to its maximum value

$$\mathrm{HPCP}_{\text{normalized}}\left(n\right) = \frac{\mathrm{HPCP}(n)}{\mathrm{Max}_n(\mathrm{HPCP}(n))} \tag{6}$$

By taking the average over all instantaneous HPCPs for a given piece, we obtain the global HPCP, which captures the harmonic information of the piece and is then used in estimating the key.

4. KEY ESTIMATION

Key describes the tonic and mode of a tonal piece of music. To validate the HPCP features, we implement the key estimation experiment described in [4]. In this section, we present the key estimation pipeline using the above mentioned HPCP audio feature. We describe the test dataset, present the estimation algorithm, and evaluate the estimation results.

4.1 Dataset

In this project, an internal dataset within my lab is used. The dataset consists of 164 pieces of classical music, which are all synthesized from MIDI files and 45 pieces of classical music recordings recorded from live performances, including 23 pieces of piano recordings and 22 pieces of cello recordings. Each audio file is annotated manually with its key and has constant tonic and mode throughout the piece. The dataset covers a variety of different tonics and modes. The key distribution is shown in Table 1.

4.2 Procedure

For each audio file in the dataset, the global HPCP is extracted using the methodology mentioned in the last section. To estimate the key, the global HPCP vector of each piece is correlated with a matrix K of key profiles.

$$R(i,j) = r(\mathrm{HPCP}, K(i,j)) \tag{7}$$

i=1,2 is the index of modes. 1 denotes major and 2 denotes minor. j=1,...,12 is the index of tonics. The highest correlation value $R\left(i_{\max}, j_{\max}\right) = \max_{i,j}(R(i,j))$ corresponds to the estimated mode and tonic, indexed by $i_{\max}$ and $j_{\max}$ respectively. The correlation value itself measures the degree of key strength, that is, how "tonal" the music is.

Key profiles describe the tonal hierarchies of major and minor keys. The key profile matrix K is of the size $2 \times 12 \times 36$, corresponding to 2 modes, 12 key notes, and 36 HPCP bins. Each key profile vector $K(i,j)$ is of the same

Tonic	Major	Minor	Total	Percentage
A	10	20	30	14.35
A♯/B♭	16	0	16	7.65
B	1	4	5	2.39
C	18	13	31	14.83
C♯/D♭	0	4	4	1.91
D	20	17	37	17.70
D♯/E♭	23	1	24	11.48
E	5	5	10	4.78
F	19	1	20	9.57
F♯/G♭	1	0	1	0.48
G	22	8	30	14.35
G♯/A♭	1	0	1	0.48

Table 1. Dataset Key Distribution

size as the HPCP vector. There are 12 different tonics and 2 different modes, so there are a total of 24 key profile vectors. The key profile values measure how each pitch class fits into different keys. As shown in Figure 3, the key profiles used in this project is directly taken from [4], which is 12 samples in length. Linear interpolation is then performed to expand the profile to the same size as the HPCP vector. The key profiles are assumed to be transposition-invariant, i.e. do not change with respect to different tonics. All 24 key profiles can thus be obtained by simply shifting the major and minor profiles shown in Figure 3.

In analyzing a polyphonic audio excerpt in C major, its global HPCP is computed and shown in Figure 4 It can be seen that the highest energy peaks occur at the tonic C and the dominant G. Other peaks can be observed at the second

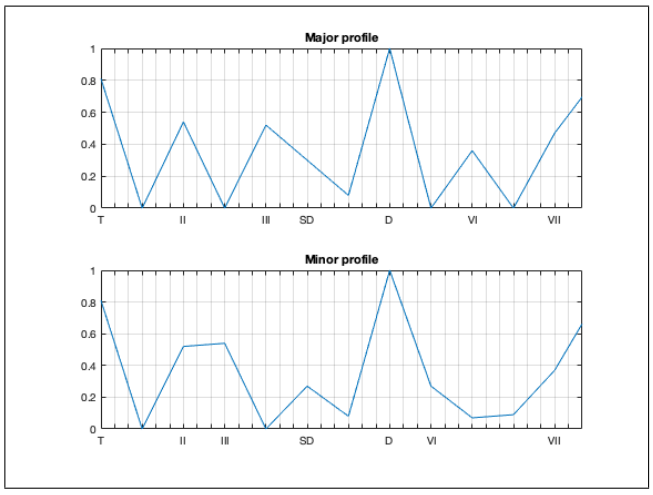

Figure 3. Interpolated Key Profiles.

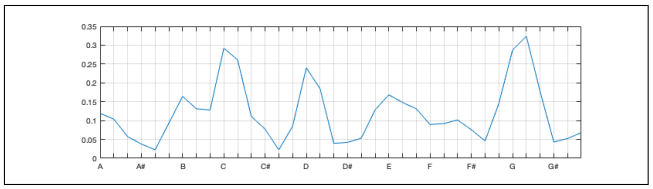

Figure 4. Global HPCP of an excerpt in C Major.

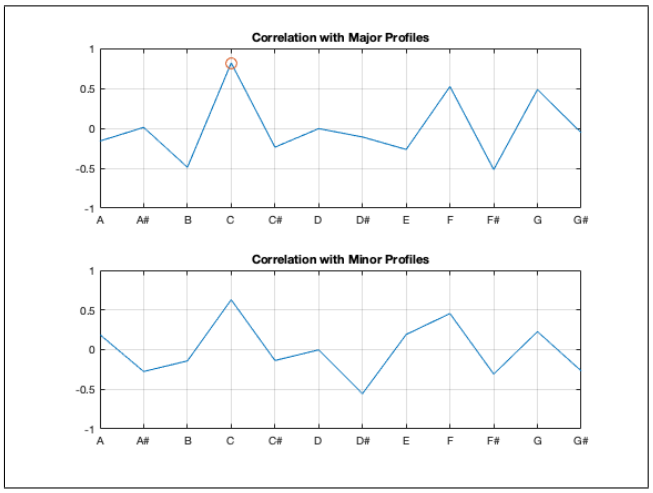

Figure 5. Correlations of a C Major HPCP with the key profiles.

degree D and the third degree E, which agrees with the major key profile. Its correlations with the 24 key profiles is shown in Figure 5 The highest correlation value is 0.82, which occurs in C major. This determines the estimated tonality. The second highest peak occurs at C minor, which is the relative minor that shares the same dominant and sub-dominant chords.

4.3 Evaluation

The same key estimation procedure is used on all excerpts in the dataset. Considering the significant difference in results, we separately evaluate the performance of the algorithm on the MIDI-synthesized data and the real recordings. The same metrics as [4] are used.

4.3.1 MIDI-Synthesized Audio

The result of key estimation on MIDI-synthesized audio excerpts is shown in Table 2. It is observed that 83.54% of excerpts are correctly estimated in terms of both tonic and mode. This performance proves that the key estimation model is doing well, and the HPCP features do manage to capture the relative intensity of each pitch class, which is the key requisite for performing key estimation.

4.3.2 Real Recordings

The algorithm is also evaluated on real recordings, i.e. recordings made from live performances, which are generally more noisy than MIDI-synthesized excerpts. The real

Metric	Number	Percentage
Correct	137	83.54
Only mode errors	3	1.83
Only key note errors	8	4.88
Key note and mode errors	16	9.75

Table 2. Evaluation results on MIDI-synthesized data

Metric	Number	Percentage
Correct	18	78.26
Only mode errors	1	4.35
Only key note errors	1	4.35
Key note and mode errors	3	13.04

Table 3. Evaluation results on piano recordings

recordings in our dataset contain both piano recordings and cello recordings. The results are shown separately for reasons that we will explain later.

As shown in Table 3, for piano recordings, 78.26% are correctly estimated. The accuracy is slightly lower than estimating synthesized audio. This is normal as real recordings generally contain more background noise and potential tempo and timbre changes, which can likely affect the accuracy.

However, the algorithm is not working well on cello recordings. The result on cello recordings is shown in Table 4

It is shocking at first to see such a poor result. Using real recordings, it is expected to see a worse performance than synthesized audio, but such a huge degradation is totally unanticipated. After observing and comparing the estimated output and the ground-truth side-by-side, it is seen that out of the 19 incorrect estimations with only key note errors, 17 of them are higher than the ground-truth by exactly one semitone. A segment of the output file showing the results is shown in Figure 6. The first column shows the audio file names (without extension). The second column shows the key labels, which are considered the ground-truth. The third and fourth columns show the estimated

Metric	Number	Percentage
Correct	1	4.55
Only mode errors	0	0
Only key note errors	19	86.36
Key note and mode errors	2	9.09

Table 4. Evaluation results on cello recordings

```
CelloSuite1iv_G    G   G# 0.469241
 CelloSuite1v_G    G   G# 0.489701
CelloSuite1vi_G    G   G# 0.436877
 CelloSuite2i_d    d   d# 0.662897
CelloSuite2ii_d    d   d# 0.697173
CelloSuite2iv_d    d   d# 0.660208
 CelloSuite2v_d    d   D# 0.691549
CelloSuite2vi_d    d   d# 0.651625
 CelloSuite3i_C    C   C# 0.553865
CelloSuite3ii_C    C   C# 0.560551
CelloSuite3iii_C   C   C# 0.453135
```

Figure 6. A segment of the output file showing results

key and key strength respectively.

To investigate this interesting error pattern, we used the Sonic Visualizer software [2] to inspect the tuning of the recordings. The global tuning, i.e. frequency of the concert pitch in Hz, is estimated on piano recordings, cello recordings, as well as MIDI-synthesized excerpts. It is seen that the piano recordings and MIDI-synthesized audio files generally use a concert pitch of around 440Hz, which is the standard tuning, while the cello recording portion of the dataset is roughly centered around A4=450Hz. This tuning difference is likely responsible for the one semitone deviation in the key estimation of the cello excerpts.

To further verify this, the HPCP bins are re-computed. In order to accommodate the tuning used by cello recordings, the reference frequency used in Equation 1 is set to 450Hz, and the center frequencies of HPCP bins are calculated accordingly. However, changing this frequency in the model only corrected some of the results on the predictions. The accuracy achieved on cello recordings is still far from the accuracy achieved on piano recordings and MIDI-synthesized audio files, which are shown in Table 2 and Table 3.

5. CONCLUSION AND FUTURE WORK

In this project, the algorithm for extracting HPCP and key estimation proposed by [4] is implemented with several modifications. The algorithm achieved satisfactory performance on both MIDI-synthesized pieces and piano excerpts recorded from live performances.

There was an interesting error pattern in which the predictions of most cello recordings were off by one semitone. An inspection of the tuning of the cello pieces indicated that they were roughly centered around A4=450Hz. However, changing this frequency in the model did not produce satisfactory results on the predictions. This suggests that more complex difference might exist between acoustic cello performances and acoustic piano performances. More investigation into this difference is left for future work.

Potential performance improvement could be achieved by making the algorithm more robust against potential tuning difference and background noise. It is also ob-

served that estimated key strength is generally lower on real recordings than MIDI-synthesized excerpts, which suggests that the noise and timbre changes in real recordings might affect the quality of extracted HPCP. In order to build a more robust model and achieve better performance, more complex preprocessing steps and chroma enhancing techniques might be required.

Moreover, there are various methods for extracting chroma features from audio signals. In this project, the method based on Short Time Fourier Transform (STFT) is implemented, which is more closely related to what we have seen in class. As seen In some other literature [1] [6]. Constant-Q Transform (CQT) is another commonly used approach for extracting chroma features, where the frequency channels are logarithmically spaced. We leave the exploration of CQT-based methodology for future work.

6. IMPLEMENTATION DETAILS

This project is implemented entirely in MATLAB. The code for all components of this project is available online [1] .

The algorithm for extracting HPCP from audio signals is implemented in *show_hpcp.m*, which reads a single audio file, computes the instantaneous HPCP for every frame and then generates a plot of the global HPCP vector. This part of the algorithm is functionized in *get_hpcp.m*.

The key profiles used in key estimation can be plotted with *show_profile.m*, where the 12 discrete values of both major and minor key profiles are linearly interpolated to get all 36 key profile values. This part is functionized in *get_profile.m*. These two scripts both call *interp_profile.m*, which is a function for linear interpolation.

To generate a key estimation for one single audio excerpt, run *show_single.m*. This script is dependent on the functions *get_profile.m* and *get_hpcp.m*. It generates a plot of the correlation values with all 24 key profiles, like the one shown in Figure 5, the highest correlation value is circled in red. The label corresponding to the highest correlation value gives the estimated key.

The key estimation process is functionized in *estm_key.m*. It makes use of *get_hpcp.m* and *get_profile.m*, and calculates a correlation value for each of the 24 key profile vectors. The highest correlation value (tonalness) and its corresponding label are returned.

Putting it all together, the experiment on the test dataset is performed by running *main.m*. It loops through all audio files in a directory and writes the key estimation result, including the file names, estimated keys, and tonalness values, into a text file.

All the scripts and audio files in the project folder are arranged in a way that is ready to run without the need for any modification. Nonetheless, everything is customizable and different audio files can also be tested.

[1] https://github.com/jwang44/HPCP-Key-Finder

7. REFERENCES

[1] Juan Pablo Bello and Jeremy Pickens. A robust mid-level representation for harmonic content in music signals. In *ISMIR*, volume 5, pages 304–311. Citeseer, 2005.

[2] Chris Cannam, Christian Landone, and Mark Sandler. Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1467–1468, 2010.

[3] Takuya Fujishima. Real-time chord recognition of musical sound: A system using common lisp music. *Proc. ICMC, Oct. 1999*, pages 464–467, 1999.

[4] Emilia Gómez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, 18(3):294–304, 2006.

[5] Marc Leman. Auditory models of pitch perception. In *Music and Schema Theory*, pages 43–60. Springer, 1995.

[6] Meinard Müller and Sebastian Ewert. Chroma toolbox: Matlab implementations for extracting variants of chroma-based audio features. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR), 2011. hal-00727791, version 2-22 Oct 2012*. Citeseer, 2011.

[7] Roger N Shepard. Circularity in judgments of relative pitch. *The Journal of the Acoustical Society of America*, 36(12):2346–2353, 1964.