CHROMA TOOLBOX: MATLAB IMPLEMENTATIONS FOR EXTRACTING VARIANTS OF CHROMA-BASED AUDIO FEATURES

Meinard Müller
Saarland University
and MPI Informatik
meinard@mpi-inf.mpg.de

Sebastian Ewert
Computer Science III
University of Bonn
ewerts@iai.uni-bonn.de

ABSTRACT

Chroma-based audio features, which closely correlate to the aspect of harmony, are a well-established tool in processing and analyzing music data. There are many ways of computing and enhancing chroma features, which results in a large number of chroma variants with different properties. In this paper, we present a chroma toolbox [13], which contains MATLAB implementations for extracting various types of recently proposed pitch-based and chroma-based audio features. Providing the MATLAB implementations on a well-documented website under a GNU-GPL license, our aim is to foster research in music information retrieval. As another goal, we want to raise awareness that there is no single chroma variant that works best in all applications. To this end, we discuss two example applications showing that the final music analysis result may crucially depend on the initial feature design step.

1. INTRODUCTION

It is a well-known phenomenon that human perception of pitch is periodic in the sense that two pitches are perceived as similar in “color” if they differ by an octave. Based on this observation, a pitch can be separated into two components, which are referred to as *tone height* and *chroma*, see [19]. Assuming the equal-tempered scale, the chromas correspond to the set $\{\mathrm{C}, \mathrm{C}^\sharp, \mathrm{D}, \ldots, \mathrm{B}\}$ that consists of the twelve pitch spelling attributes [1] as used in Western music notation. Thus, a chroma feature is represented by a 12-dimensional vector $x = (x(1), x(2), \ldots, x(12))^T$, where $x(1)$ corresponds to chroma C, $x(2)$ to chroma $\mathrm{C}^\sharp$, and so on. In the feature extraction step, a given audio signal is converted into a sequence of chroma features each expressing how the short-time energy of the signal is spread over the twelve chroma bands.

[1] Note that in the equal-tempered scale different pitch spellings such $\mathrm{C}^\sharp$ and $\mathrm{D}^\flat$ refer to the same chroma.

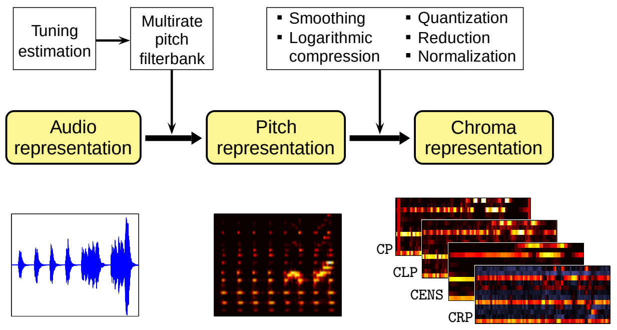

Figure 1. Overview of the feature extraction pipeline.

Identifying pitches that differ by an octave, chroma 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, are a well-established tool for processing and analyzing music data [1, 5, 12]. For example, basically every chord recognition procedure relies on some kind of chroma representation [2, 4, 11]. Also, chroma features have become the de facto standard for tasks such as music synchronization and alignment [7, 8, 12], as well as audio structure analysis [16]. Finally, chroma features have turned out to be a powerful mid-level feature representation in content-based audio retrieval such as cover song identification [3, 18] or audio matching [10, 15].

There are many ways for computing chroma-based audio features. For example, the conversion of an audio recording into a chroma representation (or chromagram) may be performed either by using short-time Fourier transforms in combination with binning strategies [1] or by employing suitable multirate filter banks [12]. Furthermore, the properties of chroma features can be significantly changed by introducing suitable pre- and post-processing steps modifying spectral, temporal, and dynamical aspects. This leads to a large number of chroma variants, which may show a quite different behavior in the context of a specific music analysis scenario.

In this paper, we introduce a chroma toolbox, which has recently been released under a GNU-GPL license, see [13]. This well-documented toolbox contains MATLAB implementations for extracting various types of recently introduced pitch-based and chroma-based audio features (referred to as `Pitch`, `CP`, `CLP`, `CENS`, and `CRP`), see also Figure 1 for an overview. In Section 2, we give a short summary on how the various feature types are computed while discussing the role of the most important parameters that can be used to modify the features' characteristics. Then, in Section 3, we describe the functions of the toolbox for feature extraction, visualization, and post-processing. One particular goal of this paper is to emphasize the importance of the feature design step by showing that the results of a specific music analysis task may crucially depend on the used chroma type. To this end, we discuss in Section 4 two illustrative example applications, namely chord recognition and audio matching.

2. FEATURE EXTRACTION

In this section, we give an overview on how the various feature types contained in the chroma toolbox are computed. As illustration, Figure 3 shows the resulting feature representations for an audio recording of the first six measures of Op. 100, No. 2 by Friedrich Burgmüller.

2.1 Pitch Representation

As basis for the chroma feature extraction, we first decompose a given audio signal into 88 frequency bands with center frequencies corresponding to the pitches $A0$ to $C8$ (MIDI pitches $p = 21$ to $p = 108$). To obtain a sufficient spectral resolution for the lower frequencies, one either needs a low sampling rate or a large temporal window. In our toolbox, we employ a constant Q multirate filter bank using a sampling rate of 22050 Hz for high pitches, 4410 Hz for medium pitches, and 882 Hz for low pitches, see [12] for details. The employed pitch filters possess a relatively wide passband, while still properly separating adjacent notes thanks to sharp cutoffs in the transition bands, see Figure 2. Actually, the pitch filters are robust to deviations of up to ± 25 cents [2] from the respective note's center frequency. To avoid large phase distortions, we use forward-backward filtering such that the resulting output signal has precisely zero phase distortion and a magnitude modified by the square of the filter's magnitude response, see [17].

[2] The *cent* is a logarithmic unit to measure musical intervals. The semitone interval of the equally-tempered scale equals 100 cents.

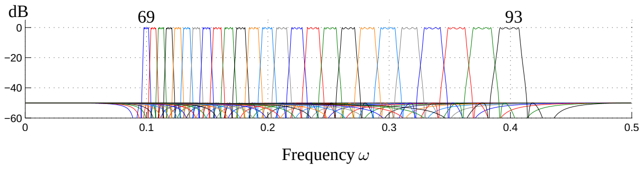

Figure 2. Magnitude responses in dB for some of the filters of the multirate pitch filter bank. The shown filters correspond to MIDI pitches $p \in [69 : 93]$ (with respect to the sampling rate 4410 Hz).

In the next step, for each of the 88 pitch subbands, we compute the short-time mean-square power (i. e., the samples of each subband output are squared) using a window of a fixed length and an overlap of 50 %. For example, using a window length of 200 milliseconds leads to a feature rate of 10 Hz (10 features per second). The resulting features, which we denote as `Pitch`, measure the short-time energy content of the audio signal within each pitch subband. We refer to Figure 3c for an illustration and to [12] for details.

2.2 Tuning

To account for the global tuning of a recording, one needs to suitably shift the center frequencies of the subband-filters of the multirate filter bank. To this end, we compute an average spectrogram vector and derive an estimate for the tuning deviation by simulating the filterbank shifts using weighted binning techniques similar to [5]. In our toolbox, we have pre-computed six different multirate filter banks corresponding to a shift of $\sigma \in \left\{0, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}\right\}$ semitones, respectively. From these filter banks, the most suitable one is chosen according to the estimated tuning deviation.

2.3 CP Feature

From the pitch representation, one obtains a chroma representation simply by adding up the corresponding values that belong to the same chroma. For example, to compute the entry corresponding to chroma C, one adds up values corresponding to the musical pitches C1, C2, …, C8 (MIDI pitches $p = 24, 36, \ldots, 108$). For each window, this yields a 12-dimensional vector $x = (x(1), x(2), \ldots, x(12))^T$, where $x(1)$ corresponds to chroma C, $x(2)$ to chroma C$^\sharp$, and so on. The resulting features are referred to as *Chroma-Pitch* and denoted by `CP`, see Figure 3d.

2.4 Normalization

To achieve invariance in dynamics, one can normalize the features with respect to some suitable norm. In the following, we only consider the ℓ^p-norm defined by $\|x\|_p := \left(\sum_{i=1}^{12} |x(i)|^p\right)^{1/p}$ for a given chroma vector $x = (x(1), x(2), \ldots, x(12))^T$ and some natural number $p \in \mathbb{N}$.

To avoid random energy distributions occurring during passages of very low energy (e. g., passages of silence before the actual start of the recording or during long pauses), we replace a chroma vector x by the uniform vector of norm one in case $\|x\|_p$ falls below a certain threshold. Note that the case $p = 2$ yields the Euclidean norm and the case $p = 1$ the Manhattan norm. If not specified otherwise, all chroma vectors to be considered are normalized with respect to the Euclidean norm, see also Figure 3e.

2.5 CLP Features

To account for the logarithmic sensation of sound intensity [20], one often applies a logarithmic amplitude compression when computing audio features. To this end, each energy values e of the pitch representation is replaced by the value $\log(\eta \cdot e + 1)$, where η is a suitable positive constant. Then, the chroma values are computed as explained in Section 2.3. The resulting features, which depend on the compression parameter η, are referred to as *Chroma-Log-Pitch* and denoted by $\texttt{CLP}[\eta]$, see Figure 3f. Note that a similar flattening effect can be achieved by spectral whitening techniques, where the pitch subbands are normalized according to short-time variances in the subbands [5, 9].

2.6 CENS Features

Adding a further degree of abstraction by considering short-time statistics over energy distributions within the chroma bands, one obtains `CENS` (Chroma Energy Normalized Statistics) features, which constitute a family of scalable and robust audio features. These features have turned out to be very useful in audio matching and retrieval applications [10, 15]. In computing `CENS` features, each chroma vector is first normalized with respect to the ℓ^1-norm thus expressing relative energy distribution. Then, a quantization is applied based on suitably chosen thresholds. Here, choosing thresholds in a logarithmic fashion introduces some kind of logarithmic compression as above, see [15] for details. In a subsequent step, the features are further smoothed over a window of length $w \in \mathbb{N}$ and downsampled by a factor of d, see Section 2.8. The resulting features are normalized with respect to the ℓ^2-norm and denoted by $\texttt{CENS}^w_d$, see also Figure 3g and Figure 3h for illustrations.

2.7 CRP Features

To boost the degree of timbre invariance, a novel family of chroma-based audio features has been introduced in [14]. The general idea is to discard timbre-related information as is captured by the lower mel-frequency cepstral coefficients (MFCCs). Starting with the `Pitch` features, one first applies a logarithmic compression and transforms the logarithmized pitch representation using a DCT. Then, one

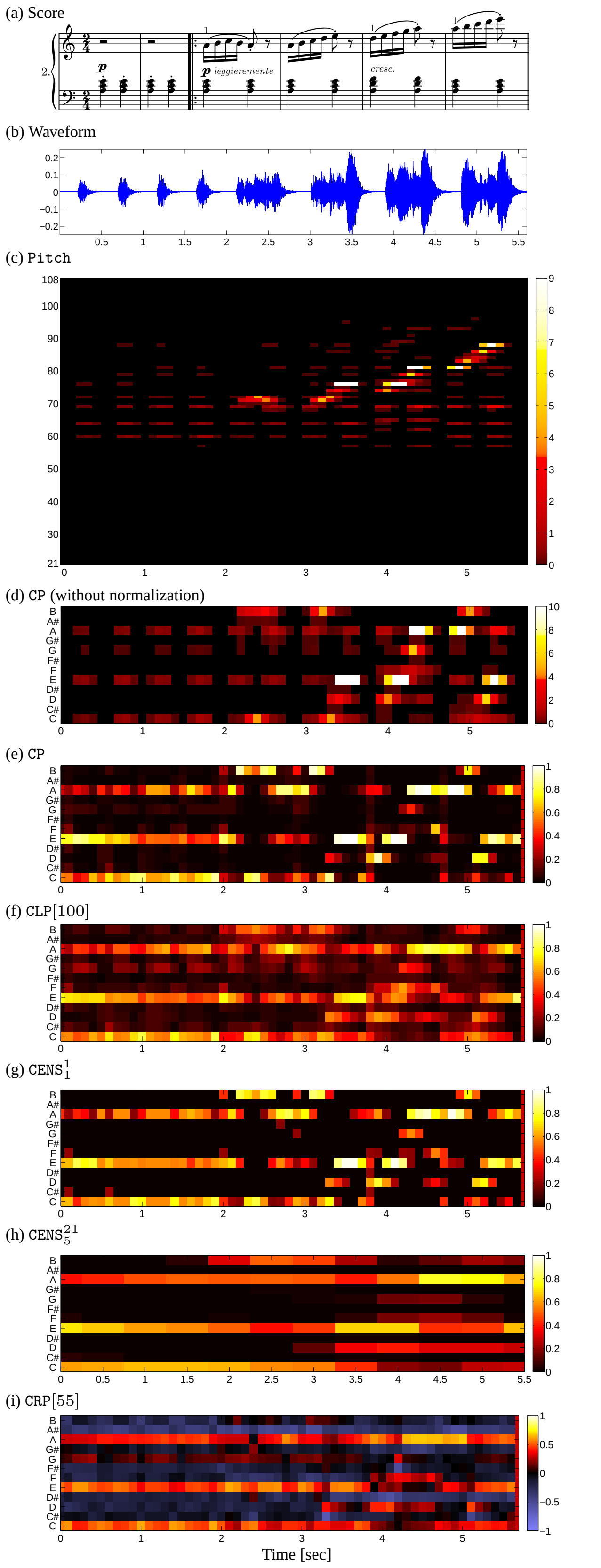

Figure 3. Score and various feature representations for an audio recording of the first four measures of Op. 100, No. 2 by Friedrich Burgmüller.

Filename	Main parameters	Description
`wav_to_audio.m`	–	Import of WAV files and conversion to expected audio format.
`estimateTuning.m`	pitchRange	Estimation of the filterbank shift parameter σ.
`audio_to_pitch_via_FB.m`	winLenSTMSP	Extraction of pitch features from audio data.
`pitch_to_chroma.m`	applyLogCompr, factorLogCompr $\widehat{=}\ \eta$	Derivation of `CP` and `CLP` features from `Pitch` features.
`pitch_to_CENS.m`	winLenSmooth $\widehat{=}\ w$, downsampSmooth $\widehat{=}\ d$	Derivation of `CENS` features from `Pitch` features.
`pitch_to_CRP.m`	coeffsToKeep $\widehat{=}\ n$, factorLogCompr $\widehat{=}\ \eta$	Derivation of `CRP` features from `Pitch` features.
`smoothDownsampleFeature.m`	winLenSmooth $\widehat{=}\ w$, downsampSmooth $\widehat{=}\ d$	Post-processing of features: smoothing and downsampling.
`normalizeFeature.m`	p	Post-processing of features: ℓ^p-normalization (default: $p = 2$).
`visualizePitch.m`	featureRate	Visualization of pitch features.
`visualizeChroma.m`	featureRate	Visualization of chroma features.
`visualizeCRP.m`	featureRate	Specialized version of visualizeChroma for `CRP` features.
`generateMultiratePitchFilterbank.m`	–	Generation of filterbanks (used in `audio_to_pitch_via_FB.m`).

Table 1. Overview of the MATLAB functions contained in the chroma toolbox [13].

only keeps the upper coefficients of the resulting pitch-frequency cepstral coefficients (PFCCs), applies an inverse DCT, and finally projects the resulting pitch vectors onto 12-dimensional chroma vectors, which are then normalized with respect to the ℓ^2-norm. These vectors are referred to as CRP (Chroma DCT-Reduced log Pitch) features. The upper coefficients to be kept are specified by a parameter $n \in [1 : 120]$. As reported in [14], the parameter $n = 55$ yields good results and constitutes our default stetting. The resulting features are denoted by $\texttt{CRP}[n]$, see Figure 3i. Note that opposed to the previously introduced chroma variants, CRP features may have negative entries.

2.8 Smoothing

As already mentioned in Section 2.6, one can further process the various chroma variants by applying smoothing and downsampling operations. For example, subsequent vectors of a feature sequences can be averaged using a sliding window of size w (given in frames) and then downsampled by a factor d. Starting with `CENS`, `CP`, $\texttt{CLP}[\eta]$, and $\texttt{CRP}[n]$, the resulting features are denoted by $\texttt{CENS}^w_d$, $\texttt{CP}^w_d$, $\texttt{CLP}[\eta]^w_d$, and $\texttt{CRP}[n]^w_d$, respectively. Even though being a simple strategy, smoothing can have a significant impact on the features' behavior within a music analysis tasks. For example, as reported in [15], the temporal blurring of CENS features makes audio matching more robust to local tempo variations. Furthermore, using the parameters w and d, one obtains a computationally inexpensive procedure to simulate tempo changes on the feature level. We illustrate this by means of a concrete example. Suppose, we start with a chroma representation having a feature rate of 10 Hz. Then using $w = 41$ and $d = 10$, one obtains one chroma vector per second, each covering roughly 4100 ms of the original audio signal. Now, using $w = 53$ (instead of $w = 41$) and $d = 13$ (instead of $d = 10$) results in a temporally scaled version of the features sequence simulating a tempo change of $10/13 \approx 0.77$. Such tempo change strategies have been applied successfully in the context of audio indexing [10].

3. TOOLBOX

The feature extraction components as described in Section 2 form the core of our chroma toolbox, which is freely available at the well-documented website [13] under a GNU-GPL license. Table 1 gives an overview of the main MATLAB functions along with the most important parameters. Note that there are many more parameters not discussed in this paper. However, for all parameters there are default settings so that none of the parameters need to be specified by the user.

To demonstrate how our toolbox can be applied, we now discuss the code example[3] shown in Table 2. Our example starts with a call to the function `wav_to_audio`, which is a simple wrapper around MATLAB's `wavread.m` and converts the input WAV file into a mono version at a sampling rate of 22050 Hz. Furthermore, the struct `sideinfo` is returned containing meta information about the WAV file. In line 3, the audio data is processed by `estimateTuning`, which computes an appropriate filter bank shift σ for the recording. Next, in lines 5–9, `Pitch` features are computed. Here, the struct `paramPitch` is used to pass optional parameters to the feature extraction function. If some parameters or the whole struct are not set manually, then meaningful default settings are used. This is a general principle throughout the toolbox. For the pitch computation, `winLenSTMSP` specifies the window length in samples. Here, 4410 together with a sampling frequency of 22050 Hz results in a window length corresponding to 200ms of audio. Using half-overlapped windows leads to a feature rate of 10 Hz. The filterbank shift is specified in line 6 using the output of `estimateTuning`. Furthermore, an internal visualization is activated using the parameter `visualize`. Then, a call to `audio_to_pitch_via_FB` results in a $120 \times N$-matrix `f_pitch` that constitutes the `Pitch` features, where N is the number of time frames and the first dimension corresponds to MIDI pitches. Actually, only the bands corresponding

[3] This example is also contained in the toolbox as function `demoChromaToolbox.m`.

```
1   filename=’Systematic_Chord-C-Major_Eight-Instruments.wav’;
2   [f_audio,sideinfo]=wav_to_audio(’’,’data_WAV/’,filename);
3   shiftFB=estimateTuning(f_audio);
4
5   paramPitch.winLenSTMSP=4410;
6   paramPitch.shiftFB=shiftFB;
7   paramPitch.visualize=1;
8   [f_pitch,sideinfo]=...
9      audio_to_pitch_via_FB(f_audio,paramPitch,sideinfo);
10
11  paramCP.applyLogCompr=0;
12  paramCP.visualize=1;
13  paramCP.inputFeatureRate=sideinfo.pitch.featureRate;
14  [f_CP,sideinfo]=pitch_to_chroma(f_pitch,paramCP,sideinfo);
15
16  paramCLP.applyLogCompr=1;
17  paramCLP.factorLogCompr=100;
18  paramCLP.visualize=1;
19  paramCLP.inputFeatureRate=sideinfo.pitch.featureRate;
20  [f_CLP,sideinfo]=pitch_to_chroma(f_pitch,paramCLP,sideinfo);
21
22  paramCENS.winLenSmooth=21;
23  paramCENS.downsampSmooth=5;
24  paramCENS.visualize=1;
25  paramCENS.inputFeatureRate=sideinfo.pitch.featureRate;
26  [f_CENS,sideinfo]=pitch_to_CENS(f_pitch,paramCENS,sideinfo);
27
28  paramCRP.coeffsToKeep=[55:120];
29  paramCRP.visualize=1;
30  paramCRP.inputFeatureRate=sideinfo.pitch.featureRate;
31  [f_CRP,sideinfo]=pitch_to_CRP(f_pitch,paramCRP,sideinfo);
32
33  paramSmooth.winLenSmooth=21;
34  paramSmooth.downsampSmooth=5;
35  paramSmooth.inputFeatureRate=sideinfo.CRP.featureRate;
36  [f_CRPSmoothed,featureRateSmoothed]=...
37     smoothDownsampleFeature(f_CRP,paramSmooth);
38  parameterVis.featureRate=featureRateSmoothed;
39  visualizeCRP(f_CRPSmoothed,parameterVis);
```

Table 2. Code example.

to MIDI pitches 21 to 108 are computed and the values of the other bands are set to zero. Furthermore, details on the feature configuration are appended to the `sideinfo` struct. Using `sideinfo` to store all relevant meta information related to the feature processing pipeline constitutes a second general principle in our toolbox.

In lines 11–31, various chroma representations are derived from the pitch features. First, in lines 11–14, CP features are computed. Then, activating the logarithmic compression using `applyLogCompr`, $\mathtt{CLP}[100]$ features are computed in lines 16–20. The compression level is specified in line 17 by the parameter `factorLogCompr`, which corresponds to the parameter η introduced in Section 2.5. Next, in lines 22–26, $\mathtt{CENS}^{21}_{5}$ features are computed. Here, the parameters `winLenSmooth` and `downsampSmooth` correspond to the parameters w and d explained in Section 2.8, respectively. Finally, in lines 28–31, $\mathtt{CRP}[55]$ features are computed, where the parameter n of Section 2.7 corresponds to the lower bound of the range specified by `coeffsToKeep`, see line 28. Finally, the use of the function `smoothDownsampleFeature` is demonstrated, where in lines 33–34 the parameters w and d are specified as for the CENS computation. At the end of our example, we visualize the smoothed CRP features using the function `visualizeCRP`.

4. ILLUSTRATING APPLICATIONS

To demonstrate the importance of the feature design step, we now discuss the various chroma variants within two different music analysis scenarios. Here, rather than commending a specific feature type, our goal is to show how different feature variants and parameter settings may crucially influence the final analysis results.

4.1 Chord Recognition

The computer-based harmonic analysis of music recordings with the goal to automatically extract chord labels directly from the given audio material constitutes a major task in music information retrieval [2, 4, 11]. In most automated chord recognition procedures, the given music recording is first converted into a sequence of chroma-based audio features and then pattern matching techniques are applied to map the chroma features to chord labels.

We now demonstrate by a small experiment, how the final recognition rates substantially depend on the underlying chroma representation and parameters that control temporal and spectral aspects. To this end, we revert to three different pattern matching techniques. The first two approaches are simple template-based approaches, referred to as $\mathtt{T}^{\mathrm{b}}$ and $\mathtt{T}^{\mathrm{a}}$, where the first approach uses data-independent binary templates and the second one data-dependent average templates. As third approach, we employ hidden Markov models denoted by HMM. Using the annotated Beatles dataset as described in [6], which consists of 180 Beatles songs, we computed recognition rates based on conventional F-measures using 3-fold cross validation. Figure 4 shows the recognition rates for the three pattern matching techniques in combination with different chroma variants.

As these experimental results indicate, the used chroma representation can have a significant influence on the chord recognition accuracy. In particular, a logarithmic compression step in the chroma extraction turns out to be crucial. Furthermore, the results reveal that temporal feature smoothing plays an important role in chord recognition–in particular for recognizers that work in a purely frame-

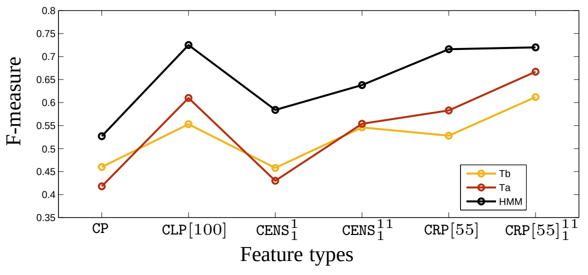

Figure 4. Dependency of the recognition rates (F-measures) of different chord recognition procedures on the used chroma variant (using a Beatles dataset and 3-fold cross validation).

wise fashion. Here, note that the Viterbi decoding in the HMM-based recognizer already introduces a different kind of smoothing in the classification stage so that feature smoothing has a less significant impact in this case.

4.2 Audio Matching

As second application scenario, we consider the task of *audio matching* with the goal to automatically retrieve all fragments from all recordings within a large audio collection that musically correspond to a given query audio clip [15]. In this task, one challenge is to cope with variations in timbre and instrumentation as they appear in different interpretations, cover songs, and arrangements of a piece of music. In a typical procedure for audio matching, the query Q as well as each database recording D are first converted into chroma feature sequences $X(Q)$ and $X(D)$, respectively. Then, a local variant of dynamic time warping is used to locally compare the query sequence $X(Q)$ with the database sequence $X(D)$ yielding a distance function Δ. Each local minimum of Δ close to zero indicates a fragment within the database recording that is close to the given query, see [14] for details.

In view of this matching application, the following two properties of Δ are of crucial importance. On the one hand, the semantically correct matches should correspond to local minima of Δ close to zero thus avoiding false negatives. On the other hand, Δ should be well above zero outside a neighborhood of the desired local minima thus avoiding false positives. In view of these requirements, the used chroma variant plays a major role. As an illustrative example, we consider a recording by Yablonsky of Shostakovich's Waltz No. 2 from the *Suite for Variety Orchestra No. 1*, which is used as the database recording. The theme of this piece occurs four times played in four different instrumentations (clarinet, strings, trombone, tutti). Denoting the four occurrences by E_1, E_2, E_3, and E_4 and using E_3 as the query, Figure 5 shows several distance functions based on different chroma variants. Note that one expects four local minima. Using conventional chroma features such as CP, the expected local minima are not significant or not even existing. However, using the chroma variant CRP$[55]$, one obtains for all four true matches concise local minima, see the black curve of Figure 5. For a detailed discussion, we refer to [14].

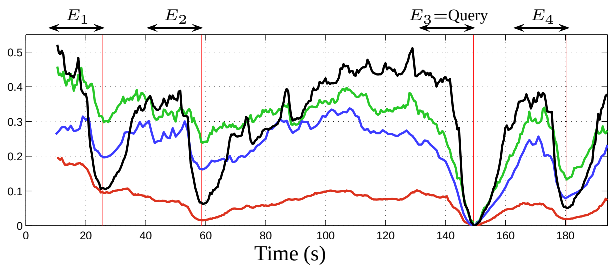

Figure 5. Several distance functions shown for the Yablonsky recording of the Shostakovich's Waltz No. 2 from the *Suite for Variety Orchestra No. 1* using the excerpt E_3 as query. The following feature types used: CP (green), CLP$[100]$ (red), CENS$^{41}_{10}$ (blue) and CRP$[55]$ (black). For the query, there are 4 annotated excerpts (true matches).

Acknowledgement. This work has been supported by the Cluster of Excellence on Multimodal Computing and Interaction at Saarland University and the German Research Foundation (DFG CL 64/6-1).

5. REFERENCES

[1] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, February 2005.

[2] Juan Pablo Bello and Jeremy 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.

[3] Daniel P. W. Ellis and Graham. E. Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, Honolulu, Hawaii, USA, April 2007.

[4] Takuya Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proc. ICMC*, pages 464–467, Beijing, 1999.

[5] Emilia Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, UPF Barcelona, 2006.

[6] Christopher Harte, Mark Sandler, Samer Abdallah, and Emilia Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, GB, 2005.

[7] Ning Hu, Roger Dannenberg, and George 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, US, October 2003.

[8] Cyril Joder, Slim Essid, and Gaël 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.

[9] Anssi Klapuri. Multipitch analysis of polyphonic music and speech signals using an auditory model. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):255–266, 2008.

[10] Frank Kurth and Meinard Müller. Efficient index-based audio matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):382–395, February 2008.

[11] Matthias Mauch and Simon Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1280–1289, 2010.

[12] Meinard Müller. *Information Retrieval for Music and Motion*. Springer Verlag, 2007.

[13] Meinard Müller and Sebastian Ewert. Chroma toolbox. `http://www.mpi-inf.mpg.de/resources/MIR/chromatoolbox/`, retrieved 01.04.2011.

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

[15] Meinard Müller, Frank Kurth, and Michael Clausen. Audio matching via chroma-based statistical features. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 288–295, 2005.

[16] Jouni Paulus, Meinard Müller, and Anssi Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 625–636, Utrecht, Netherlands, 2010.

[17] John G. Proakis and Dimitris G. Manolakis. *Digital Signal Processsing*. Prentice Hall, 1996.

[18] 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*, 16:1138–1151, October 2008.

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

[20] Eberhard Zwicker and Hugo Fastl. *Psychoacoustics, facts and models*. Springer Verlag, New York, NY, US, 1990.