Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2007, Article ID 89686, 18 pages
doi:10.1155/2007/89686

Research Article

Towards Structural Analysis of Audio Recordings in the Presence of Musical Variations

Meinard Müller and Frank Kurth

Department of Computer Science III, University of Bonn, Römerstraße 164, 53117 Bonn, Germany

Received 1 December 2005; Revised 24 July 2006; Accepted 13 August 2006

Recommended by Ichiro Fujinaga

One major goal of structural analysis of an audio recording is to automatically extract the repetitive structure or, more generally, the musical form of the underlying piece of music. Recent approaches to this problem work well for music, where the repetitions largely agree with respect to instrumentation and tempo, as is typically the case for popular music. For other classes of music such as Western classical music, however, musically similar audio segments may exhibit significant variations in parameters such as dynamics, timbre, execution of note groups, modulation, articulation, and tempo progression. In this paper, we propose a robust and efficient algorithm for audio structure analysis, which allows to identify musically similar segments even in the presence of large variations in these parameters. To account for such variations, our main idea is to incorporate invariance at various levels simultaneously: we design a new type of statistical features to absorb microvariations, introduce an enhanced local distance measure to account for local variations, and describe a new strategy for structure extraction that can cope with the global variations. Our experimental results with classical and popular music show that our algorithm performs successfully even in the presence of significant musical variations.

1. INTRODUCTION

Content-based document analysis and efficient audio browsing in large music databases has become an important issue in music information retrieval. Here, the automatic annotation of audio data by descriptive high-level features as well as the automatic generation of crosslinks between audio excerpts of similar musical content is of major concern. In this context, the subproblem of *audio structure analysis* or, more specifically, the automatic identification of musically relevant repeating patterns in some audio recording has been of considerable research interest; see, for example, [1–7]. Here, the crucial point is the notion of *similarity* used to compare different audio segments, because such segments may be regarded as musically similar in spite of considerable variations in parameters such as dynamics, timbre, execution of note groups (e.g., grace notes, trills, arpeggios), modulation, articulation, or tempo progression. In this paper, we introduce a robust and efficient algorithm for the structural analysis of audio recordings, which can cope with significant variations in the parameters mentioned above including local tempo deformations. In particular, we introduce a new class of robust audio features as well as a new class of similarity measures that yield a high degree of invariance as needed to compare musically similar segments. As opposed to previous approaches, which mainly deal with popular music and assume constant tempo throughout the piece, we have applied our techniques to musically complex and versatile Western classical music. Before giving a more detailed overview of our contributions and the structure of this paper (Section 1.3), we summarize a general strategy for audio structure analysis and introduce some notation that is used throughout this paper (Section 1.1). Related work will be discussed in Section 1.2.

1.1. General strategy and notation

To extract the repetitive structure from audio signals, most of the existing approaches proceed in four steps. In the first step, a suitable high-level representation of the audio signal is computed. To this end, the audio signal is transformed into a sequence $\vec{V} := (\vec{v}^1, \vec{v}^2, \ldots, \vec{v}^N)$ of feature vectors $\vec{v}^n \in \mathcal{F}$, $1 \leq n \leq N$. Here, $\mathcal{F}$ denotes a suitable feature space, for example, a space of spectral, MFCC, or chroma vectors.

Based on a suitable similarity measure $d : \mathcal{F} \times \mathcal{F} \to \mathbb{R}_{\geq 0}$, one then computes an N-square *self-similarity*[1] *matrix* $\mathcal{S}$ defined by $\mathcal{S}(n, m) := d(\vec{v}^n, \vec{v}^m)$, effectively comparing all feature vectors $\vec{v}^n$ and $\vec{v}^m$ for $1 \leq n, m \leq N$ in a pairwise fashion. In the third step, the path structure is extracted from the resulting self-similarity matrix. Here, the underlying principle is that similar segments in the audio signal are revealed as paths along diagonals in the corresponding self-similarity matrix, where each such path corresponds to a pair of similar segments. Finally, in the fourth step, the global repetitive structure is derived from the information about pairs of similar segments using suitable clustering techniques.

To illustrate this approach, we consider two examples, which also serve as running examples throughout this paper. The first example, for short referred to as *Brahms example*, consists of an Ormandy interpretation of Brahms' Hungarian Dance no. 5. This piece has the musical form $A_1A_2B_1B_2CA_3B_3B_4D$ consisting of three repeating A-parts A_1, A_2, and A_3, four repeating B-parts B_1, B_2, B_3, and B_4, as well as a C- and a D-part. Generally, we will denote musical parts of a piece of music by capital letters such as X, where all repetitions of X are enumerated as X_1, X_2, and so on. In the following, we will distinguish between a *piece of music* (in an abstract sense) and a particular *audio recording* (a concrete interpretation) of the piece. Here, the term *part* will be used in the context of the abstract music domain, whereas the term *segment* will be used for the audio domain.

The self-similarity matrix of the Brahms recording (with respect to suitable audio features and a particular similarity measure) is shown in Figure 1. Here, the repetitions implied by the musical form are reflected by the path structure of the matrix. For example, the path starting at $(1, 22)$ and ending at $(22, 42)$ (measured in seconds) indicates that the audio segment represented by the time interval $[1 : 22]$ is similar to the segment $[22 : 42]$. Manual inspection reveals that the segment $[1 : 22]$ corresponds to part A_1, whereas $[22 : 42]$ corresponds to A_2. Furthermore, the curved path starting at $(42, 69)$ and ending at $(69, 89)$ indicates that the segment $[42 : 69]$ (corresponding to B_1) is similar to $[69 : 89]$ (corresponding to B_2). Note that in the Ormandy interpretation, the B_2-part is played much faster than the B_1-part. This fact is also revealed by the gradient of the path, which encodes the relative tempo difference between the two segments.

As a second example, for short referred to as *Shostakovich example*, we consider Shostakovich's Waltz 2 from his Jazz Suite no. 2 in a Chailly interpretation. This piece has the musical form $A_1A_2BC_1C_2A_3A_4D$, where the theme, represented by the A-part, appears four times. However, there are significant variations in the four A-parts concerning instrumentation, articulation, as well as dynamics. For example, in A_1 the theme is played by a clarinet, in A_2 by strings, in A_3 by a trombone, and in A_4 by the full orchestra. As is illustrated by Figure 2, these variations result in a fragmented path structure of low quality, making it hard to identify the musically similar segments $[4 : 40]$, $[43 : 78]$, $[145 : 179]$, and $[182 : 217]$ corresponding to A_1, A_2, A_3, and A_4, respectively.

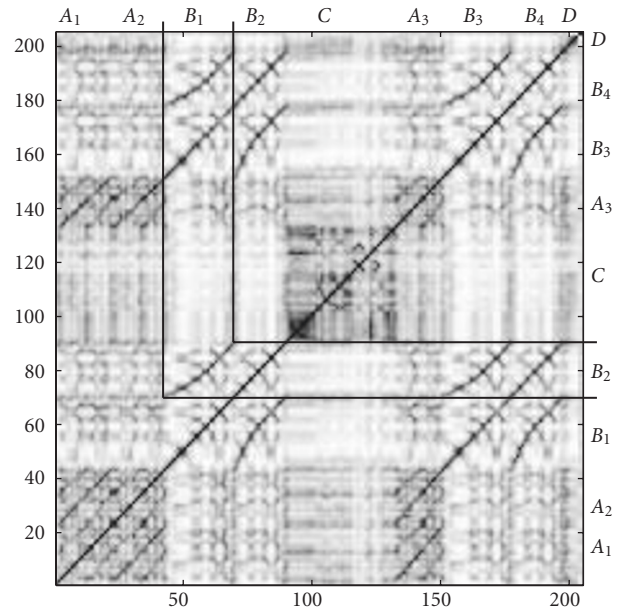

Figure 1: Self-similarity matrix $\mathcal{S}[41, 10]$ of an Ormandy interpretation of Brahms' Hungarian Dance no. 5. Here, dark colors correspond to low values (high similarity) and light colors correspond to high values (low similarity). The musical form $A_1A_2B_1B_2CA_3B_3B_4D$ is reflected by the path structure. For example, the curved path marked by the horizontal and vertical lines indicates the similarity between the segments corresponding to B_1 and B_2.

1.2. Related work

Most of the recent approaches to structural audio analysis focus on the detection of repeating patterns in popular music based on the strategy as described in Section 1.1. The concept of similarity matrices has been introduced to the music context by Foote in order to visualize the time structure of audio and music [8]. Based on these matrices, Foote and Cooper [2] report on first experiments on automatic audio summarization using mel frequency cepstral coefficients (MFCCs). To allow for small variations in performance, orchestration, and lyrics, Bartsch and Wakefield [1, 9] introduced chroma-based audio features to structural audio analysis. Chroma features, representing the spectral energy of each of the 12 traditional pitch classes of the equal-tempered scale, were also used in subsequent works such as [3, 4]. Goto [4] describes a method that detects the chorus sections in audio recordings of popular music. Important contributions of this work are, among others, the automatic identification of both ends of a chorus section (without prior knowledge of the chorus length) and the introduction of some shifting technique which allows to deal with modulations. Furthermore,

[1] In this paper, d is a distance measure rather than a similarity measure assuming small values for similar and large values for dissimilar feature vectors. Hence, the resulting matrix should strictly be called *distance matrix*. Nevertheless, we use the term *similarity matrix* according to the standard term used in previous work.

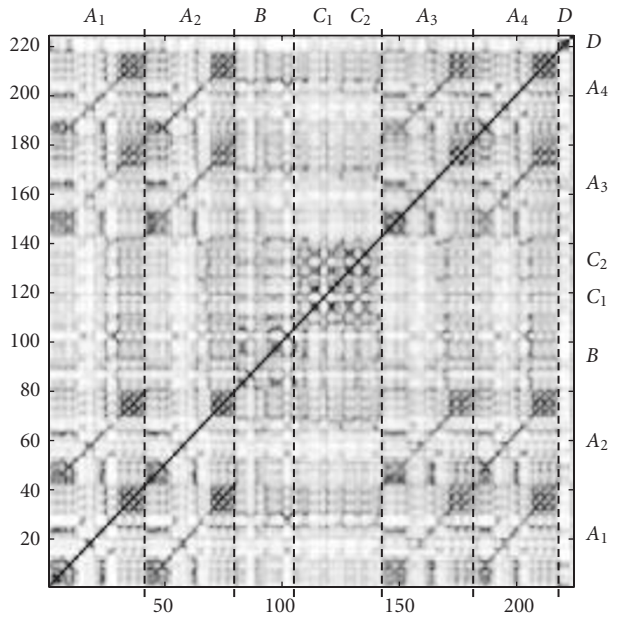

FIGURE 2: Self-similarity matrix $\mathcal{S}[41, 10]$ of a Chailly interpretation of Shostakovich's Waltz 2, Jazz Suite no. 2, having the musical form $A_1A_2BC_1C_2A_3A_4D$. Due to significant variations in the audio recording, the path structure is fragmented and of low quality. See also Figure 6.

Goto introduces a technique to cope with missing or inaccurately extracted candidates of repeating segments. In their work on repeating pattern discovery, Lu et al. [5] suggest a local distance measures that is invariant with respect to harmonic intervals, introducing some robustness to variations in instrumentation. Furthermore, they describe a postprocessing technique to optimize boundaries of the candidate segments. At this point we note that the above-mentioned approaches, while exploiting that repeating segments are of the same duration, are based on the constant tempo assumption. Dannenberg and Hu [3] describe several general strategies for path extraction, which indicate how to achieve robustness to small local tempo variations. There are also several approaches to structural analysis based on learning methods such as hidden Markov models (HMMs) used to cluster similar segments into groups; see, for example, [7, 10] and the references therein. In the context of *music summarization*, where the aim is to generate a list of the most representative musical segments without considering musical structure, Xu et al. [11] use support vector machines (SVMs) for classifying audio recordings into segments of pure and vocal music.

Maddage et al. [6] exploit some heuristics on the typical structure of popular music for both determining candidate segments and deriving the musical structure of a particular recording based on those segments. Their approach to structure analysis relies on the assumption that the analyzed recording follows a typical *verse-chorus pattern repetition*. As opposed to the general strategy introduced in Section 1.1, their approach only requires to implicitly calculate parts of a self-similarity matrix by considering only the candidate segments.

In summary, there have been several recent approaches to audio structure analysis that work well for music where the repetitions largely agree with respect to instrumentation, articulation, and tempo progression—as is often the case for popular music. In particular, most of the proposed strategies assume constant tempo throughout the piece (i.e., the path candidates have gradient $(1, 1)$ in the self-similarity matrix), which is then exploited in the path extraction and clustering procedure. For example, this assumption is used by Goto [4] in his strategy for segment recovery, by Lu et al. [5] in their boundary refinement, and by Chai et al. [12, 13] in their step of segment merging. The reported experimental results refer almost entirely to popular music. For this genre, the proposed structure analysis algorithms report on good results even in presence of variations with respect to instrumentation and lyrics.

For music, however, where musically similar segments exhibit significant variations in instrumentation, execution of note groups, and local tempo, there are yet no effective and efficient solutions to audio structure analysis. Here, the main difficulties arise from the fact that, due to spectral and temporal variations, the quality of the resulting path structure of the self-similarity matrix significantly suffers from missing and fragmented paths; see Figure 2. Furthermore, the presence of significant local tempo variations—as they frequently occur in Western classical music—cannot be dealt with by the suggested strategies. As another problem, the high time and space complexity of $O(N^2)$ to compute and store the similarity matrices makes the usage of self-similarity matrices infeasible for large N. It is the objective of this paper to introduce several fundamental techniques, which allow to efficiently perform structural audio analysis even in presence of significant musical variations; see Section 1.3.

Finally, we mention that first audio interfaces have been developed facilitating intuitive audio browsing based on the extracted audio structure. The SmartMusicKIOSK system [14] integrates functionalities for jumping to the chorus section and other key parts of a popular song as well as for visualizing song structure. The system constitutes the first interface that allows the user to easily skip sections of low interest even within a song. The SyncPlayer system [15] allows a multimodal presentation of audio and associated music-related data. Here, a recently developed audio structure plug-in not only allows for an efficient audio browsing but also for a direct comparison of musically related segments, which constitutes a valuable tool in music research.

Further suitable references related to work will be given in the respective sections.

1.3. Contributions

In this paper, we introduce several new techniques, to afford an automatic and efficient structure analysis even in the presence of large musical variations. For the first time, we report on our experiments on Western classical music including

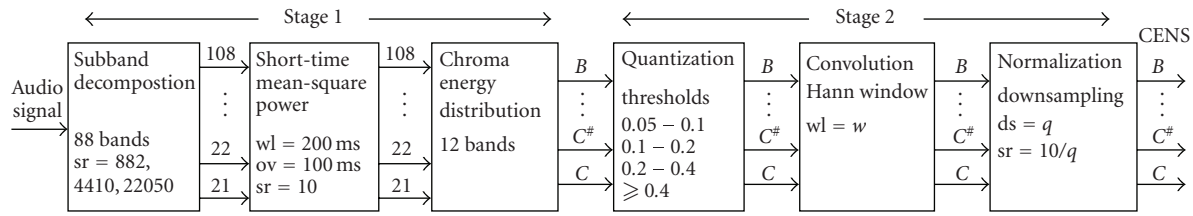

Figure 3: Two-stage CENS feature design (wl = window length, ov = overlap, sr = sampling rate, ds = downsampling factor).

complex orchestral pieces. Our proposed structure analysis algorithm follows the four-stage strategy as described in Section 1.1. Here, one essential idea is that we account for musical variations by incorporating invariance and robustness at all four stages simultaneously. The following overview summarizes the main contributions and describes the structure of this paper.

(1) Audio features

We introduce a new class of robust and scalable audio features considering short-time statistics over chroma-based energy distributions (Section 2). Such features not only allow to absorb variations in parameters such as dynamics, timbre, articulation, execution of note groups, and temporal microdeviations, but can also be efficiently processed in the subsequent steps due to their low resolution. The proposed features strongly correlate to the short-time harmonic content of the underlying audio signal.

(2) Similarity measure

As a second contribution, we significantly enhance the path structure of a self-similarity matrix by incorporating contextual information at various tempo levels into the local similarity measure (Section 3). This accounts for local temporal variations and significantly smooths the path structures.

(3) Path extraction

Based on the enhanced matrix, we suggest a robust and efficient path extraction procedure using a greedy strategy (Section 4). This step takes care of relative differences in the tempo progression between musically similar segments.

(4) Global structure

Each path encodes a pair of musically similar segments. To determine the global repetitive structure, we describe a one-step transitivity clustering procedure which balances out the inconsistencies introduced by inaccurate and incorrect path extractions (Section 5).

We evaluated our structure extraction algorithm on a wide range of Western classical music including complex orchestral and vocal works (Section 6). The experimental results show that our method successfully identifies the repetitive structure—often corresponding to the musical form of the underlying piece—even in the presence of significant variations as indicated by the Brahms and Shostakovich examples. Our MATLAB implementation performs the structure analysis task within a couple of minutes even for long and versatile audio recordings such as Ravel's Bolero, which has a duration of more than 15 minutes and possesses a rich path structure. Further results and an audio demonstration can be found at http://www-mmdb.iai.uni-bonn.de/projects/audiostructure.

2. ROBUST AUDIO FEATURES

In this section, we consider the design of audio features, where one has to deal with two mutually conflicting goals: robustness to admissible variations on the one hand and accuracy with respect to the relevant characteristics on the other hand. Furthermore, the features should support an efficient algorithmic solution of the problem they are designed for. In our structure analysis scenario, we consider audio segments as similar if they represent the same musical content regardless of the specific articulation and instrumentation. In other words, the structure extraction procedure has to be robust to variations in timbre, dynamics, articulation, local tempo changes, and global tempo up to the point of variations in note groups such as trills or grace notes.

In this section, we introduce a new class of audio features, which possess a high degree of robustness to variations of the above-mentioned parameters and strongly correlate to the harmonics information contained in the audio signals. In the feature extraction, we proceed in two stages as indicated by Figure 3. In the first stage, we use a small analysis window to investigate how the signal's energy locally distributes among the 12 chroma classes (Section 2.1). Using chroma distributions not only takes into account the close octave relationship in both melody and harmony as prominent in Western music, see [1], but also introduces a high degree of robustness to variations in dynamics, timbre, and articulation. In the second stage, we use a much larger statistics window to compute thresholded short-time statistics over these chroma energy distributions in order to introduce robustness to local time deviations and additional notes (Section 2.2). (As a general strategy, statistics such as pitch histograms for audio signals have been proven to be a useful tool in music genre classification, see, e.g., [16].) In the following, we identify the musical notes $A0$ to $C8$ (the range of a standard piano) with the MIDI pitches $p = 21$ to $p = 108$. For example, we speak of the note $A4$ (frequency 440 Hz) and simply write $p = 69$.

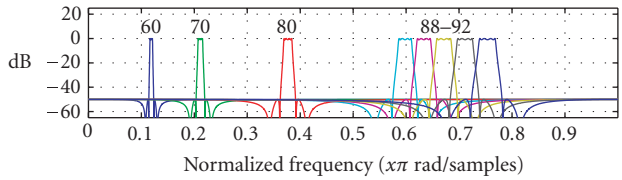

FIGURE 4: Magnitude responses in dB for the elliptic filters corresponding to the MIDI notes 60, 70, 80, and 88 to 92 with respect to the sampling rate of 4410 Hz.

2.1. First stage: local chroma energy distribution

First, we decompose the audio signal into 88 frequency bands with center frequencies corresponding to the MIDI pitches $p = 21$ to $p = 108$. To properly separate adjacent pitches, we need filters with narrow passbands, high rejection in the stopbands, and sharp cutoffs. In order to design a set of filters satisfying these stringent requirements for all MIDI notes in question, we work with three different sampling rates: 22050 Hz for high frequencies ($p = 96, \ldots, 108$), 4410 Hz for medium frequencies ($p = 60, \ldots, 95$), and 882 Hz for low frequencies ($p = 21, \ldots, 59$). To this end, the original audio signal is downsampled to the required sampling rates after applying suitable antialiasing filters. Working with different sampling rates also takes into account that the time resolution naturally decreases in the analysis of lower frequencies. Each of the 88 filters is realized as an eighth-order elliptic filter with 1 dB passband ripple and 50 dB rejection in the stopband. To separate the notes, we use a Q factor (ratio of center frequency to bandwidth) of $Q = 25$ and a transition band having half the width of the passband. Figure 4 shows the magnitude response of some of these filters.

Elliptic filters have excellent cutoff properties as well as low filter orders. However, these properties are at the expense of large-phase distortions and group delays. Since in our offline scenario the entire audio signals are known prior to the filtering step, one can apply the following trick: after filtering in the forward direction, the filtered signal is reversed and run back through the filter. The resulting output signal has precisely zero-phase distortion and a magnitude modified by the square of the filter's magnitude response. Further details may be found in standard text books on digital signal processing such as [17].

As a next step, we compute the short-time mean-square power (STMSP) for each of the 88 subbands by convolving the squared subband signals by a 200 ms rectangular window with an overlap of half the window size. Note that the actual window size depends on the respective sampling rate of 22050, 4410, and 882 Hz, which is compensated in the energy computation by introducing an additional factor of 1, 5, and 25, respectively. Then, we compute STMSPs of all chroma classes $C, C^{\#}, \ldots, B$ by adding up the corresponding STMSPs of all pitches belonging to the respective class. For example, to compute the STMSP of the chroma C, we add up the STMSPs of the pitches $C1, C2, \ldots, C8$ (MIDI pitches $24, 36, \ldots, 108$). This yields for every 100 ms a real 12-dimensional vector $\vec{v} = (v_1, v_2 \ldots, v_{12}) \in \mathbb{R}^{12}$, where v_1 corresponds to chroma C, v_2 to chroma $C^{\#}$, and so on. Finally, we compute the energy distribution relative to the 12 chroma classes by replacing $\vec{v}$ by $\vec{v}/(\sum_{i=1}^{12} v_i)$.

In summary, in the first stage the audio signal is converted into a sequence $(\vec{v}^1, \vec{v}^2, \ldots, \vec{v}^N)$ of 12-dimensional chroma distribution vectors $\vec{v}^n \in [0,1]^{12}$ for $1 \leq n \leq N$. For the Brahms example given in the introduction, the resulting sequence is shown in Figure 5 (light curve). Furthermore, 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 assign an equally distributed chroma energy to such passages. We also tested the short-time Fourier transform (STFT) to compute the chroma features by pooling the spectral coefficients as suggested in [1]. Even though obtaining similar features, our filter bank approach, while having a comparable computational cost, allows a better control over the frequency bands. This particularly holds for the low frequencies, which is due to the more adequate resolution in time and frequency.

2.2. Second stage: normalized short-time statistics

In view of possible variations in local tempo, articulation, and note execution, the local chroma energy distribution features are still too sensitive. Furthermore, as it will turn out in Section 3, a flexible and computationally inexpensive procedure is needed to adjust the feature resolution. Therefore, we further process the chroma features by introducing a second much larger statistics window and consider *short-time statistics* concerning the chroma energy distribution over this window. More specifically, let $Q : [0,1] \to \{0,1,2,3,4\}$ be a quantization function defined by

$$Q(a) := \begin{cases} 0 & \text{for } 0 \leq a < 0.05, \\ 1 & \text{for } 0.05 \leq a < 0.1, \\ 2 & \text{for } 0.1 \leq a < 0.2, \\ 3 & \text{for } 0.2 \leq a < 0.4, \\ 4 & \text{for } 0.4 \leq a \leq 1. \end{cases} \tag{1}$$

Then, we quantize each chroma energy distribution vector $\vec{v}^n = (v_1^n, \ldots, v_{12}^n) \in [0,1]^{12}$ by applying Q to each component of $\vec{v}^n$, yielding $Q(\vec{v}^n) := (Q(v_1^n), \ldots, Q(v_{12}^n))$. Intuitively, this quantization assigns a value of 4 to a chroma component v_i^n if the corresponding chroma class contains more than 40 percent of the signal's total energy and so on. The thresholds are chosen in a logarithmic fashion. Furthermore, chroma components below a 5-percent threshold are excluded from further considerations. For example, the vector $\vec{v}^n = (0.02, 0.5, 0.3, 0.07, 0.11, 0, \ldots, 0)$ is transformed into the vector $Q(\vec{v}^n) := (0, 4, 3, 1, 2, 0, \ldots, 0)$.

In a subsequent step, we convolve the sequence $(Q(\vec{v}^1), \ldots, Q(\vec{v}^N))$ componentwise with a Hann window of length $w \in \mathbb{N}$. This again results in a sequence of 12-dimensional vectors with nonnegative entries, representing a kind of

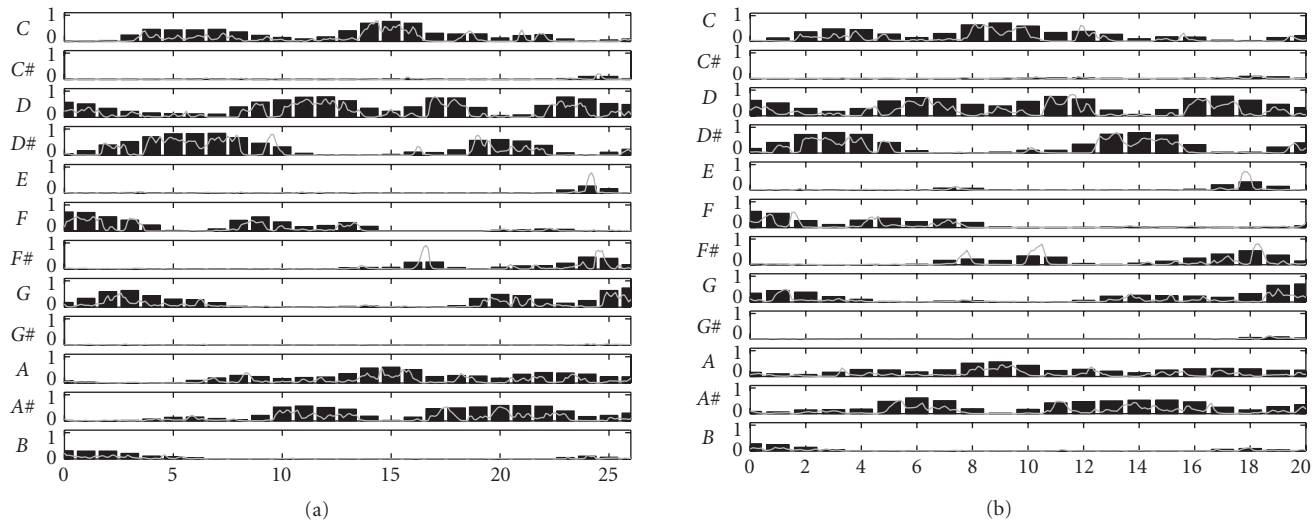

FIGURE 5: Local chroma energy distributions (light curves, 10 feature vectors per second) and CENS feature sequence (dark bars, 1 feature vector per second) of the segment [42 : 69] ((a) corresponding to B_1) and segment [69 : 89] ((b) corresponding to B_2) of the Brahms example shown in Figure 1. Note that even though the relative tempo progression in the parts B_1 and B_2 is different, the harmonic progression at the low resolution level of the CENS features is very similar.

weighted statistics of the energy distribution over a window of w consecutive vectors. In a last step, this sequence is downsampled by a factor of q. The resulting vectors are normalized with respect to the Euclidean norm. For example, if $w = 41$ and $q = 10$, one obtains one feature vector per second, each corresponding to roughly 4100 ms of audio. For short, the resulting features are referred to as CENS$[w, q]$ (chroma energy distribution normalized statistics). These features are elements of the following set of vectors:

$$\mathcal{F} := \left\{ \vec{v} = (v_1, \ldots, v_{12})^\top \in [0,1]^{12} \mid \sum_{i=1}^{12} v_i^2 = 1 \right\}. \quad (2)$$

Figure 5 shows the resulting sequence of CENS feature vectors for our Brahms example. Similar features have been applied in the audio matching scenario; see [18].

By modifying the parameters w and q, we may adjust the feature granularity and sampling rate without repeating the cost-intensive computations in Section 2.1. Furthermore, changing the thresholds and values of the quantization function Q allows to enhance or mask out certain aspects of the audio signal, for example, making the CENS features insensitive to noise components that may arise during note attacks. Finally, using statistics over relatively large windows not only smooths out microtemporal deviations, as may occur for articulatory reasons, but also compensates for different realizations of note groups such as trills or arpeggios.

In conclusion, we mention some potential problems concerning the proposed CENS features. The usage of a filter bank with fixed frequency bands is based on the assumption of well-tuned instruments. Slight deviations of up to 30–40 cents from the center frequencies can be compensated by the filters which have relatively wide passbands of constant amplitude response. Global deviations in tuning can be compensated by employing a suitably adjusted filter bank. However, phenomena such as strong string vibratos or pitch oscillation as is typical for, for example, kettledrums lead to significant and problematic pitch smearing effects. Here, the detection and smoothing of such fluctuations, which is certainly not an easy task, may be necessary prior to the filtering step. However, as we will see in Section 6, the CENS features generally still lead to good analysis results even in the presence of the artifacts mentioned above.

3. SIMILARITY MEASURE

In this section, we introduce a strategy for enhancing the path structure of a self-similarity matrix by designing a suitable local similarity measure. To this end, we proceed in three steps. As a starting point, let $d : \mathcal{F} \times \mathcal{F} \to [0,1]$ be the similarity measure on the space $\mathcal{F} \subset \mathbb{R}^{12}$ of CENS feature vectors (see (2)) defined by

$$d(\vec{v}, \vec{w}) := 1 - \langle \vec{v}, \vec{w} \rangle \quad (3)$$

for CENS$[w, q]$-vectors $\vec{v}, \vec{w} \in \mathcal{F}$. Since $\vec{v}$ and $\vec{w}$ are normalized, the inner product $\langle \vec{v}, \vec{w} \rangle$ coincides with the cosine of the angle between $\vec{v}$ and $\vec{w}$. For short, the resulting self-similarity matrix will also be denoted by $\mathcal{S}[w, q]$ or simply by $\mathcal{S}$ if w and q are clear from the context.

To further enhance the path structure of $\mathcal{S}[w, q]$, we incorporate contextual information into the local similarity measure. A similar approach has been suggested in [1] or [5], where the self-similarity matrix is filtered along diagonals assuming constant tempo. We will show later in this section how to remove this assumption by, intuitively speaking, filtering along various directions simultaneously, where each of the directions corresponds to a different local tempo. In [7],

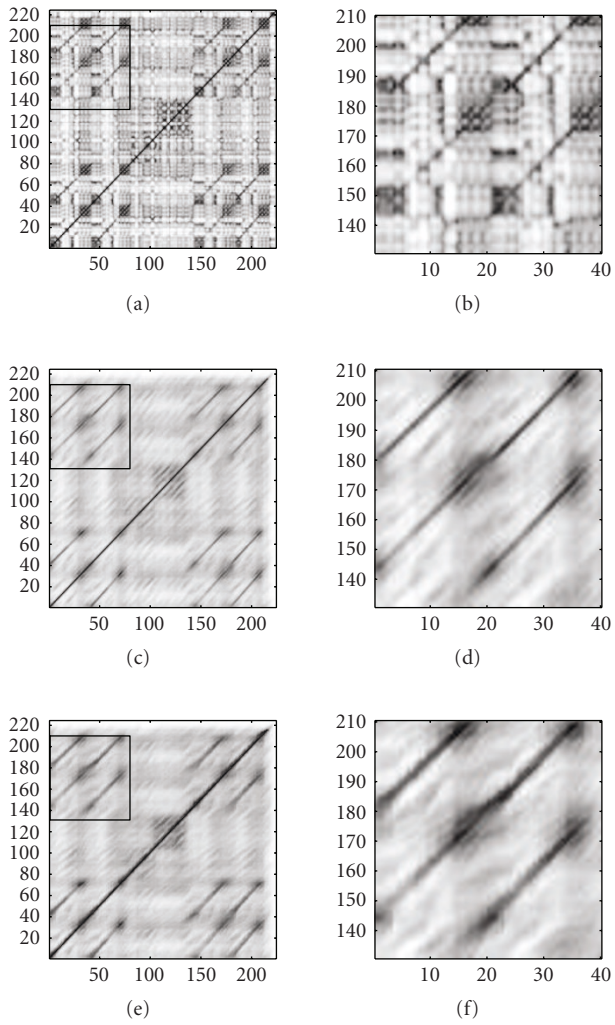

FIGURE 6: Enhancement of the similarity matrix of the Shostakovich example; see Figure 2. (a) and (b): $\mathcal{S}[41, 10]$ and enlargement. (c) and (d): $\mathcal{S}_{10}[41, 10]$ and enlargement. (e) and (f): $\mathcal{S}_{10}^{\min}[41, 10]$ and enlargement.

matrix enhancement is achieved by using HMM-based "dynamic" features, which model the temporal evolution of the spectral shape over a fixed time duration. For the moment, we also assume constant tempo and then, in a second step, describe how to get rid of this assumption. Let $L \in \mathbb{N}$ be a length parameter. We define the *contextual similarity measure* d_L by

$$d_L(n, m) := \frac{1}{L} \sum_{\ell=0}^{L-1} d(\vec{v}^{n+\ell}, \vec{v}^{m+\ell}), \tag{4}$$

where $1 \leq n, m \leq N - L + 1$. By suitably extending the CENS sequence $(\vec{v}^1, \ldots, \vec{v}^N)$, for example, via zero-padding, one may extend the definition to $1 \leq n, m \leq N$. Then, the *contextual similarity matrix* $\mathcal{S}_L$ is defined by $\mathcal{S}_L(n, m) := d_L(n, m)$. In this matrix, a value $d_L(n, m) \in [0, 1]$ close to zero implies that the entire L-sequence $(\vec{v}^n, \ldots, \vec{v}^{n+L-1})$ is similar to the L-sequence $(\vec{v}^m, \ldots, \vec{v}^{m+L-1})$, resulting in an enhancement of the diagonal path structure in the similarity matrix.

TABLE 1: Tempo changes (tc) simulated by changing the statistics window size w and the downsampling factor q.

w	29	33	37	**41**	45	49	53	57
q	7	8	9	**10**	11	12	13	14
tc	1.43	1.25	1.1	**1.0**	0.9	0.83	0.77	0.7

This is also illustrated by our Shostakovich example, showing $\mathcal{S}[41, 10]$ in Figure 6(a) and $\mathcal{S}_{10}[41, 10]$ in Figure 6(c). Here, the diagonal path structure of $\mathcal{S}_{10}[41, 10]$—as opposed to the one of $\mathcal{S}[41, 10]$—is much clearer, which not only facilitates the extraction of structural information but also allows to further decrease the feature sampling rate. Note that the contextual similarity matrix $\mathcal{S}_L$ can be efficiently computed from $\mathcal{S}$ by applying an averaging filter along the diagonals. More precisely, $\mathcal{S}_L(n, m) = (1/L) \sum_{\ell=0}^{L-1} \mathcal{S}(n + \ell, m + \ell)$ (with a suitable zero-padding of $\mathcal{S}$).

So far, we have enhanced similarity matrices by regarding the context of L consecutive features vectors. This procedure is problematic when similar segments do not have the same tempo. Such a situation frequently occurs in classical music—even within the same interpretation—as is shown by our Brahms example; see Figure 1. To account for such variations we, intuitively speaking, create several versions of one of the audio data streams, each corresponding to a different global tempo, which are then incorporated into one single similarity measure. More precisely, let $\vec{V}[w, q]$ denote the CENS$[w, q]$ sequence of length $N[w, q]$ obtained from the audio data stream in question. For the sake of concreteness, we choose $w = 41$ and $q = 10$ as reference parameters, resulting in a feature sampling rate of 1 Hz. We now simulate a tempo change of the data stream by modifying the values of w and q. For example, using a window size of $w = 53$ (instead of 41) and a downsampling factor of $q = 13$ (instead of 10) simulates a tempo change of the original data stream by a factor of $10/13 \approx 0.77$. In our experiments, we used 8 different tempi as indicated by Table 1, covering tempo variations of roughly -30 to $+40$ percent. We then define a new similarity measure $d_L^{\min}$ by

$$d_L^{\min}(n, m) := \min_{[w,q]} \frac{1}{L} \sum_{\ell=0}^{L-1} d\Big(\vec{v}[41, 10]^{n+\ell}, \vec{v}[w, q]^{\widetilde{m}+\ell}\Big), \tag{5}$$

where the minimum is taken over the pairs $[w, q]$ listed in Table 1 and $\widetilde{m} = \lceil m \cdot 10/q \rceil$. In other words, at position (n, m), the L-subsequence of $\vec{V}[41, 10]$ starting at absolute time n (note that the feature sampling rate is 1 Hz) is compared with the L-subsequence of $\vec{V}[w, q]$ (simulating a tempo change of $10/q$) starting at absolute time m (corresponding to feature position $\widetilde{m} = \lceil m \cdot 10/q \rceil$). From this we obtain the modified contextual similarity matrix $\mathcal{S}_L^{\min}$ defined by $\mathcal{S}_L^{\min}(n, m) := d_L^{\min}(n, m)$. Figure 7 shows that incorporating local tempo variations into contextual similarity matrices significantly improves the quality of the path structure, in particular for the case that similar audio segments exhibit different local relative tempi.

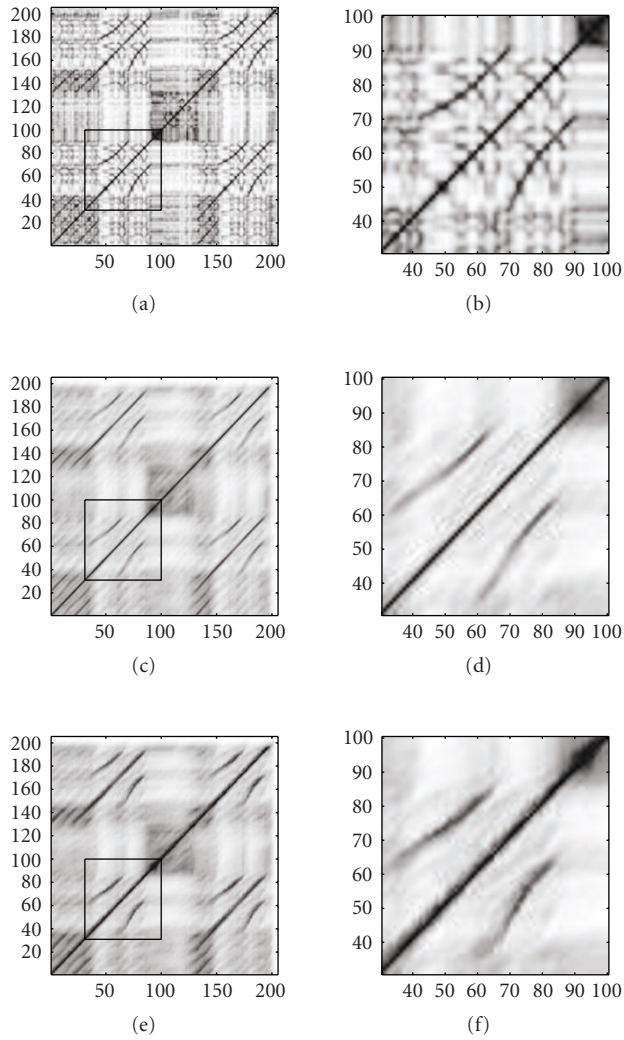

FIGURE 7: Enhancement of the similarity matrix of the Brahms example; see Figure 1. (a) and (b): $\mathcal{S}[41, 10]$ and enlargement. (c) and (d): $\mathcal{S}_{10}[41, 10]$ and enlargement. (e) and (f): $\mathcal{S}_{10}^{\min}[41, 10]$ and enlargement.

4. PATH EXTRACTION

In the last two sections, we have introduced a combination of techniques—robust CENS features and usage of contextual information—resulting in smooth and structurally enhanced self-similarity matrices. We now describe a flexible and efficient strategy to extract the paths of a given self-similarity matrix $\mathcal{S} = \mathcal{S}_L^{\min}[w, q]$.

Mathematically, we define a path to be a sequence $P = (p_1, p_2, \ldots, p_K)$ of pairs of indices $p_k = (n_k, m_k) \in [1 : N]^2$, $1 \leq k \leq K$, satisfying the path constraints

$$p_{k+1} = p_k + \delta \quad \text{for some } \delta \in \Delta, \tag{6}$$

where $\Delta := \{(1, 1), (1, 2), (2, 1)\}$ and $1 \leq k \leq K - 1$. The pairs p_k will also be called the *links* of P. Then the *cost* of link $p_k = (n_k, m_k)$ is defined as $\mathcal{S}(n_k, m_k)$. Now, it is the objective to extract long paths consisting of links having low costs. Our path extraction algorithm consists of three steps. In step (1), we start with a link of minimal cost, referred to as *initial link*, and construct a path in a greedy fashion by iteratively adding links of low cost, referred to as *admissible links*. In step (2), all links in a neighborhood of the constructed path are excluded from further considerations by suitably modifying $\mathcal{S}$. Then, steps (1) and (2) are repeated until there are no links of low cost left. Finally, the extracted paths are postprocessed in step (3). The details are as follows.

(0) Initialization

Set $\mathcal{S} = \mathcal{S}_L^{\min}[w, q]$ and let $C_{\text{in}}, C_{\text{ad}} \in \mathbb{R}_{>0}$ be two suitable thresholds for the maximal cost of the initial links and the admissible links, respectively. (In our experiments, we typically chose $0.08 \leq C_{\text{in}} \leq 0.15$ and $0.12 \leq C_{\text{ad}} \leq 0.2$.) We modify $\mathcal{S}$ by setting $\mathcal{S}(n, m) = C_{\text{ad}}$ for $n \leq m$, that is, the links below the diagonal will be excluded in the following steps. Similarly, we exclude the neighborhood of the diagonal path $P = ((1, 1), (2, 2), \ldots, (N, N))$ by modifying $\mathcal{S}$ using the path removal strategy as described in step (2).

(1) Path construction

Let $p_0 = (n_0, m_0) \in [1 : N]^2$ be the indices minimizing $\mathcal{S}(n, m)$. If $\mathcal{S}(n_0, m_0) \geq C_{\text{in}}$, the algorithm terminates. Otherwise, we construct a new path P by extending p_0 iteratively, where all possible extensions are described by Figure 8(a). Suppose we have already constructed $P = (p_a, \ldots, p_0, \ldots, p_b)$ for $a \leq 0$ and $b \geq 0$. Then, if $\min_{\delta \in \Delta}(\mathcal{S}(p_b + \delta)) < C_{\text{ad}}$, we extend P by setting

$$p_{b+1} := p_b + \arg\min_{\delta \in \Delta}(\mathcal{S}(p_b + \delta)), \tag{7}$$

and if $\min_{\delta \in \Delta}(\mathcal{S}(p_a - \delta)) < C_{\text{ad}}$, extend P by setting

$$p_{a-1} := p_a - \arg\min_{\delta \in \Delta}(\mathcal{S}(p_a - \delta)). \tag{8}$$

Figure 8(b) illustrates such a path. If there are no further extensions with admissible links, we proceed with step (2). Shifting the indices by $a + 1$, we may assume that the resulting path is of the form $P = (p_1, \ldots, p_K)$ with $K = a + b + 1$.

(2) Path removal

For a fixed link $p_k = (n_k, m_k)$ of P, we consider the maximal number $m_k \leq m^* \leq N$ with the property that $\mathcal{S}(n_k, m_k) \leq \mathcal{S}(n_k, m_k + 1) \leq \cdots \leq \mathcal{S}(n_k, m^*)$. In other words, the sequence $(n_k, m_k), (n_k, m_k + 1), \ldots, (n_k, m^*)$ defines a *ray* starting at position (n_k, m_k) and running horizontally to the right such that $\mathcal{S}$ is monotonically increasing. Analogously, we consider three other types of rays starting at position (n_k, m_k) running horizontally to the left, vertically upwards, and vertically downwards; see Figure 8(c) for an illustration. We then consider all such rays for all links p_k of P. Let $\mathcal{N}(P) \subset [1 : N]^2$ be the set of all pairs (n, m) lying on one of these rays. Note that $\mathcal{N}(P)$ defines a neighborhood of the path P. To exclude the links of $\mathcal{N}(P)$ from further consideration, we set $\mathcal{S}(n, m) = C_{\text{ad}}$ for all $(n, m) \in \mathcal{N}(P)$ and continue by repeating step (1).

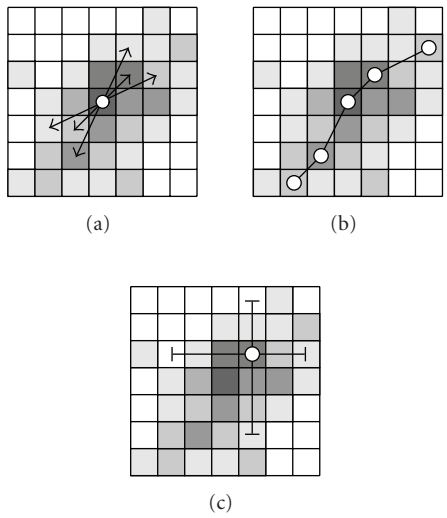

FIGURE 8: (a) Initial link and possible path extensions. (b) Path resulting from step (1). (c) Rays used for path removal in step (2).

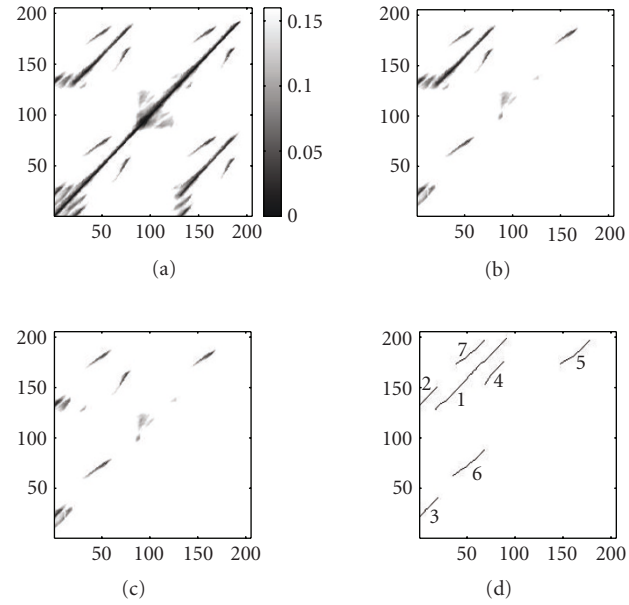

FIGURE 9: Illustration of the path extraction algorithm for the Brahms example of Figure 1. (a) Self-similarity matrix $\mathcal{S} = \mathcal{S}_{16}^{\min}[41, 10]$. Here, all values exceeding the threshold $C_{\text{ad}} = 0.16$ are plotted in white. (b) Matrix $\mathcal{S}$ after step (0) (initialization). (c) Matrix $\mathcal{S}$ after performing steps (1) and (2) once using the thresholds $C_{\text{in}} = 0.08$ and $C_{\text{ad}} = 0.16$. Note that a long path in the left upper corner was constructed, the neighborhood of which has then been removed. (d) Resulting path set $\mathcal{P} = \{P_1, \ldots, P_7\}$ after the postprocessing of step (3) using $K_0 = 5$ and $C_{\text{pr}} = 0.10$. The index m of P_m is indicated along each respective path.

In our actual implementation, we made step (2) more robust by softening the monotonicity condition on the rays. After the above algorithm terminates, we obtain a set of paths denoted by $\mathcal{P}$, which is postprocessed in a third step by means of some heuristics. For the following, let $P = (p_1, p_2, \ldots, p_K)$ denote a path in $\mathcal{P}$.

(3a) Removing short paths

All paths that have a length K shorter than a threshold $K_0 \in \mathbb{N}$ are removed. (In our experiments, we chose $5 \leq K_0 \leq 10$.) Such paths frequently occur as a result of residual links that have not been correctly removed by step (2).

(3b) Pruning paths

We prune each path $P \in \mathcal{P}$ at the beginning by removing the links $p_1, p_2, \ldots, p_{k_0}$ up to the index $0 \leq k_0 \leq K$, where k_0 denotes the maximal index such that the cost of each link $p_1, p_2, \ldots, p_{k_0}$ exceeds some suitably chosen threshold C_{pr} lying in between C_{in} and C_{ad}. Analogously, we prune the end of each path. This step is performed due to the following observation: introducing contextual information into the local similarity measure results in a smoothing effect of the paths along the diagonal direction. This, in turn, results in a blurring effect at the beginning and end of such paths—as illustrated by Figure 6(f)—unnaturally extending such paths at both ends in the construction of step (1).

(3c) Extending paths

We then extend each path $P \in \mathcal{P}$ at its end by adding suitable links $p_{K+1}, \ldots, p_{K+L_0}$. This step is performed due to the following reason: since we have incorporated contextual information into the local similarity measure, a low cost $\mathcal{S}(p_K) = d_L^{\min}(n_K, m_K)$ of the link $p_K = (n_K, m_K)$ implies that the whole sequence $(\vec{v}^{n_K}[41, 10], \ldots, \vec{v}^{n_K+L-1}[41, 10])$ is similar to $(\vec{v}^{m_K}[w, q], \ldots, \vec{v}^{m_K+L-1}[w, q])$ for the minimizing $[w, q]$ of Table 1; see Section 3. Here the length and direction of the extension $p_{K+1}, \ldots, p_{K+L_0}$ depends on the values $[w, q]$. (In the case $[w, q] = [41, 10]$, we set $L_0 = L$ and $p_k = p_K + (k, k)$ for $k = 1, \ldots, L_0$.)

Figure 9 illustrates the steps of our path extraction algorithm for the Brahms example. Part (d) shows the resulting path set $\mathcal{P}$. Note that each path corresponds to a pair of similar segments and encodes the relative tempo progression between these two segments. Figure 10(b) shows the set $\mathcal{P}$ for the Shostakovich example. In spite of the matrix enhancement, the similarity between the segments corresponding to A_1 and A_3 has not been correctly identified, resulting in the aborted path P_1 (which should correctly start at link (4, 145)). Even though, as we will show in the next section, the extracted information is sufficient to correctly derive the global structure.

5. GLOBAL STRUCTURE ANALYSIS

In this section, we propose an algorithm to determine the global repetitive structure of the underlying piece of music from the relations defined by the extracted paths. We first introduce some notation. A *segment* $\alpha = [s : t]$ is given by its starting point s and end point t, where s and t are given

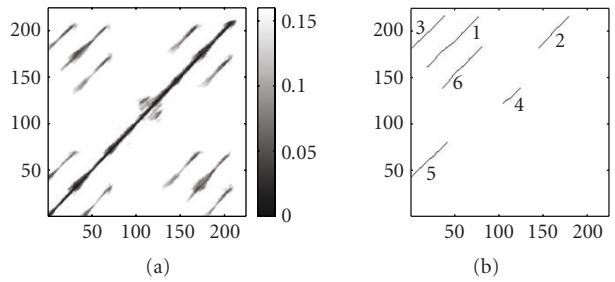

FIGURE 10: Shostakovich example of Figure 2. (a) $\mathcal{S}_{16}^{\min}[41, 10]$. (b) $\mathcal{P} = \{P_1, \ldots, P_6\}$ based on the same parameters as in the Brahms example of Figure 9. The index m of P_m is indicated along each respective path.

in terms of the corresponding indices in the feature sequence $\vec{V} = (\vec{v}^1, \vec{v}^2, \ldots, \vec{v}^N)$; see Section 1. A *similarity cluster* $\mathcal{A} := \{\alpha_1, \ldots, \alpha_M\}$ of size $M \in \mathbb{N}$ is defined to be a set of segments α_m, $1 \leq m \leq M$, which are considered to be mutually similar. Then, the *global structure* is described by a complete list of relevant similarity clusters of maximal size.

In other words, the list should represent all repetitions of musically relevant segments. Furthermore, if a cluster contains a segment α, then the cluster should also contain all other segments similar to α. For example, in our Shostakovich example of Figure 2 the global structure is described by the clusters $\mathcal{A}_1 = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ and $\mathcal{A}_2 = \{\gamma_1, \gamma_2\}$, where the segments α_k correspond to the parts A_k for $1 \leq k \leq 4$ and the segments γ_k to the parts C_k for $1 \leq k \leq 2$. Given a cluster $\mathcal{A} = \{\alpha_1, \ldots, \alpha_M\}$ with $\alpha_m = [s_m : t_m]$, $1 \leq m \leq M$, the *support* of $\mathcal{A}$ is defined to be the subset

$$\operatorname{supp}(\mathcal{A}) := \bigcup_{m=1}^{M} [s_m : t_m] \subset [1 : N]. \tag{9}$$

Recall that each path P indicates a pair of similar segments. More precisely, the path $P = (p_1, \ldots, p_K)$ with $p_k = (n_k, m_k)$ indicates that the segment $\pi_1(P) := [n_1 : n_K]$ is similar to the segment $\pi_2(P) := [m_1 : m_K]$. Such a pair of segments will also be referred to as a *path relation*. As an example, Figure 11(a) shows the path relations of our Shostakovich example. In this section, we describe an algorithm that derives large and consistent similarity clusters from the path relations induced by the set $\mathcal{P}$ of extracted paths. From a theoretical point of view, one has to construct some kind of transitive closure of the path relations; see also [3]. For example, if segment α is similar to segment β, and segment β is similar to segment γ, then α should also be regarded as similar to γ resulting in the cluster $\{\alpha, \beta, \gamma\}$. The situation becomes more complicated when α overlaps with some segment β which, in turn, is similar to segment γ. This would imply that a subsegment of α is similar to some subsegment of γ. In practice, the construction of similarity clusters by iteratively continuing in the above fashion is problematic. Here, inconsistencies in the path relations due to semantic (vague concept of musical similarity) or algorithmic

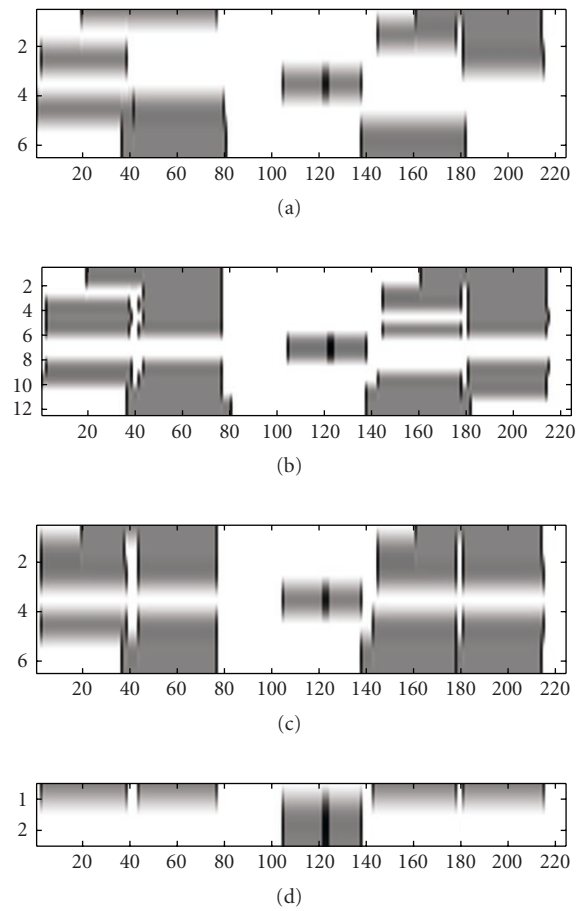

FIGURE 11: Illustration of the clustering algorithm for the Shostakovich example. The path set $\mathcal{P} = \{P_1, \ldots, P_6\}$ is shown in Figure 10(b). Segments are indicated by gray bars and overlaps are indicated by black regions. (a) Illustration of the two segments $\pi_1(P_m)$ and $\pi_2(P_m)$ for each path $P_m \in \mathcal{P}$, $1 \leq m \leq 6$. Row m corresponds to P_m. (b) Clusters $\mathcal{A}_m^1$ and $\mathcal{A}_m^2$ (rows $2m - 1$ and $2m$) computed in step (1) with $T_{\text{ts}} = 90$. (c) Clusters $\mathcal{A}_m$ (row m) computed in step (2). (d) Final result of the clustering algorithm after performing step (3) with $T_{\text{dc}} = 90$. The derived global structure is given by two similarity clusters. The first cluster corresponds to the musical parts $\{A_1, A_2, A_3, A_4\}$ (first row) and the second cluster to $\{C_1, C_2\}$ (second row) (cf. Figure 2).

(inaccurately extracted or missing paths) reasons may lead to meaningless clusters, for example, containing a series of segments where each segment is a slightly shifted version of its predecessor. For example, let $\alpha = [1 : 10]$, $\beta = [11 : 20]$, $\gamma = [22 : 31]$, and $\delta = [3 : 11]$. Then similarity relations between α and β, β and γ, and γ and δ would imply that $\alpha = [1 : 10]$ has to be regarded as similar to $\delta = [3 : 11]$, and so on. To balance out such inconsistencies, previous strategies such as [4] rely upon the constant tempo assumption. To achieve a robust and meaningful clustering even in the presence of significant local tempo variations, we suggest a new clustering algorithm, which proceeds in three steps. To this end, let $\mathcal{P} = \{P_1, P_2, \ldots, P_M\}$ be the set of extracted paths

P_m, $1 \leq m \leq M$. In step (1) (transitivity step) and step (2) (merging step), we compute for each P_m a similarity cluster $\mathcal{A}_m$ consisting of all segments that are either similar to $\pi_1(P_m)$ or to $\pi_2(P_m)$. In step (3), we then discard the redundant clusters. We exemplarily explain the procedure of steps (1) and (2) by considering the path P_1.

(1) Transitivity step

Let T_{ts} be a suitable tolerance parameter measured in percent (in our experiments we used $T_{\text{ts}} = 90$). First, we construct a cluster $\mathcal{A}_1^1$ for the path P_1 and the segment $\alpha := \pi_1(P_1)$. To this end, we check for all paths P_m whether the intersection $\alpha_0 := \alpha \cap \pi_1(P_m)$ contains more than T_{ts} percent of α, that is, whether $|\alpha_0|/|\alpha| \geq T_{\text{ts}}/100$. In the affirmative case, let β_0 be the subsegment of $\pi_2(P_m)$ that corresponds under P_m to the subsegment α_0 of $\pi_1(P_m)$. We add α_0 and β_0 to $\mathcal{A}_1^1$. Similarly, we check for all paths P_m whether the intersection $\alpha_0 := \alpha \cap \pi_2(P_m)$ contains more than T_{ts} percent of α and add in the affirmative case α_0 and β_0 to $\mathcal{A}_1^1$, where this time β_0 is the subsegment of $\pi_1(P_m)$ that corresponds under P_m to α_0. Note that β_0 generally does not have the same length as α_0. (Recall that the relative tempo variation is encoded by the gradient of P_m.) Analogously, we construct a cluster $\mathcal{A}_1^2$ for the path P_1 and the segment $\alpha := \pi_2(P_1)$. The clusters $\mathcal{A}_1^1$ and $\mathcal{A}_1^2$ can be regarded as the result of the first iterative step towards forming the transitive closure.

(2) Merging step

The cluster $\mathcal{A}_1$ is constructed by basically merging the clusters $\mathcal{A}_1^1$ and $\mathcal{A}_1^2$. To this end, we compare each segment $\alpha \in \mathcal{A}_1^1$ with each segment $\beta \in \mathcal{A}_1^2$. In the case that the intersection $\gamma := \alpha \cap \beta$ contains more than T_{ts} percent of α and of β (i.e., α essentially coincides with β), we add the segment γ to $\mathcal{A}_1$. In the case that for a fixed $\alpha \in \mathcal{A}_1^1$ the intersection $\alpha \cap \text{supp}(\mathcal{A}_1^2)$ contains less than $(100 - T_{\text{ts}})$ percent of α (i.e., α is essentially disjoint with all $\beta \in \mathcal{A}_1^2$), we add α to $\mathcal{A}_1$. Symmetrically, if for a fixed $\beta \in \mathcal{A}_1^2$ the intersection $\beta \cap \text{supp}(\mathcal{A}_1^1)$ contains less than $(100 - T_{\text{ts}})$ percent of β, we add β to $\mathcal{A}_1$. Note that by this procedure, the first case balances out small inconsistencies, whereas the second case and the third case compensate for missing path relations. Furthermore, segments $\alpha \in \mathcal{A}_1^1$ and $\beta \in \mathcal{A}_1^2$ that do not fall into one of the above categories indicate significant inconsistencies and are left unconsidered in the construction of $\mathcal{A}_1$.

After steps (1) and (2), we obtain a cluster $\mathcal{A}_1$ for the path P_1. In an analogous fashion, we compute clusters $\mathcal{A}_m$ for all paths P_m, $1 \leq m \leq M$.

(3) Discarding clusters

Let T_{dc} be a suitable tolerance parameter measured in percent (in our experiments we chose T_{dc} between 80 and 90 percent). We say that cluster $\mathcal{A}$ is a T_{dc}-*cover* of cluster $\mathcal{B}$ if the intersection $\text{supp}(\mathcal{A}) \cap \text{supp}(\mathcal{B})$ contains more than T_{dc} percent of $\text{supp}(\mathcal{B})$. By pairwise comparison of all clusters $\mathcal{A}_m$, we successively discard all clusters that are T_{dc}-covered by some other cluster consisting of a larger number of segments. (Here the idea is that a cluster with a larger number of smaller segments contains more information than a cluster having the same support while consisting of a smaller number of larger segments.) In the case that two clusters are mutual T_{dc}-covers and consist of the same number of segments, we discard the cluster with the smaller support.

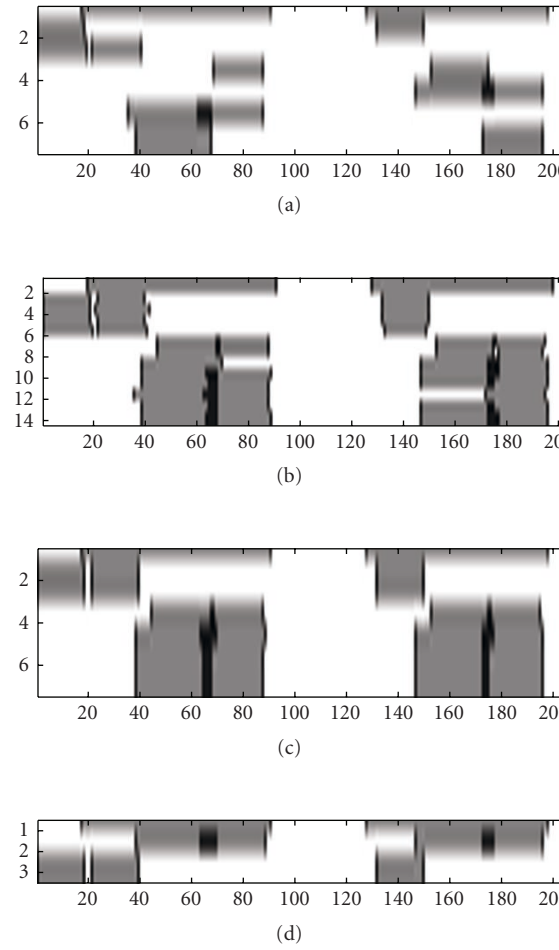

FIGURE 12: Steps of the clustering algorithm for the Brahms example, see Figure 9. For details we refer to Figure 11. The final result correctly represents the global structure: the cluster of the second row corresponds to $\{B_1, B_2, B_3, B_4\}$, and the one of the third row to $\{A_1, A_2, A_3\}$. Finally, the cluster of the first row expresses the similarity between $A_2B_1B_2$ and $A_3B_3B_4$ (cf. Figure 1).

The steps of the clustering algorithm are also illustrated by Figures 11 and 12. Recall from Section 4 that in the Shostakovich example, the significant variations in the instrumentation led to a defective path extraction. In particular, the similarity of the segments corresponding to parts A_1 and A_3 could not be correctly identified as reflected by the truncated path P_1; see Figures 10(b) and 11(a). Nevertheless, the correct global structure was derived by the clustering algorithm (cf. Figure 11(d)). Here, the missing relation was recovered by step (1) (transitivity step) from the

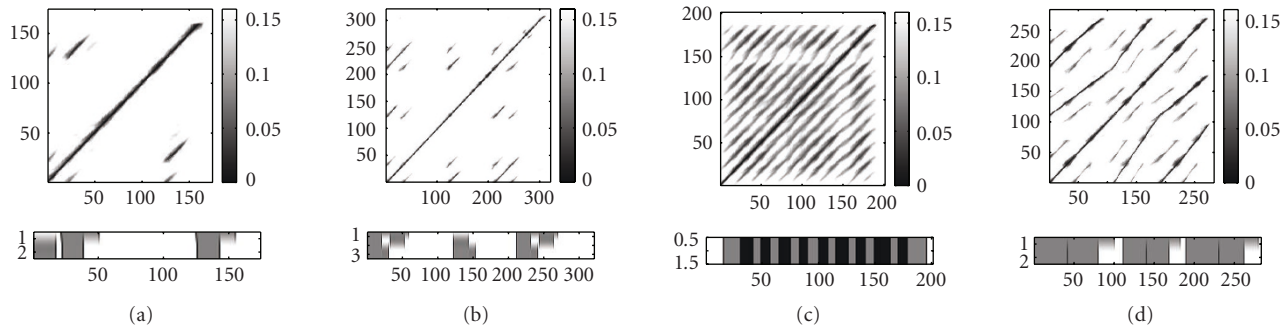

Figure 13: (a) Chopin, "Tristesse," Etude op. 10/3, played by Varsi. (b) Beethoven, "Pathetique," second movement, op. 13, played by Barenboim. (c) Gloria Gaynor, "I will survive." (d) Part A_1A_2B of the Shostakovich example of Figure 2 repeated three times in modified tempi (normal tempo, 140 percent of normal tempo, accelerating tempo from 100 to 140 percent).

correctly identified similarity relation between segments corresponding to A_3 and A_4 (path P_2) and between segments corresponding to A_1 and A_4 (path P_3). The effect of step (3) is illustrated by comparing (c) and (d) of Figure 11. Since the cluster $\mathcal{A}_5$ is a 90-percent cover of the clusters $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$, and $\mathcal{A}_6$, and has the largest support, the latter clusters are discarded.

6. EXPERIMENTS

We implemented our algorithm for audio structure analysis in MATLAB and tested it on about 100 audio recordings reflecting a wide range of mainly Western classical music, including pieces by Bach, Beethoven, Brahms, Chopin, Mozart, Ravel, Schubert, Schumann, Shostakovich, and Vivaldi. In particular, we used musically complex orchestral pieces exhibiting a large degree of variations in their repetitions with respect to instrumentation, articulation, and local tempo variations. From a musical point of view, the global repetitive structure is often ambiguous since it depends on the particular notion of similarity, on the degree of admissible variations, as well as on the musical significance and duration of the respective repetitions. Furthermore, the structural analysis can be performed at various levels: at a global level (e.g., segmenting a sonata into exposition, repetition of the exposition, development, and recapitulation), an intermediary level (e.g., further splitting up the exposition into first and second theme), or on a fine level (e.g., segmenting into repeating motifs). This makes the automatic structure extraction as well as an objective evaluation of the results a difficult and problematic task.

In our experiments, we looked for repetitions at a global to intermediary level corresponding to segments of at least 15–20 seconds of duration, which is reflected in our choice of parameters; see Section 6.1. In that section, we will also present some general results and discuss in detail two complex examples: Mendelssohn's Wedding March and Ravel's Bolero. In Section 6.2, we discuss the running time behavior of our implementation. It turns out that the algorithm is applicable to pieces even longer than 45 minutes, which covers essentially any piece of Western classical music. To account for transposed (pitch-shifted) repeating segments, we adopted the shifting technique suggested by Goto [4]. Some results will be discussed in Section 6.3. Further results and an audio demonstration can be found at http://www-mmdb.iai.uni-bonn.de/projects/audiostructure.

6.1. General Results

In order to demonstrate the capability of our structure analysis algorithm, we discuss some representative results in detail. This will also illustrate the kind of difficulties generally found in music structure analysis. Our algorithm is fully automatic, in other words, no prior knowledge about the respective piece is exploited in the analysis. In all examples, we use the following fixed set of parameters. For the self-similarity matrix, we use $\mathcal{S}_{16}^{\min}[41, 10]$ with a corresponding feature resolution of 1 Hz; see Section 3. In the path extraction algorithm of Section 4, we set $C_{\text{in}} = 0.08$, $C_{\text{ad}} = 0.16$, $C_{\text{pr}} = 0.10$, and $K_0 = 5$. Finally, in the clustering algorithm of Section 5, we set $T_{\text{ts}} = 90$ and $T_{\text{dc}} = 90$. The choice of the above parameters and thresholds constitutes a trade-off between being tolerant enough to allow relevant variations and being robust enough to deal with artifacts and inconsistencies.

As a first example, we consider a Varsi recording of Chopin's Etude op. 10/3 "Tristesse." The underlying piece has the musical form $A_1A_2B_1CA_3B_2D$. This structure has successfully been extracted by our algorithm; see Figure 13(a). Here, the first cluster $\mathcal{A}_1$ corresponds to the parts A_2B_1 and A_3B_2, whereas the second cluster $\mathcal{A}_2$ corresponds to the parts A_1, A_2, and A_3. For simplicity, we use the notation $\mathcal{A}_1 \sim \{A_2B_1, A_3B_2\}$ and $\mathcal{A}_2 \sim \{A_1, A_2, A_3\}$. The similarity relation between B_1 and B_2 is induced from cluster $\mathcal{A}_1$ by "subtracting" the respective A-part which is known from cluster $\mathcal{A}_2$. The small gaps between the segments in cluster $\mathcal{A}_2$ are due to the fact that the tail of A_1 (passage to A_2) is different from the tail of A_2 (passage to B_1).

The next example is a Barenboim interpretation of the second movement of Beethoven's Pathetique, which has the

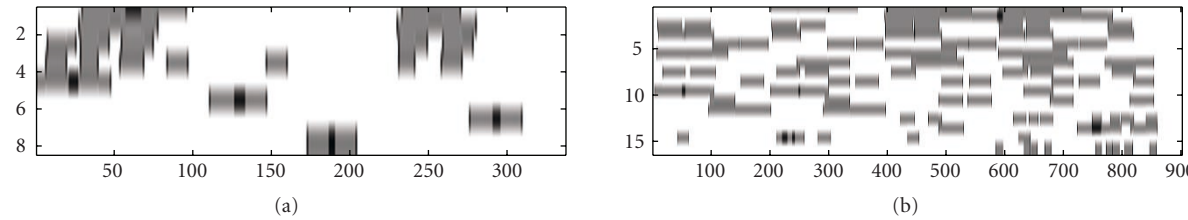

Figure 14: (a) Mendelssohn, "Wedding March," op. 21-7, conducted by Tate. (b) Ravel, "Bolero," conducted by Ozawa.

musical form $A_1A_2BA_3CA_4A_5D$. The interesting point of this piece is that the A-parts are variations of each other. For example, the melody in A_2 and A_4 is played one octave higher than the melody in A_1 and A_3. Furthermore, A_3 and A_4 are rhythmic variations of A_1 and A_2. Nevertheless, the correct global structure has been extracted; see Figure 13(b). The three clusters are in correspondence with $\mathcal{A}_1 \sim \{A_1A_2, A_4A_5\}$, $\mathcal{A}_3 \sim \{A_1, A_3, A_5\}$, and $\mathcal{A}_2 \sim \{A_1', A_2', A_3', A_4', A_5'\}$, where A_k' denotes a truncated version of A_k. Hence, the segments A_1, A_3, and A_5 are identified as a whole, whereas the other A-parts are identified only up to their tail. This is due to the fact that the tails of the A-parts exhibit some deviations leading to higher costs in the self-similarity matrix, as illustrated by Figure 13(b).

The popular song "I will survive" by Gloria Gaynor consists of an introduction I followed by eleven repetitions A_k, $1 \leq k \leq 11$, of the chorus. This highly repetitive structure is reflected by the secondary diagonals in the self-similarity matrix; see Figure 13(c). The segments exhibit variations not only with respect to the lyrics but also with respect to instrumentation and tempo. For example, some segments include a secondary voice in the violin, others harp arpeggios, or trumpet syncopes. The first chorus A_1 is played without percussion, whereas A_5 is a purely instrumental version. Also note that there is a significant ritardando in A_9 between seconds 150 and 160. In spite of these variations, the structure analysis algorithm works almost correctly. However, there are two artifacts that have not been ruled out by our strategy. Each chorus A_k can be split up into two subparts $A_k = A_k'A_k''$. The computed cluster $\mathcal{A}_1$ corresponds to the ten parts $A_{k-1}''A_k'A_k''$, $2 \leq k \leq 11$, revealing an overlap in the A''-parts. In particular, the extracted segments are "out of phase" since they start with subsegments corresponding to the A''-parts. This may be due to extreme variations in A_1' making this part dissimilar to the other A'-parts. Since A_1' constitutes the beginning of the extracted paths, it has been (mistakenly) truncated in step (3b) (pruning paths) of Section 4.

To check the robustness of our algorithm with respect to global and local tempo variations, we conducted a series of experiments with synthetically time-stretched audio signals (i.e., we changed the tempo progression without changing the pitch). As it turns out, there are no problems in identifying similar segments that exhibit global tempo variations of up to 50 percent as well as local tempo variations such as ritardandi and accelerandi. As an example, we consider the audio file corresponding to the part A_1A_2B of the Shostakovich example of Figure 2. From this, we generated two additional time-stretched variations: a faster version at 140 percent of the normal tempo and an accelerating version speeding up from 100 to 140 percent. The musical form of the concatenation of these three versions is $A_1A_2B_1A_3A_4B_2A_5A_6B_3$. This structure has been correctly extracted by our algorithm; see Figure 13(d). The correspondences of the two resulting clusters are $\mathcal{A}_1 \sim \{A_1A_2B_1, A_3A_4B_2, A_5A_6B_3\}$ and $\mathcal{A}_2 \sim \{A_1, A_2, A_3, A_4, A_5, A_6\}$.

Next, we discuss an example with a musically more complicated structure. This will also illustrate some problems typically appearing in automatic structure analysis. The "Wedding March" by Mendelssohn has the musical form

$$\begin{aligned} &A_1B_1A_2B_2C_1B_3C_2B_4\ D_1D_2E_1D_3E_1D_4 \cdots \\ &\cdots B_5\ F_1G_1G_2H_1\ A_3B_6C_3B_7A_4\ I_1I_2J_1. \end{aligned} \tag{10}$$

Furthermore, each segment B_k for $1 \leq k \leq 7$ has a substructure $B_k = B_k'B_k''$ consisting of two musically similar subsegments B_k' and B_k''. However, the B''-parts reveal significant variations even at the note level. Our algorithm has computed seven clusters, which are arranged according to the lengths of their support; see Figure 14(a). Even though not visible at first glance, these clusters represent most of the musical structure accurately. Manual inspection reveals that the cluster segments correspond, up to some tolerance, to the musical parts as follows:

$$\begin{aligned} \mathcal{A}_1 &\sim \{B_2C_1B_3', B_3C_2B_4', B_6C_3B_7'\}, \\ \mathcal{A}_2 &\sim \{B_2C_1B_3+, B_6C_3B_7+\}, \\ \mathcal{A}_3 &\sim \{B_1, B_2, B_3, B_6, B_7\}, \\ \mathcal{A}_4 &\sim \{B_1', B_2', B_3', B_4', B_5', B_6', B_7'\}, \\ \mathcal{A}_5 &\sim \{A_1B_1A_2, A_2B_2+\}, \\ \mathcal{A}_6 &\sim \{D_2E_1D_3, D_3E_2D_4\}, \\ \mathcal{A}_7 &\sim \{G_1, G_2\}, \\ \mathcal{A}_8 &\sim \{I_1, I_2\}. \end{aligned} \tag{11}$$

In particular, all seven B'-parts (truncated B-parts) are represented by cluster $\mathcal{A}_4$, whereas $\mathcal{A}_3$ contains five of the seven B-parts. The missing and truncated B-parts can be explained as in the Beethoven example of Figure 13(b). Cluster $\mathcal{A}_1$ reveals the similarity of the three C-parts, which are enclosed between the B- and B'-parts known from $\mathcal{A}_3$ and $\mathcal{A}_4$. The A-parts, an opening motif, have a duration of less than 8 seconds—too short to be recognized by our algorithm as a separate cluster. Due to the close harmonic relationship of the A-parts with the tails of the B-parts and the heads of the C-parts, it is hard to exactly determine the boundaries of these parts. This leads to clusters such as $\mathcal{A}_2$ and $\mathcal{A}_5$, whose segments enclose several parts or only fragments of some parts (indicated by the "+" sign). Furthermore, the segments of cluster $\mathcal{A}_6$ enclose several musical parts. Due to the overlap in D_3, one can derive the similarity of D_2, D_3, and D_4 as well as the similarity of E_1 and E_2. The D- and E-parts are too short (less than 10 seconds) to be detected as separate clusters. This also explains the undetected part D_1. Finally, the clusters $\mathcal{A}_7$ and $\mathcal{A}_8$ correctly represent the repetitions of the G- and I-parts, respectively.

Another complex example, in particular with respect to the occurring variations, is Ravel's Bolero, which has the musical form $D_1D_2D_3D_4A_9B_9C$ with $D_k = A_{2k-1}A_{2k}B_{2k-1}B_{2k}$ for $1 \leq k \leq 4$. The piece repeats two tunes (corresponding to the A- and B-parts) over and over again, each time played in a different instrumentation including flute, clarinet, bassoon, saxophone, trumpet, strings, and culminating in the full orchestra. Furthermore, the volume gradually grows from quiet pianissimo to a vehement fortissimo. Note that playing an instrument in piano or in fortissimo not only makes a difference in volume but also in the relative energy distribution within the chroma bands, which is due to effects such as noise, vibration, and reverberation. Nevertheless, the CENS features absorb most of the resulting variations. The extracted clusters represent the global structure up to a few missing segments; see Figure 14(b). In particular, the cluster $\mathcal{A}_3 \sim \{A_k- \mid 1 \leq k \leq 9\}$ correctly identifies all nine A-parts in a slightly truncated form (indicated by the "$-$" sign). Note that the truncation may result from step (2) (merging step) of Section 5, where path inconsistencies are ironed out by segment intersections. The cluster $\mathcal{A}_4$ correctly identifies the full-size A-parts with only part A_4 missing. Here, an additional transitivity step might have helped to perfectly identify all nine A-parts in full length. The similarity of the B-parts is reflected by $\mathcal{A}_5$, where only part B_9 is missing. All other clusters reflect superordinate similarity relations (e.g., $\mathcal{A}_1 \sim \{A_3A_4B_3, A_5A_6B_5, A_7A_8B_7\}$ or $\mathcal{A}_2 = \{D_3+, D_4+\}$) or similarity relations of smaller fragments.

For other pieces of music—we manually analyzed the results for about 100 pieces—our structure analysis algorithm typically performs as indicated by the above examples and the global repetitive structure can be recovered to a high degree. We summarize some typical problems associated with the extracted similarity clusters. Firstly, some clusters consist of segments that only correspond to fragments or truncated versions of musical parts. Note that this problem is not only due to algorithmic reasons such as the inconsistencies stemming from inaccurate path relations but also due to musical reasons such as extreme variations in tails of musical parts. Secondly, the set of extracted clusters is sometimes redundant as in the case of the Bolero—some clusters almost coincide while differing only by a missing part and by a slight shift and length difference of their respective segments. Here, a higher degree of transitivity and a more involved merging step in Section 5 could help to improve the overall result. (Due to the inconsistencies, however, a higher degree of transitivity may also degrade the result in other cases.) Thirdly, the global structure is sometimes not given explicitly but is somehow hidden in the clusters. For example, the similarity of the B-parts in the Chopin example results from "subtracting" the segments corresponding to the A-parts given by $\mathcal{A}_2$ from the segments of $\mathcal{A}_1$. Or, in the Mendelssohn example, the similarity of the D- and E-parts can be derived from cluster $\mathcal{A}_6$ by exploiting the overlap of the segments in a subsegment corresponding to part D_3. It seems promising to exploit such overlap relations in combination with a subtraction strategy to further improve the cluster structure. Furthermore, we expect an additional improvement in expressing the global structure by means of some hierarchical approach as discussed in Section 7.

6.2. Running time behavior

In this section, we discuss the running time behavior of the MATLAB implementation of our structure analysis algorithm. Tests were run on an Intel Pentium IV, 3.6 GHz, with 2 GB RAM under Windows 2000. Table 2 shows the running times for several pieces sorted by duration.

The first step of our algorithm consists of the extraction of robust audio features; see Section 2. The running time to compute the CENS feature sequence is linear in the duration of the audio file under consideration—in our tests roughly one third of the duration of the piece; see the third column of Table 2. Here, the decomposition of the audio signal into the 88 frequency bands as described in Section 2.1 constitutes the bottleneck of the feature extraction, consuming far more than 99% of the entire running time. The subsequent computations to derive the CENS features from the filter subbands only take a fraction of a second even for long pieces such as Ravel's Bolero. In view of our experiments, we computed the chroma features of Section 2.1 at a resolution of 10 Hz for each piece in our music database and stored them on hard disk, making them available for the subsequent steps irrespective of the parameter choice made in Sections 4 and 5.

The time and space complexity to compute a self-similarity matrix $\mathcal{S}$ is quadratic in the length N of the feature sequence. This makes the usage of such matrices infeasible for large N. Here, our strategy is to use coarse CENS features, which not only introduces a high degree of robustness towards admissible variations but also keeps the feature resolution low. In the above experiments, we used CENS[41,10]-features with a sampling rate of 1 Hz. Furthermore, incorporating the desired invariances into the features itself allows us to use a local distance measure based on the inner product that can be evaluated by a computationally inexpensive

Table 2: Running time behavior of the overall structure analysis algorithm. All time durations are measured in seconds. The columns indicate the respective piece of music, the duration of the piece, the running time to compute the CENS features (Section 2), the running time to compute the self-similarity matrix (Section 3), the running time for the path extraction (Section 4), the number of extracted paths, and the running time for the clustering algorithm (Section 5).

Piece	Length	CENS	$\mathcal{S}_{16}^{\min}[41, 10]$	Path Extr.	#(paths)	Clustering
Chopin, "Tristesse," Figure 13(a)	173.1	54.6	0.20	0.06	3	0.17
Gaynor, "I will survive," Figure 13(c)	200.0	63.0	0.25	0.16	24	0.33
Brahms, "Hungarian Dance," Figure 1	204.1	64.3	0.31	0.09	7	0.19
Shostakovich, "Waltz," Figure 2	223.6	70.5	0.34	0.09	6	0.20
Beethoven, "Pathetique 2nd," Figure 13(b)	320.0	100.8	0.66	0.15	9	0.21
Mendelssohn, "Wedding March," Figure 14(a)	336.6	105.7	0.70	0.27	17	0.27
Schubert, "Unfinished 1st" Figure 15(a)	900.0	282.1	4.40	0.85	10	0.21
Ravel, "Bolero," Figure 14(b)	901.0	282.7	4.36	5.53	71	1.05
2× "Bolero"	1802.0		17.06	84.05	279	9.81
3× "Bolero"	2703.0		37.91	422.69	643	97.94

algorithm. This affords an efficient computation of $\mathcal{S}$ even for long pieces of up to 45 minutes of duration; see the fourth column of Table 2. For example, in case of the Bolero, it took 4.36 seconds to compute $\mathcal{S}_{16}^{\min}[41, 10]$ from a feature sequence of length $N = 901$, corresponding to 15 minutes of audio. Tripling the length N by using a threefold concatenation of the Bolero results in a running time of 37.9 seconds, showing an increase by a factor of nine.

The running time for the path extraction algorithm as described in Section 4 mainly depends on the structure of the self-similarity matrix below the threshold C_{ad} (rather than on the size of the matrix); see the fifth column of Table 2. Here, crucial parameters are the number as well as the lengths of the path candidates to be extracted, which influences the running time in a linear fashion. Even for long pieces with a very rich path structure—as is the case for the Bolero—the running time of the path extraction is only a couple of seconds.

Finally, the running time of the clustering algorithm of Section 5 is negligible; see the last column of Table 2. Only for a very large (and practically irrelevant) number of paths, the running time seems to increase significantly.

Basically, the overall performance of the structure analysis algorithm depends on the feature extraction step, which depends linearly on the input size.

6.3. Modulation

It is often the case, in particular for classical music, that certain musical parts are repeated in another key. For example, the second theme in the exposition of a sonata is often repeated in the recapitulation transposed by a fifth (i.e., shifted by seven semitones upwards). To account for such modulations, we have adopted the idea of Goto [4], which is based on the observation that the twelve cyclic shifts of a 12-dimensional chroma vector naturally correspond to the twelve possible modulations. In [4], similarity clusters (called line segment groups) are computed for all twelve modulations separately, which are then suitably merged in a post-processing step. In contrast to this, we incorporate all modulations into a single self-similarity matrix, which then allows to perform a singly joint path extraction and clustering step only. The details of the modulation procedure are as follows. Let $\sigma : \mathbb{R}^{12} \to \mathbb{R}^{12}$ denote the *cyclic shift* defined by

$$\sigma\left((v_1, v_2, \ldots, v_{12})^\top\right) := (v_2, \ldots, v_{12}, v_1)^\top \tag{12}$$

for $\vec{v} := (v_1, \ldots, v_{12})^\top \in \mathbb{R}^{12}$. Then, for a given audio data stream with CENS feature sequence $\vec{V} := (\vec{v}^1, \vec{v}^2, \ldots, \vec{v}^N)$, the *i-modulated self-similarity matrix* $\sigma^i(\mathcal{S})$ is defined by

$$\sigma^i(\mathcal{S})(n, m) := d(\vec{v}^n, \sigma^i(\vec{v}^m)), \tag{13}$$

$1 \leq n, m \leq N$. $\sigma^i(\mathcal{S})$ describes the similarity relations between the original audio data stream and the audio data stream modulated by i semitones, $i \in \mathbb{Z}$. Obviously, one has $\sigma^{12}(\mathcal{S}) = \mathcal{S}$. Taking the minimum over all twelve modulations, we obtain the *modulated self-similarity matrix* $\sigma^{\min}(\mathcal{S})$ defined by

$$\sigma^{\min}(\mathcal{S})(n, m) := \min_{i \in [0:11]} (\sigma^i(\mathcal{S})(n, m)). \tag{14}$$

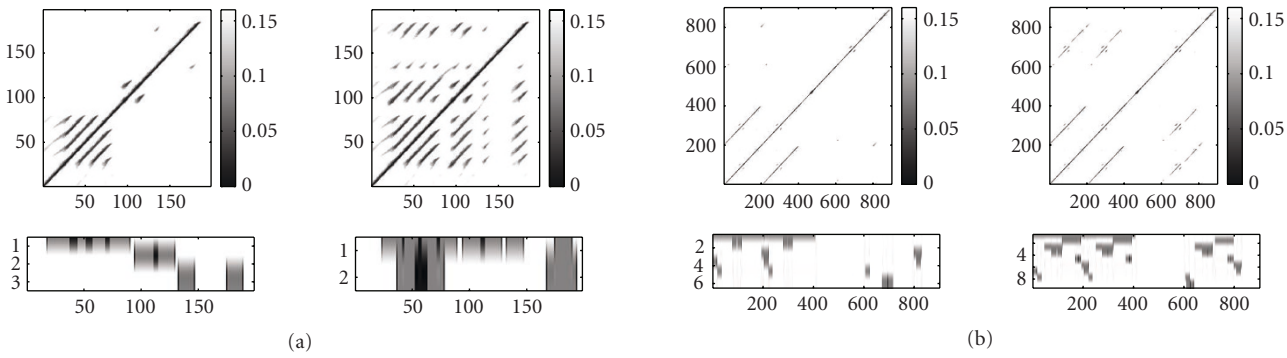

FIGURE 15: (a) Zager & Evans, "In the year 2525." Left: $\mathcal{S}$ with the resulting similarity clusters. Right: $\sigma^{\min}(\mathcal{S})$ with the resulting similarity clusters. The parameters are fixed as described in Section 6.1. (b) Schubert, "Unfinished," first movement, D 759, conduced by Abbado. Left and right parts are analogous to (a).

Furthermore, we store the minimizing shift indices in an additional N-square matrix I:

$$I(n,m) := \underset{i\in[0:11]}{\arg\min}(\sigma^i(\mathcal{S})(n,m)). \tag{15}$$

Analogously, one defines $\sigma^{\min}(\mathcal{S}_L^{\min}[w,q])$. Now, replacing the self-similarity matrix by its modulated version, one can proceed with the structure analysis as described in Sections 4 and 5. The only difference is that in step (1) of the path extension (Section 4) one has to ensure that each path $P = (p_1, p_2, \ldots, p_K)$ consists of links exhibiting the same modulation index: $I(p_1) = I(p_2) = \cdots = I(p_K)$.

We illustrate this procedure by means of two examples. The song "In the year 2525" by Zager & Evans is of the musical form $AB_1^0B_2^0B_3^0B_4^0CB_5^1B_6^1DB_7^2EB_8^2F$, where the chorus, the B-part, is repeated 8 times. Here, B_5^1 and B_6^1 are modulations by one semitone and B_7^2 and B_8^2 are modulations of the parts B_1^0 to B_4^0 by two semitones upwards. Figure 15(a) shows the similarity clusters derived from the structure analysis based on $\mathcal{S} = \mathcal{S}_{16}^{\min}[41, 10]$. Note that the modulated parts are separated into different clusters corresponding to $\mathcal{A}_1 \sim \{B_1^0, B_2^0, B_3^0, B_4^0\}$, $\mathcal{A}_2 \sim \{B_5^1, B_6^1\}$, and $\mathcal{A}_3 \sim \{B_7^2, B_8^2\}$. In contrast, the analysis based on $\sigma^{\min}(\mathcal{S})$ leads to a cluster $\mathcal{A}_1$ corresponding to all eight B-parts.

As a second example, we consider an Abbado recording of the first movement of Schubert's "Unfinished." This piece, which is composed in the sonata form, has the rough musical form $A_1^0B_1^0C_1^0A_2^0B_2^0C_2^0D\widetilde{A_3}B_3^7C_3^4E$, where $A_1^0B_1^0C_1^0$ corresponds to the exposition, $A_2^0B_2^0C_2^0$ to the repetition of the exposition, D to the development, $\widetilde{A_3}B_3^7C_3^4$ to the recapitulation, and E to the coda. Note that the B_1^0-part of the exposition is repeated up a fifth as B_3^7 (shifted by 7 semitones upwards) and the C_1^0-part is repeated up a third as C_3^4 (shifted by 4 semitones upwards). Furthermore, the A_1^0-part is repeated as $\widetilde{A_3}$, however in form of a multilevel transition from the tonic to the dominant. Again the structure is revealed by the analysis based on $\sigma^{\min}(\mathcal{S})$, where one has, among others, the correspondences $\mathcal{A}_1 \sim \{A_1^0B_1^0C_1^0, A_2^0B_2^0C_2^0\}$, $\mathcal{A}_2 \sim \{B_1^0, B_2^0, B_3^7\}$, and $\mathcal{A}_3 \sim \{C_1^0, C_2^0, C_3^4\}$. The other clusters correspond to further structures on a finer level.

Finally, since the modulated similarity matrix $\sigma^{\min}(\mathcal{S})$ is derived from the twelve i-modulated matrices $\sigma^i(\mathcal{S})$, $i \in [0:11]$, the resulting running time to compute $\sigma^{\min}(\mathcal{S})$ is roughly twelve times longer than the time to compute $\mathcal{S}$. For example, it took 51.4 seconds to compute $\sigma^{\min}(\mathcal{S})$ for Schubert's "Unfinished" as opposed to 4.4 seconds needed to compute $\sigma(\mathcal{S})$ (cf. Table 2).

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have described a robust and efficient algorithm that extracts the repetitive structure of an audio recording. As opposed to previous methods, our approach is robust to significant variations in the repetitions concerning instrumentation, execution of note groups, dynamics, articulation, modulation, and tempo. For the first time, detailed experiments have been conducted for a wide range of Western classical music. The results show that the extracted audio structures often closely correspond to the musical form of the underlying piece, even though no a priori knowledge of the music structure has been used. In our approach, we converted the audio signal into a sequence of coarse, harmony-related CENS features. Such features are well suited to characterize pieces of Western classical music, which often exhibit prominent harmonic progressions. Furthermore, instead of relying on complicated and delicate path extraction algorithms, we suggested a different approach by taking care of local variations at the feature and similarity measure levels. This way we improved the path structure of the self-similarity matrix, which then allowed for an efficient robust path extraction.

To obtain a more comprehensive representation of audio structure, obvious extensions of this work consist of combining harmony-based features with other types of features describing the rhythm, dynamics, or timbre of music.

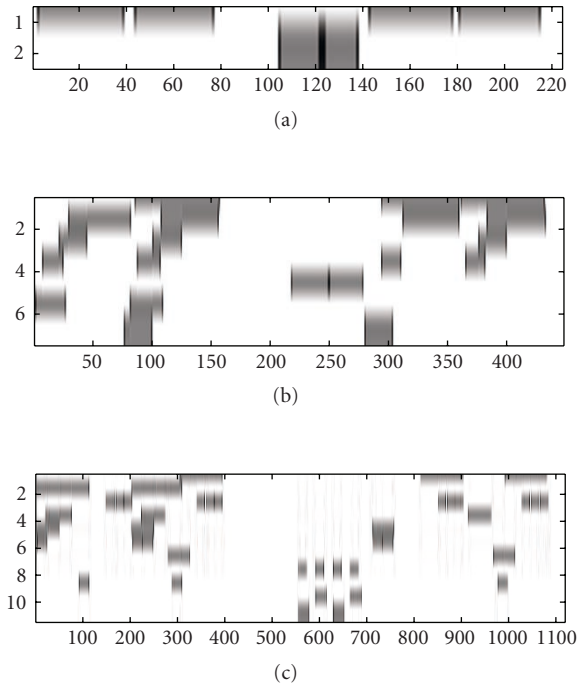

FIGURE 16: Similarity clusters for the Shostakovich example of Figure 2 resulting from a structure analysis using (a) $\mathcal{S}_{16}^{\min}[41, 10]$, (b) $\mathcal{S}_{16}^{\min}[21, 5]$, and (c) $\mathcal{S}_{16}[9, 2]$.

Another extension regards the hierarchical nature of music. So far, we looked in our analysis for repetitions at a global to intermediary levels corresponding to segments of at least 15–20 seconds of duration. As has also been noted by other researches, music structure can often be expressed in a hierarchical manner, starting with the coarse musical form and ascending to finer substructures such as repeating themes and motifs. Here, one typically allows larger variations in the analysis of coarser structures than in the analysis of finer structures. For future work, we suggest a hierarchical approach to structure analysis by simultaneously computing and combining structural information at various temporal resolutions. To this end, we conducted first experiments based on the self-similarity matrices $\mathcal{S}_{16}^{\min}[41, 10]$, $\mathcal{S}_{16}^{\min}[21, 5]$, and $\mathcal{S}_{16}[9, 2]$ with corresponding feature resolutions of 1 Hz, 2 Hz, and 5 Hz, respectively. The resulting similarity clusters are shown in Figure 16 for the Shostakovich example. Note that the musical form $A_1A_2BC_1C_2A_3A_4D$ has been correctly identified at the low resolution level; see Figure 16(a). Increasing the feature resolution has two effects: on the one hand, finer repetitive substructures are revealed, as illustrated by Figure 16(c). On the other hand, the algorithm becomes more sensitive towards local variations, resulting in fragmentation and incompleteness of the coarser structures. One very difficult problem to be solved is to integrate the extracted similarity relations at all resolutions into a single hierarchical model that best describes the musical structure.

ACKNOWLEDGMENTS

We would like to thank Michael Clausen and Tido Röder for helpful discussions and comments.

REFERENCES

[1] 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.

[2] M. Cooper and J. Foote, "Automatic music summarization via similarity analysis," in *Proceedings of 3rd International Conference on Music Information Retrieval (ISMIR '02)*, Paris, France, October 2002.

[3] R. Dannenberg and N. Hu, "Pattern discovery techniques for music audio," in *Proceedings of 3rd International Conference on Music Information Retrieval (ISMIR '02)*, Paris, France, October 2002.

[4] M. Goto, "A chorus-section detecting method for musical audio signals," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '03)*, vol. 5, pp. 437–440, Hong Kong, April 2003.

[5] L. Lu, M. Wang, and H.-J. Zhang, "Repeating pattern discovery and structure analysis from acoustic music data," in *Proceedings of the 6th ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR '04)*, pp. 275–282, New York, NY, USA, October 2004.

[6] N. C. Maddage, C. Xu, M. S. Kankanhalli, and X. Shao, "Content-based music structure analysis with applications to music semantics understanding," in *proceedings of the 12th ACM International Conference on Multimedia*, pp. 112–119, New York, NY, USA, October 2004.

[7] G. Peeters, A. L. Burthe, and X. Rodet, "Toward automatic music audio summary generation from signal analysis," in *Proceedings of 3rd International Conference on Music Information Retrieval (ISMIR '02)*, pp. 94–100, Paris, France, October 2002.

[8] J. Foote, "Visualizing music and audio using selfsimilarity," in *Proceedings of the 7th ACM International Conference on Multimedia (MM '99)*, pp. 77–80, Orlando, Fla, USA, October-November 1999.

[9] M. A. Bartsch and G. H. Wakefield, "To catch a chorus: using chroma-based representations for audio thumbnailing," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA '01)*, pp. 15–18, New Paltz, NY, USA, October 2001.

[10] B. Logan and S. Chu, "Music summarization using key phrases," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '00)*, vol. 2, pp. 749–752, Istanbul, Turkey, June 2000.

[11] C. Xu, N. C. Maddage, and X. Shao, "Automatic music classification and summarization," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 441–450, 2005.

[12] W. Chai, "Structural analysis of musical signals via pattern matching," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '03)*, vol. 5, pp. 549–552, Hong Kong, April 2003.

[13] W. Chai and B. Vercoe, "Music thumbnailing via structural analysis," in *Proceedings of the ACM International Multimedia Conference and Exhibition (MM '03)*, pp. 223–226, Berkeley, Calif, USA, November 2003.

[14] M. Goto, "SmartMusicKIOSK: music listening station with chorus-search function," in *Proceedings of the Annual ACM Symposium on User Interface Softaware and Technology (UIST '03)*, pp. 31–40, Vancouver, BC, Canada, November 2003.

[15] F. Kurth, M. Müller, D. Damm, C. Fremerey, A. Ribbrock, and M. Clausen, "Syncplayer—an advanced system for content-based audio access," in *Proceedings of 6th International Conference on Music Information Retrieval (ISMIR '05)*, London, UK, September 2005.

[16] G. Tzanetakis, A. Ermolinskyi, and P. Cook, "Pitch histograms in audio and symbolic music information retrieval," in *Proceedings of 3rd International Conference on Music Information Retrieval (ISMIR '02)*, Paris, France, October 2002.

[17] J. G. Proakis and D. G. Manolakis, *Digital Signal Processsing*, Prentice Hall, Englewood Cliffs, NJ, USA, 1996.

[18] M. Müller, F. Kurth, and M. Clausen, "Audio matching via chroma-based statistical features," in *Proceedings of 6th International Conference on Music Information Retrieval (ISMIR '05)*, London, UK, September 2005.

Meinard Müller studied mathematics and computer science at Bonn University, Germany, where he received both a Master's degree in mathematics and the Doctor of Natural Sciences (Dr. rer. nat.) in 1997 and 2001, respectively. In 2002/2003, he conducted postdoctoral research in combinatorics at the Mathematical Department of Keio University, Japan. Currently, he is a Member of the Multimedia Signal Processing Group, Bonn University, working as a Researcher and Assistant Lecturer. His research interests include digital signal processing, multimedia information retrieval, computational group theory, and combinatorics. His special research topics include audio signal processing, computational musicology, analysis of 3D motion capture data, and content-based retrieval in multimedia documents.

Frank Kurth studied computer science and mathematics at Bonn University, Germany, where he received both a Master's degree in computer science and the degree of a Doctor of Natural Sciences (Dr. rer. nat.) in 1997 and 1999, respectively. Currently, he is with the Multimedia Signal Processing group at Bonn University, where he is working as an Assistant Lecturer. Since his Habilitation (postdoctoral lecture qualification) in computer science in 2004, he holds the title of a Privatdozent. His research interests include audio signal processing, fast algorithms, multimedia information retrieval, and digital libraries for multimedia documents. Particular fields of interest are music information retrieval, fast content-based retrieval, and bioacoustical pattern matching.