Friedrich-Alexander-Universität Erlangen-Nürnberg

Lab Course

Short-Time Fourier Transform and Chroma Features

International Audio Laboratories Erlangen

Prof. Dr. Meinard Müller

Friedrich-Alexander Universität Erlangen-Nürnberg
International Audio Laboratories Erlangen
Lehrstuhl Semantic Audio Processing
Am Wolfsmantel 33, 91058 Erlangen

meinard.mueller@audiolabs-erlangen.de

AUDIO LABS

International Audio Laboratories Erlangen
A Joint Institution of the
Friedrich-Alexander Universität Erlangen-Nürnberg (FAU) and
the Fraunhofer-Institut für Integrierte Schaltungen IIS

Fraunhofer IIS

Authors:

Meinard Müller
Stefan Balke

Tutors:

Meinard Müller
Stefan Balke

Contact:

Meinard Müller, Stefan Balke
Friedrich-Alexander Universität Erlangen-Nürnberg
International Audio Laboratories Erlangen
Lehrstuhl Semantic Audio Processing
Am Wolfsmantel 33, 91058 Erlangen
meinard.mueller@audiolabs-erlangen.de
stefan.balke@audiolabs-erlangen.de

Lab Course

Short-Time Fourier Transform and Chroma Features

Abstract

The Fourier transform, which is used to convert a time-dependent signal to a frequency-dependent signal, is one of the most important mathematical tools in audio signal processing. Applying the Fourier transform to local sections of an audio signal, one obtains the short-time Fourier transform (STFT). In this lab course, we study a discrete version of the STFT. To work with the discrete STFT in practice, one needs to correctly interpret the discrete time and frequency parameters. Using MATLAB, we compute a discrete STFT and visualize its magnitude in form of a spectrogram representation. Then, we derive from the STFT various audio features that are useful for analyzing music signals. In particular, we develop a log-frequency spectrogram, where the frequency axis is converted into an axis corresponding to musical pitches. From this, we derive a chroma representation, which is a useful tool for capturing harmonic information of music.

1 Introduction

Audio signals can be complex mixtures consisting of a multitude of different sound components. A first step in better understanding a given signal is to decompose it into building blocks that are better accessible for the subsequent processing steps. In the case that these building blocks consist of complex-valued sinusoidal functions, such a process is also called Fourier analysis. The Fourier transform maps a time-dependent signal to a frequency-dependent function which reveals the spectrum of frequency components that compose the original signal. Loosely speaking, a signal and its Fourier transform are two sides of the same coin. On the one side, the signal displays the time information and hides the information about frequencies. On the other side, the Fourier transform reveals information about frequencies and hides the time information.

To obtain back the hidden time information, Dennis Gabor introduced in the year 1946 the modified Fourier transform, now known as *short-time Fourier transform* or simply STFT. This transform is a compromise between a time- and a frequency-based representation by determining the sinusoidal magnitude and phase content of local sections of a signal as it changes over time. In this way, the STFT does not only tell which frequencies are "contained" in the signal but also at which points of times or, to be more precise, in which time intervals these frequencies appear.

The main objective of this lab course is to acquire a good understanding of the STFT. To this end, we study a discrete version of the STFT using the discrete Fourier transform (DFT), which can be efficiently computed using the fast Fourier transform (FFT). The discrete STFT yields a discrete set of Fourier coefficients that are indexed by time and frequency parameters. The correct physical interpretation of these parameters in terms of units such as seconds and Hertz depends on the sampling rate, the window size, and the hop size used in the STFT computation. In this lab course, we will compute a discrete STFT using MATLAB and then visualize its magnitude by a spectrogram representation, see Section 2 and Figure 1b. By applying the STFT to different audio examples and by modifying the various parameters, one should get a better understanding on how the STFT works in practice.

To make music data comparable and algorithmically accessible, the first step in basically all music processing tasks is to extract suitable *features* that capture relevant aspects while suppressing irrelevant details. In the second part of this lab course, we study audio features and mid-level representations that are particularly useful for capturing pitch information of music signals. Assuming that we are dealing with music that is based on the equal-tempered scale (the scale that

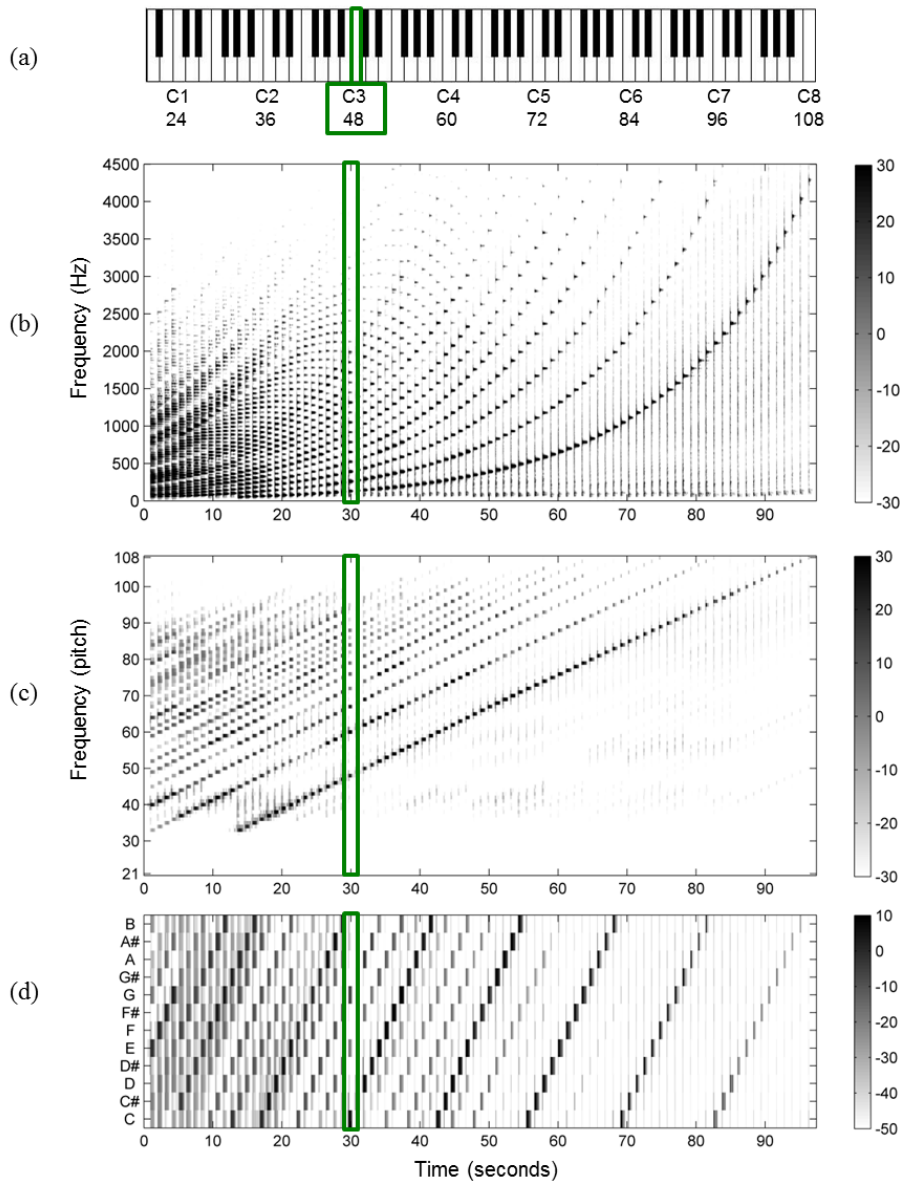

Figure 1: Various representations for a piano recording of the chromatic scale ranging from A0 ($p = 21$) to C8 ($p = 108$). **(a)** Piano keys representing the chromatic scale. **(b)** Spectrogram representation. **(c)** Pitch-based log-frequency spectrogram. **(d)** Chromagram representation. For visualization purposes the values are color-coded using a logarithmic scale. The C3 ($p = 48$) played at time $t = 30$ sec has been highlighted by the rectangular frames.

corresponds to the keys of a piano keyboard), we will convert an audio recording into a feature representation that reveals the distribution of the signal's energy across the different pitches, see Section 3 and Figure 1c. Technically, these features are obtained from a spectrogram by converting the linear frequency axis (measured in Hertz) into a logarithmic axis (measured in pitches). From this log-frequency spectrogram, we then derive a time-chroma representation by suitably combining pitch bands that correspond to the same chroma, see Section 4 and Figure 1d. The resulting chroma features show a high degree of robustness to variations in timbre and instrumentation.

2 STFT

The Fourier transform and in particular the discrete STFT serve as *front-end transform*, the first computing step, for deriving a large number of different musically relevant audio features. We now recall the definition of the discrete STFT while fixing some notation. Let $x : [0 : L-1] := \{0, 1, \ldots, L-1\} \to \mathbb{R}$ be a real-valued discrete-time signal of length L obtained by equidistant sampling with respect to a fixed sampling rate F_s given in Hertz (Hz). Furthermore, let $w : [0 : N-1] := \{0, 1, \ldots, N-1\} \to \mathbb{R}$ be a discrete-time window of length $N \in \mathbb{N}$ (usually a power of two) and let $H \in \mathbb{N}$ be a hop size parameter. With regards to these parameters, the discrete STFT $\mathcal{X}$ of the signal x is given by

$$\mathcal{X}(m,k) := \sum_{n=0}^{N-1} x(n+mH)w(n)\exp(-2\pi ikn/N) \tag{1}$$

with $m \in [0 : \lfloor \frac{L-N}{H} \rfloor]$ and $k \in [0 : K]$. The complex number $\mathcal{X}(m,k)$ denotes the k^{th} Fourier coefficient for the m^{th} time frame, where $K = N/2$ is the frequency index corresponding to the Nyquist frequency. Each Fourier coefficient $\mathcal{X}(m,k)$ is associated with the physical time position (using the start position of the window as reference point)

$$T_{\text{coef}}(m) := \frac{m \cdot H}{F_s} \tag{2}$$

given in seconds (sec) and with the physical frequency

$$F_{\text{coef}}(k) := \frac{k \cdot F_s}{N} \tag{3}$$

given in Hertz (Hz). For example, using $F_s = 44100$ Hz as for a CD recording, a window length of $N = 4096$, and a hop size of $H = N/2$, we obtain a time resolution of $H/F_s \approx 46.4$ ms and frequency resolution of $F_s/N \approx 10.8$ Hz.

Homework Exercise 1

(a) Compute the time and frequency resolution of the resulting STFT when using the following parameters. What are the Nyquist frequencies?
   (i) $F_s = 22050$, $N = 1024$, $H = 512$
   (ii) $F_s = 48000$, $N = 1024$, $H = 256$
   (iii) $F_s = 4000$, $N = 4096$, $H = 1024$

(b) Using $F_s = 44100$, $N = 2048$ and $H = 1024$, what is the physical meaning of the Fourier coefficients $\mathcal{X}(1000, 1000)$, $\mathcal{X}(17, 0)$, and $\mathcal{X}(56, 1024)$?

The STFT is often visualized by means of a *spectrogram*, which is a two-dimensional representation of the squared magnitude:

$$\mathcal{Y}(m,k) = |X(m,k)|^2. \tag{4}$$

When generating an image of a spectrogram, the horizontal axis represents time, the vertical axis is frequency, and the dimension indicating the spectrogram value of a particular frequency at a particular time is represented by the intensity or color in the image.

Lab Experiment 1

- Use the function `audioread` to read the file `Sound_TwoSineTwoImpulse.wav`. This defines a signal x as well as the sampling rate F_s. In the case that the signal is stereo, only use the first channel.
- Initialize a length parameter $N = 4096$ and a hop size parameter $H = 2048$.
- Define a window function w of length N (using `hann`).
- Compute $\mathcal{X}$ using the function `S=spectrogram(X,WINDOW,NOVERLAP)`. To this end, one needs to compute the window overlap from N and H. The matrix `S` contains the complex-valued Fourier coefficients $\mathcal{X}(m, k)$.
- Compute the spectrogram $\mathcal{Y}(m, k)$ as in (4).
- Using (2), compute a vector `T` that contains the physical time positions (in seconds) of the time indices.
- Using (3), compute a vector `F` that contains the frequency values (in Hertz) of the frequency indices.
- Visualize the spectrogram in various ways using the functions `image`, `imagesc`, `axis xy`, `colorbar`, and so on. Doing so, also get familiar with the various visualization parameters and tools offered by MATLAB.
- Plot the spectrogram with the axis given in form of indices.
- Plot the spectrogram with the axis given in seconds and Hertz. This should be done by applying the functions `image` or `imagesc` using `T` and `F` as additional parameters.
- Next, use a logarithmic decibel-scale for visualizing the values $\mathcal{Y}(m, k)$. (Recall that, given a value $v \in \mathbb{R}$, the decibel value is $10 \log_{10}(v)$.)
- Compute spectrograms using different window sizes (for example, $N \in \{256, 1024, 4096, 8192\}$) and different hop sizes (for example, $H \in \{1, N/4, N/2\}$). Discuss the trade-off between time resolution and frequency resolution.
- Try out other audio files.

The human sensation of the intensity of a sound is logarithmic in nature. In practice, sounds that have an extremely small intensity may still be relevant for human listeners. Therefore, one often uses a decibel scale, which is a logarithmic unit expressing the ratio between two values. As alternative of using a decibel scale, one often applies in audio processing a step also referred to as *logarithmic compression*, which works as follows. Let $\gamma \in \mathbb{R}_{>0}$ be a positive constant and $\Gamma_\gamma : \mathbb{R}_{>0} \to \mathbb{R}_{>0}$ a function defined by

$$\Gamma_\gamma(v) := \log(1 + \gamma \cdot v). \tag{5}$$

for $v \in \mathbb{R}_{>0}$, where we use the natural logarithm. Note that the function Γ_γ yields a positive value $\Gamma_\gamma(v)$ for any positive value $v \in \mathbb{R}_{>0}$. Now, for a representation with positive values such as a spectrogram, one obtains a compressed version by applying the function Γ_γ to each of the values:

$$(\Gamma_\gamma \circ \mathcal{Y})(m, k) := \log(1 + \gamma \cdot \mathcal{Y}(m, k)). \tag{6}$$

Why is this operation called "compression" and what is the role of the constant γ? The problem with representations such as a spectrogram is that its values possess a large dynamic range. As a result, small, but still relevant values may be dominated by large values. Therefore, the idea of compression is to balance out this discrepancy by reducing the difference between large and small values with the effect to enhance the small values. This exactly is done by the function Γ_γ, where the degree of compression can be adjusted by the constant γ. The larger γ, the larger the resulting compression

Homework Exercise 2

Plot the function Γ_γ for the parameters $\gamma \in \{1, 10, 100\}$.

Lab Experiment 2

- Use the file `Tone_C4_Piano.wav` to define a signal x.
- Compute the STFT and the spectrogram $\mathcal{Y}$ as above using a Hann window of size $N = 4096$ and a hop size $H = 2048$.
- Compute the compressed version $\Gamma_\gamma \circ \mathcal{Y}$ of the spectrogram using different constants $\gamma \in \{1, 10, 100, 1000, 10000\}$.
- Visualize the original spectrogram and its compressed versions. What do you see? Discuss the results.
- Try out other audio files.

3 Log-Frequency Spectrogram

We now derive some audio features from the STFT by converting the frequency axis (given in Hertz) into an axis that corresponds to musical pitches. In Western music, the *equal-tempered scale* is most often used, where the pitches of the scale correspond to the keys of a piano keyboard. In this scale, each octave (the interval between two tones, whose fundamental frequencies differ by a factor of two) is split up into twelve logarithmically spaced units. In MIDI notation, one considers 128 pitches, which are serially numbered starting with 0 and ending with 127. The MIDI pitch $p = 69$ corresponds to the pitch A4 (having a center frequency of 440 Hz), which is often used as standard for tuning musical instruments. In general, the center frequency $F_{\text{pitch}}(p)$ of a pitch $p \in [0 : 127]$ is given by the formula

$$F_{\text{pitch}}(p) = 2^{(p-69)/12} \cdot 440. \tag{7}$$

The logarithmic perception of frequency motivates the use of a time-frequency representation with a logarithmic frequency axis labeled by the pitches of the equal-tempered scale. To derive such a representation from a given spectrogram representation, the basic idea is to assign each spectral coefficient $\mathcal{X}(m, k)$ to the pitch with center frequency that is closest to the frequency $F_{\text{coef}}(k)$. More precisely, we define for each pitch $p \in [0 : 127]$ the set

$$P(p) := \{k \in [0 : K] : F_{\text{pitch}}(p - 0.5) \leq F_{\text{coef}}(k) < F_{\text{pitch}}(p + 0.5)\}. \tag{8}$$

From this, we obtain a log-frequency spectrogram $\mathcal{Y}_{\text{LF}} : \mathbb{Z} \times [0 : 127] \to \mathbb{R}_{\geq 0}$ defined by

$$\mathcal{Y}_{\text{LF}}(m, p) := \sum_{k \in P(p)} |\mathcal{X}(m, k)|^2. \tag{9}$$

By this definition, the frequency axis is partitioned logarithmically and labeled linearly according to MIDI pitches.

Homework Exercise 3

(a) Compute the center frequencies $F_{\text{pitch}}(p)$ for $p = 68$, $p = 69$, and $p = 70$.
(b) Compute the cutoff frequencies $F_{\text{pitch}}(p - 0.5)$ and $F_{\text{pitch}}(p + 0.5)$ of the frequency band corresponding to pitch $p = 69$.
(c) Using $F_s = 22050$ and $N = 4096$, determine the set $P(p) \subseteq [0 : K]$ for $p = 69$.
(d) Also compute $P(p)$ for $p = 57$, $p = 45$, and $p = 33$.
(e) Please explain with your own words why the definition of $\mathcal{Y}_{\text{LF}}(m, p)$ in (9) may be problematic for small values of pitch p? How is the size of the set $P(p)$ influenced? Support your answer with a brief example.

Lab Experiment 3

- Use the file `Scale_Cmajor_Piano.wav` to define a signal x.
- Compute the STFT and the spectrogram as above using a Hann window of size $N = 4096$ and a hop size $H = 2048$. In the following you also need the information contained in the frequency vector `F`.
- Compute the log-frequency spectrogram $\mathcal{Y}_{\mathrm{LF}}$ as defined in (9).
- Visualize the log-frequency spectrogram with the axes given in seconds and MIDI pitch, respectively.
- Use log-compression as in (5) to enhance the visualization.
- Play around with different parameter settings for N and H. Also, try out some other audio files.

4 Chroma Features

The human perception of pitch is periodic in the sense that two pitches are perceived as similar in "color" (playing a similar harmonic role) if they differ by one or several octaves (where, in our scale, an octave is defined as the distance of 12 pitches). For example, the pitches $p = 60$ and $p = 72$ are one octave apart, and the pitches $p = 57$ and $p = 81$ are two octaves apart. A pitch can be separated into two components, which are referred to as *tone height* and *chroma*. The tone height refers to the octave number and the chroma to the respective pitch spelling attribute. In Western music notation, the 12 pitch attributes are given by the set $\{\mathrm{C}, \mathrm{C}^\sharp, \mathrm{D}, \ldots, \mathrm{B}\}$. Enumerating the chroma values, we identify this set with $[0:11]$ where $c = 0$ refers to chroma C, $c = 1$ to $\mathrm{C}^\sharp$, and so on. A *pitch class* is defined as the set of all pitches that share the same chroma. For example, the pitch class that corresponds to the chroma $c = 0$ (C) consists of the set $\{0, 12, 24, 36, 48, 60, 72, 84, 96, 108, 120\}$ (which are the musical notes $\{\ldots, \mathrm{C0}, \mathrm{C1}, \mathrm{C2}, \mathrm{C3}\ldots\}$).

The main idea of *chroma features* is to aggregate all spectral information that relates to a given pitch class into a single coefficient. Given a pitch-based log-frequency spectrogram $\mathcal{Y}_{\mathrm{LF}} : \mathbb{Z} \times [0:127] \to \mathbb{R}_{\geq 0}$ as defined in (9), a chroma representation or *chromagram* $\mathbb{Z} \times [0:11] \to \mathbb{R}_{\geq 0}$ can be derived by summing up all pitch coefficients that belong to the same chroma:

$$\mathcal{C}(m, c) := \sum_{\{p \in [0:127] \,|\, p \bmod 12 = c\}} \mathcal{Y}_{\mathrm{LF}}(m, p) \tag{10}$$

for $c \in [0:11]$.

Lab Experiment 4

- Derive the chroma representation $\mathcal{C}$ from the log-frequency spectrogram as computed in the last exercise.
- Visualize the chroma representation with the axes given in seconds and chroma indices, respectively.
- Try to explain what you see in the chroma visualization.
- Also play around with different parameter settings for N and H and try out some other audio files.

5 Further Notes

In this lab course, we have addressed the issue on computing and visualizing a discrete version of the short-time Fourier transform. Furthermore, we have used this transform to derive audio

features that can be used for analyzing music signals. If you are interested in the topic of music processing, you may attend the lecture *Music Processing – Analysis*, which is held in the winter semester. For an overview, please have a look at

`http://www.audiolabs-erlangen.com/m_mueller/teaching/ws2013_mpa/`

The basic definitions and main properties of the Fourier transform are covered in most introductory books on signal processing. As example references, we want to mention the classical textbook on *Signals and Systems* by Oppenheim et al. [1] or the book on *Digital Signal Processsing* by Proakis and Manolakis [2]. As for the notation, we have closely followed [3, Section 2.2]. An entertaining and non-technical introduction to the main ideas of time-frequency analysis can be found in the book *The World According to Wavelets* by Hubbard [4]. Also Wikipedia contains many interesting articles.

As one important audio feature, we have discussed the concept of chroma features. These features show a high degree of robustness to variations in timbre and closely correlate to the musical aspect of harmony. This is the reason why chroma-based audio features, sometimes also referred to as *pitch class profiles*, have become a major tool for processing and analyzing music data [5, 6, 3, 7]. Besides music synchronization and alignment [8, 9, 3], chroma features have de facto become the standard for many other tasks such as chord recognition [10, 11, 12], audio structure analysis [13], or content-based audio retrieval such as cover song and version identification [14, 15]. This is only a small selection of research problems that are tackled in the area known as *Music Information Retrieval* (MIR) or *Music Processing*.

There are many ways for computing chroma-based audio features. In this lab course, we have studied a strategy based on a short-time Fourier transforms in combination with pooling strategies [5, 6]. To obtain better frequency resolutions, one popular alternative to using a single spectrogram is to construct a multirate bank of bandpass filters, each filter corresponding to a pitch with an appropriately tuned bandwidth [3, Section 3.1]. Various implementations of chroma-based audio features are publicly available such as the chroma-variants by Ellis[1] and the Chroma-Toolbox[2] including extractors for a variety of pitch- and chroma-based audio features [16]. In conclusion, one should keep in mind that there is no "best" chroma variant and that the results of a specific music analysis task may crucially depend on the used chroma type.

References

[1] A. V. Oppenheim, A. S. Willsky, and H. Nawab, *Signals and Systems*. Prentice Hall, 1996.

[2] J. G. Proakis and D. G. Manolakis, *Digital Signal Processsing*. Prentice Hall, 1996.

[3] M. Müller, *Information Retrieval for Music and Motion*. Springer Verlag, 2007.

[4] B. B. Hubbard, *The world according to wavelets*. AK Peters, Wellesley, Massachusetts, 1996.

[5] M. A. Bartsch and G. H. Wakefield, "Audio thumbnailing of popular music using chroma-based representations," *IEEE Transactions on Multimedia*, vol. 7, no. 1, pp. 96–104, 2005.

[6] E. Gómez, "Tonal description of music audio signals," Ph.D. dissertation, UPF Barcelona, 2006.

[7] M. Müller, D. P. W. Ellis, A. Klapuri, and G. Richard, "Signal processing for music analysis," *IEEE Journal on Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1088–1110, 2011. [Online]. Available: http://dx.doi.org/10.1109/JSTSP.2011.2112333

[1] `http://www.ee.columbia.edu/~dpwe/resources/matlab/chroma-ansyn/`

[2] `http://www.mpi-inf.mpg.de/resources/MIR/chromatoolbox/`

[8] N. Hu, R. B. Dannenberg, and G. Tzanetakis, "Polyphonic audio matching and alignment for music retrieval," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, USA, 2003.

[9] C. Joder, S. Essid, and G. Richard, "A comparative study of tonal acoustic features for a symbolic level music-to-score alignment," in *Proceedings of the 35nd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Dallas, USA, 2010.

[10] J. P. Bello and J. Pickens, "A robust mid-level representation for harmonic content in music signals," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, UK, 2005, pp. 304–311.

[11] T. Fujishima, "Realtime chord recognition of musical sound: A system using common lisp music," in *Proceedings of the International Computer Music Conference (ICMC)*, Beijing, 1999, pp. 464–467.

[12] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1280–1289, 2010.

[13] J. Paulus, M. Müller, and A. Klapuri, "Audio-based music structure analysis," in *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, Utrecht, Netherlands, 2010, pp. 625–636.

[14] F. Kurth and M. Müller, "Efficient Index-Based Audio Matching," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 382–395, Feb. 2008. [Online]. Available: http://dx.doi.org/10.1109/TASL.2007.911552

[15] J. Serrà, E. Gómez, P. Herrera, and X. Serra, "Chroma binary similarity and local alignment applied to cover song identification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, pp. 1138–1151, 2008.

[16] M. Müller and S. 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)*, Miami, USA, 2011, pp. 215–220.