Towards Cross-Version Harmonic Analysis of Music

Sebastian Ewert, *Student Member, IEEE*, Meinard Müller, *Member, IEEE*, Verena Konz, Daniel Müllensiefen, and Geraint A. Wiggins

***Abstract*—For a given piece of music, there often exist multiple versions belonging to the symbolic (e.g., MIDI representations), acoustic (audio recordings), or visual (sheet music) domain. Each type of information allows for applying specialized, domain-specific approaches to music analysis tasks. In this paper, we formulate the idea of a *cross-version analysis* for comparing and/or combining analysis results from different representations. As an example, we realize this idea in the context of harmonic analysis to automatically evaluate MIDI-based chord labeling procedures using annotations given for corresponding audio recordings. To this end, one needs reliable synchronization procedures that automatically establish the musical relationship between the multiple versions of a given piece. This becomes a hard problem when there are significant local deviations in these versions. We introduce a novel late-fusion approach that combines different alignment procedures in order to identify reliable parts in synchronization results. Then, the cross-version comparison of the various chord labeling results is performed only on the basis of the reliable parts. Finally, we show how inconsistencies in these results across the different versions allow for a quantitative and qualitative evaluation, which not only indicates limitations of the employed chord labeling strategies but also deepens the understanding of the underlying music material.**

***Index Terms*—Alignment, chord recognition, music information retrieval, music synchronization.**

I. Introduction

A MUSICAL work can be described in various ways using different representations. Symbolic formats (e.g., MusicXML, MIDI, Lilypond) conventionally describe a piece of music by specifying important musical parameters like pitch, rhythm, and dynamics. Interpreting these parameters as part of a musical performance leads to an acoustical representation that can be described by audio formats encoding the physical properties of sound (e.g., WAV, MP3). Depending on the type of representation, some musical properties are directly accessible while others may be implicit or even absent. For example, extracting pitch information from a MIDI file is straightforward, while extracting the same information from an audio file is a nontrivial task. On the other hand, while timbre and other complex musical properties are richly represented in an audio recording, the corresponding options in a MIDI file are very limited. Thus, an audio recording is close to be expressively complete in the sense that it represents music close to what is heard by a listener [1]. On the other hand, a MIDI representation contains structural information in an explicit form, but usually does not encode expressive information. Such differences between music representations allow for conceptually very different approaches to higher-level music analysis tasks such as melody extraction or structure analysis. Typically, each approach has intrinsic domain-specific strengths and weaknesses.

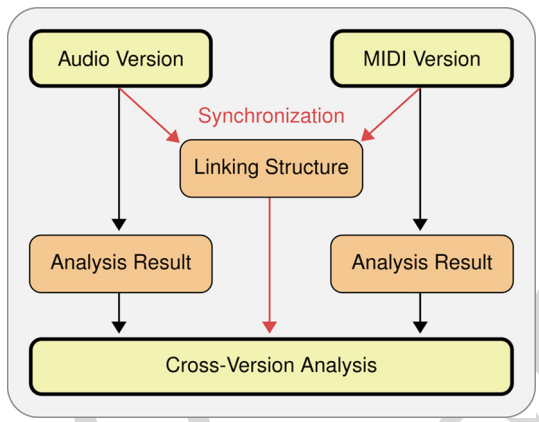

Fig. 1. Cross-version music analysis based on synchronization techniques.

As our main conceptual contribution, we formulate the idea of a *cross-version analysis* for comparing and/or combining analysis results from different domains. Our main idea is to incorporate music synchronization techniques to temporally align music representations across the different domains (see Fig. 1). Here, *music synchronization* refers to a procedure which, for a given position in one representation of a piece of music, determines the corresponding position within another representation. In general, a cross-version approach presents many varied opportunities to compare methods across different domains or to create methods that unite the domain-specific strengths while attenuating the weaknesses. In this paper, we present an instance of such a cross-version analysis procedure, considering the task of automated chord labeling. Here, the objective is to induce the harmonic structure of a piece of music. The output of a chord labeling process is a sequence of chord labels with time stamps, either in musical time (i.e., in bars and beats) or in physical time measured in seconds. Because chord progressions

Manuscript received February 11, 2011; revised August 08, 2011 and November 21, 2011; accepted February 20, 2012. The work of S. Ewert was supported by the German Research Foundation (DFG CL 64/6-1). The work of M. Müller and V. Konz was supported by Cluster of Excellence on Multimodal Computing and Interaction (MMCI). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Svetha Venkatesh.

S. Ewert is with the Multimedia Signal Processing Group, Department of Computer Science III, University of Bonn, Bonn, Germany (e-mail: ewerts@iai.uni-bonn.de).

M. Müller and V. Konz are with the Saarland University and the Max-Planck Institut Informatik, Saarbrücken, Germany (e-mail: meinard@mpi-inf.mpg.de; vkonz@mpi-inf.mpg.de).

D. Müllensiefen is with the Department of Psychology, Goldsmiths, University of London, London, U.K. (e-mail: d.mullensiefen@gold.ac.uk).

G. A. Wiggins is with the Centre for Digital Music, School of Electronic Engineering and Computer Science, Queen Mary, University of London, London, U.K. (e-mail: geraint.wiggins@eecs.qmul.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2012.2190047

describe the structure of a piece in a very musical and compact way, they often form the basis of musicological analyses and further automatic music processing applications. In particular, we demonstrate our cross-version approach by evaluating two state-of-the-art MIDI-based chord labelers using a ground truth originally created for audio recordings. Using synchronization techniques, we can compare chord labels obtained from different procedures (automated or manual) and from different music representations (MIDI or audio). Having a unified view of the analysis results not only allows for an automated evaluation of the various analysis procedures but also deepens the understanding of the algorithms's behavior and the properties of the underlying music material.

This simple, yet powerful concept is not restricted to harmony analysis or music data. It is equally applicable to general multimedia data, where several versions or representations are given for an object to be analyzed. For example, a robust alignment between given music recordings and lyrics allows for creating karaoke applications [2], [3] or for combining genre classification results across the audio and the text domain [4]. Similarly, combining web-based text information, symbolic music representations and audio data was shown to lead to significant performance gains for general music classification tasks [5]. As another example, in motion capturing, an actor is typically recorded from different angles resulting in several video streams showing the same scene from different perspectives. Here, the corresponding audio tracks can be used to synchronize the various videos streams, which facilitates a multi-version analysis of the given scene [6].

Independent of the application scenario, the alignment of data from different domains depends crucially, for reliability, on robust synchronization techniques. However, in a musical context, synchronization becomes a hard problem when the music representations to be aligned reveal significant differences not only in tempo, instrumentation, or dynamics but also in structure or polyphony [7], [8]. Because of the complexity and diversity of music data, one cannot expect to find a universal synchronization algorithm that yields good results for all musical contexts and kinds of input data. Therefore, we present a novel method that allows for the automatic identification of the reliable parts of synchronization results. Instead of relying on one single strategy, our idea is to employ a late-fusion approach that combines several types of conceptually different alignment strategies within an extensible framework. Looking for consistencies and inconsistencies across the synchronization results, our method automatically classifies the alignments locally as reliable or critical. Considering only the reliable parts yields a high-precision partial alignment.

Altogether, the main contributions of this paper are threefold. Firstly, the idea of a cross-version analysis is formulated—a concept that is applicable for general multimedia data. Secondly, a novel method allowing for a reliable partial synchronization of music data from different domains is presented. Thirdly, as an example application of our concept, a cross-version evaluation of symbolic chord labeling methods using audio-based manual annotations is discussed.

The remainder of the paper is organized as follows. We start by describing classical alignment procedures (Section II) and then introduce our late-fusion synchronization framework (Section III). In Section IV, we give a short overview of available chord labeling methods as well as a more detailed description of two state-of-the-art symbolic domain methods. In Section V, we present our evaluation while demonstrating how a cross-version visualization greatly deepens the understanding of the analysis results. Finally, conclusions and prospects for future work are given in Section VI. Parts of this work have been published in [9]. Related work is discussed in the respective sections.

II. Alignment Procedures

Most alignment and synchronization procedures basically proceed in three steps. In the first step, the data streams to be aligned are converted to a suitable feature representation. Then, a local cost measure is used to compare features from the two streams. In the final step, based on this comparison, the actual synchronization result is computed using an alignment strategy. For synchronizing a pair of MIDI and audio representations of a piece of music, chroma-based features in combination with contextual cost measures have proven to be suitable tools, which we introduce in Section II-A. Then, in the remainder of this section, we focus on the third step and describe three conceptually different alignment strategies: dynamic time warping (Section II-C), a recursive version of Smith-Waterman (Section II-D), and partial matching (Section II-E). While these three approaches share similar algorithmic roots (dynamic programming) and possess a close mathematical modeling (Section II-B), they produce fundamentally different types of alignments; see also Section II-F. It is one goal of this section to give a unifying view on these approaches while highlighting the conceptual differences. For relevant and related work, we refer to the respective sections.

A. Feature Representation and Cost Measure

To compare a MIDI file with an audio recording of the same song, we convert both representations into a common mid-level representation. Depending on the type of this representation, the comparison can be based on musical properties such as harmony, rhythm, or timbre. Here, we use chroma-based music features, which have turned out to be a powerful tool for relating harmony-based music [7], [10]. For details on how to derive chroma features from audio and MIDI files, we refer to [10] and [11]. In the subsequent discussion, we employ normalized 12-dimensional chroma features with a temporal resolution of 2 Hz (2 features per second). Such feature rates have also turned out to be suitable for related tasks such as audio matching [12] and cover song detection [13].

Let $V := (v^1, v^2, \ldots, v^N)$ and $W := (w^1, w^2, \ldots, w^M)$ be two chroma feature sequences. To relate two chroma vectors, we use the cosine distance defined by $c(v^n, w^m) = 1 - \langle v^n, w^m \rangle$ for normalized vectors. By comparing the features of the two sequences in a pairwise fashion, one obtains an $(N \times M)$-*cost matrix* C defined by $C(n, m) := c(v^n, w^m)$; see Fig. 2(a). Each tuple (n, m) is called a *cell* of the matrix. To increase the robustness of the overall alignment procedure, it is often beneficial to also include the local temporal evolution of the features in order to enhance the structural properties of a cost matrix. To this end,

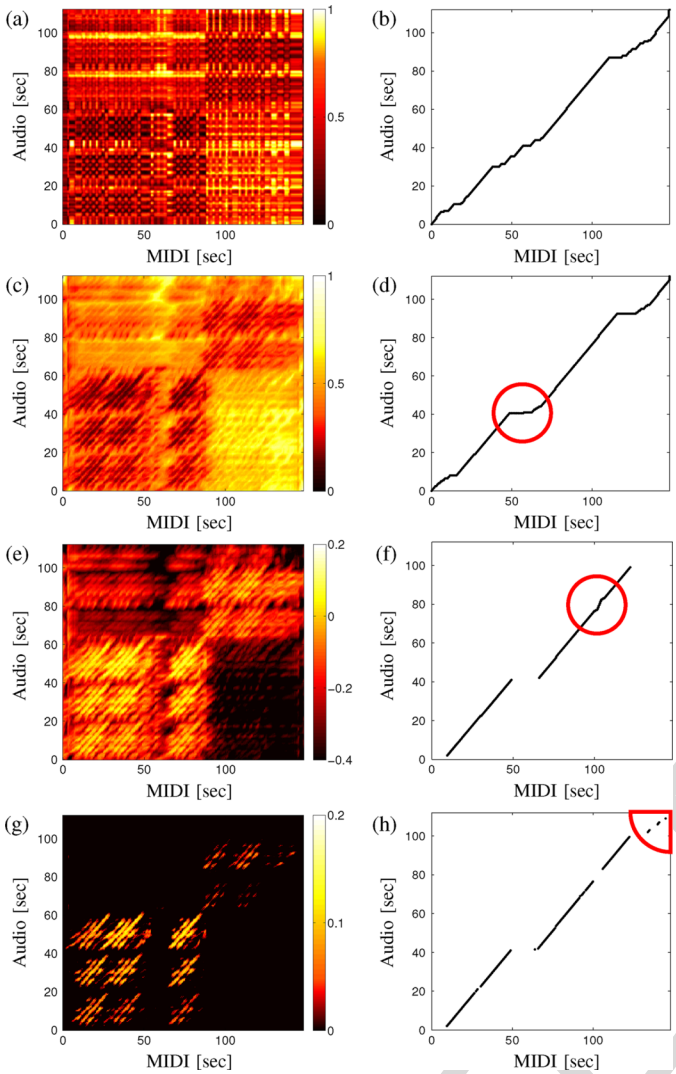

Fig. 2. Several techniques for the alignment of an audio recording (vertical axis) and a MIDI version (horizontal axis) of the song *And I Love Her* by the Beatles. The marked regions are further discussed in the text. (a) Chroma-based cost matrix. (b) Optimal global path obtained via DTW based on the chroma cost matrix. (c) Smoothed cost matrix C using $\lambda = 12$. (d) Optimal global path obtained via DTW based on matrix C. (e) Score matrix S. (f) Family of paths obtained via Smith-Waterman based on matrix S. (g) Thresholded score matrix $S_{\geq 0}$. (h) Optimal match obtained via partial matching based on matrix $S_{\geq 0}$.

Foote [14] proposed to average the cost values from a number of consecutive frames and to use that as the new cost value. This results in a smoothing effect of C. Müller and Kurth [15] extended these ideas by suggesting a contextual distance measure that allows for handling local tempo variations in the underlying audio recording. The enhancement procedure can be thought of as a multiple filtering of C along various directions given by gradients in a neighborhood of the gradient (1,1). We denote the smoothed cost matrix again by C. The degree of smoothing depends on a parameter λ, which specifies the number of consecutive frames taken into account for the filtering. The role of this parameter will be discussed in Section III-C. For an example, see Fig. 2(c).

B. Alignment Methods

We now introduce some common mathematical notations that are shared by all three alignment procedures to be discussed. Generally, an *alignment* between the feature sequences $V := (v^1, v^2, \ldots, v^N)$ and $W := (w^1, w^2, \ldots, w^M)$ is regarded as a set $\mathcal{A} \subseteq [1:N] \times [1:M]$, where $[1:N]$ is a shorthand for $\{1, 2, \ldots, N\}$. Here, each cell $\pi = (n, m) \in \mathcal{A}$ encodes a correspondence between the feature vectors v^n and w^m. By ordering its elements lexicographically $\mathcal{A}$ takes the form of a sequence, i.e., $\mathcal{A} = (\pi_1, \ldots, \pi_L)$ with $\pi_\ell = (n_\ell, m_\ell)$, $\ell \in [1:L]$. Additional constraints on the set ensure that only musically meaningful alignments are permitted. We say that the set $\mathcal{A}$ is *monotonic* if

$$n_1 \leq n_2 \leq \cdots \leq n_L \text{ and } m_1 \leq m_2 \leq \cdots \leq m_L.$$

Similarly, we say that $\mathcal{A}$ is *strictly monotonic* if

$$n_1 < n_2 < \cdots < n_L \text{ and } m_1 < m_2 < \cdots < m_L.$$

Note that the monotonicity condition reflects the requirement of faithful timing: if an event in V precedes a second one, this also should hold for the aligned events in W. A strictly monotonic set $\mathcal{A}$ will also be referred to as *match*, denoted by the symbol $\mathcal{M} = \mathcal{A}$. To ensure certain continuity conditions, we introduce step-size constraints by requiring

$$\gamma_{\ell+1} - \gamma_\ell \in \Sigma$$

for $\ell \in [1 : L-1]$, in which Σ denotes a set of admissible step sizes. A typical choice is $\Sigma = \Sigma_1 := \{(1,1), (1,0), (0,1)\}$ or $\Sigma = \Sigma_2 := \{(1,1), (2,1), (1,2)\}$. A set $\mathcal{A}$ that fulfills the step-size condition is also referred to as *path* denoted by the symbol $\mathcal{P} = \mathcal{A}$. Note that when using Σ_1, the set $\mathcal{A}$ also becomes monotonic allowing a relatively high degree of flexibility in the alignment path. Using Σ_2 instead typically results in more restricted alignments with additional slope constraints, which, on the positive side, often introduces a higher degree of robustness. As final constraint, the boundary condition

$$\gamma_1 = (1,1) \quad \text{and} \quad \gamma_L = (N, M)$$

ensures in combination with a step-size condition the alignment of V and W as a whole. If both the step-size as well as the boundary condition hold for a set $\mathcal{A}$, then $\mathcal{A}$ will be referred to as *global path* (or *warping path*) denoted by $\mathcal{G}$. Finally, a monotonic set $\mathcal{A}$ is referred to as *family of paths*, denoted by $\mathcal{F}$, if there exist paths $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_K$ with $\mathcal{F} = \mathcal{A} = \bigcup_{k \in [1:K]} \mathcal{P}_k$.

C. Dynamic Time Warping

If it is known a-priori that the two sequences to be aligned correspond to each other globally, then a global path is the correct alignment model. Here, classical *dynamic time warping* (DTW), which has originally been used to compare different speech patterns in automatic speech recognition [16], can be used to compute a global path. In this context, the *cost* of an alignment $\mathcal{A}$ is defined as $\sum_{\ell=1}^{L} C(n_\ell, m_\ell)$. Then, after fixing a set of admissible step-sizes Σ, DTW yields an optimal global

path having minimal cost among all possible global paths. For the subsequent discussion, we use $\mathrm{A}(s,t)$ to refer to the segment in the audio recording starting at s seconds and terminating at t seconds. Similarly, $\mathrm{M}(s,t)$ refers to a MIDI segment. So listening to $\mathrm{M}(55,65)$ of the song *And I Love Her* (used throughout Fig. 2) reveals a short bridge in the song. However, in the particular audio recording used here, the bridge is omitted. Since DTW always aligns the sequences as a whole, we find a musically inappropriate alignment between $\mathrm{A}(40,42$ and $\mathrm{M}(48,65)$; see also the marked region in Fig. 2(d). A similar observation can be made at the beginning and the end of the optimal global path. Here, the intro and outro in the audio recording deviate strongly from those in the MIDI version. In our experiments, we choose $\Sigma = \Sigma_1$ in the DTW alignment, since this leads to more flexibility in cases where the assumption of global correspondence between the sequences is violated.

D. Recursive Smith-Waterman

In general, using DTW in the case that elements in one sequence do not have suitable counterparts in the other sequence is problematic. In particular, in the presence of structural differences between the two sequences, this typically leads to misalignments. Therefore, if it is known a-priori that the two sequences to be aligned only partially correspond to each other, a path or a family of paths allows for a more flexible alignment than a global path.

To align two sequences that correspond only locally, one can deploy the *Smith-Waterman algorithm*—a well-known technique originally used in biological sequence analysis [17], [18]. In the music context, this algorithm has also been successfully used for the task of cover song identification [13]. Instead of using the concept of a cost matrix with the goal of finding a cost-minimizing alignment, one now uses the concept of a *score matrix* with the goal to find a score-maximizing alignment. To obtain a score matrix S from a cost matrix C, we fix a threshold $\tau > 0$ and set $S = \tau - C$. Fig. 2(e) shows a score matrix derived from the cost matrix shown in Fig. 2(c). The *score* of an alignment $\mathcal{A}$ is defined as $\sum_{\ell=1}^{L} S(n_\ell, m_\ell)$. Then, after fixing a set of admissible step-sizes Σ, the Smith-Waterman algorithm computes an optimal path having maximal score among all possible paths using a dynamic programming algorithm similar to DTW. Cells of $\mathcal{A}$ having negative score are often referred to as gaps, where one considers gap openings and gap extensions. Typically, such gaps are further penalized by introducing additional *gap-penalty parameters* [13], [18]. In our setting, for simplicity, we use a single gap parameter γ for openings as well as extensions. Then, this parameter can be realized by a subtraction of γ from all negative entries in the score matrix S. The role of the parameters τ and γ will be further investigated in Section III-C.

The original Smith-Waterman algorithm only delivers a single alignment path, which is often not enough to encode a suitable alignment. Therefore, we now introduce a novel recursive variant of the Smith-Waterman algorithm. First, we derive an optimal path $\mathcal{P}$ as described above; see Fig. 3(a). Then, we define two submatrices in the underlying score matrix S; see Fig. 3(b). The first matrix is defined by the cell (1,1) and the starting cell of $\mathcal{P}$, and the second matrix by the ending

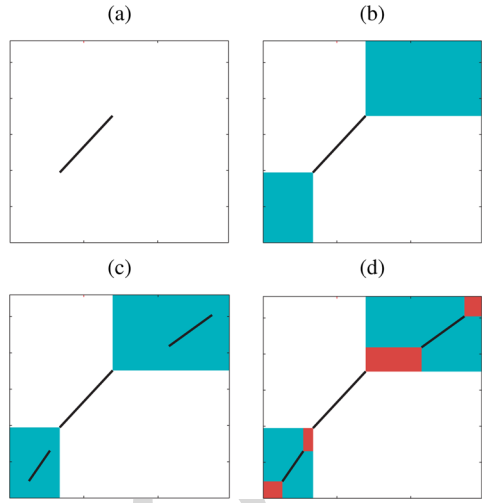

Fig. 3. First steps of our recursive Smith-Waterman variant. (a) Optimal path $\mathcal{P}$ derived via classical Smith-Waterman. (b) Submatrices defined via $\mathcal{P}$. (c) Result after the first recursion. Optimal paths have been derived from the submatrices. (d) New submatrices for the next recursive step are defined.

cell of $\mathcal{P}$ and the cell (N,M). For these submatrices, we call the Smith-Waterman algorithm recursively to derive another optimal path for each submatrix; see Fig. 3(c). These new paths define new submatrices on which Smith-Waterman is called again; see Fig. 3(d). This procedure is repeated until either the score of an optimal path or the size of a submatrix is below a given threshold. This results in a monotonic alignment set in form of a family of paths $\mathcal{F}$. Fig. 2(f) shows a family of two paths derived from the score matrix in Fig. 2(e) using our recursive Smith-Waterman variant. Using this method, the missing bridge in the audio as well as the different intros and outros in the audio and MIDI version are detected and, in this example, the recursive Smith-Waterman approach avoids the misalignment of the DTW case; see Fig. 2(d).

While this example highlights some of the strengths of the Smith-Waterman algorithm, it also illustrates one of its weaknesses. Listening to $\mathrm{A}(75,83)$ and $\mathrm{M}(99,107)$ reveals a solo improvisation which differs in the audio and MIDI versions, so they should not be aligned. Also, the corresponding area in the score matrix shows negative values. However, the Smith-Waterman algorithm aligns these two segments as part of the second path; see marked region in Fig. 2(f). The reason is that Smith-Waterman always tries to find the path with maximum score, where even a small number (relative to the total length of the path) of gaps are tolerated.

Opposed to DTW, we choose the more robust $\Sigma = \Sigma_2$ in the Smith-Waterman procedure. Here, the reason is that Smith-Waterman can better deal with local deviations in the two sequences to be aligned and therefore does not require the flexibility offered by Σ_1.

E. Partial Matching

As a third approach, we use a *partial matching* strategy, which gives the least constrained alignment [10], [18], [19]. Here, similar to the Smith-Waterman approach, the goal is to find an alignment that maximizes the score. However, in this case we require that the alignment is a match (strictly

monotonous alignment) without imposing any further step size conditions. Therefore, opposed to a score-maximizing path, there are no cells of negative score in a score-maximizing match. Thus, negative scores can be ignored completely and we therefore use the rectified version $S_{\geq 0}$, in which every negative entry in S is replaced by zero; see Fig. 2(g). Again, a score-maximizing match can be computed efficiently using dynamic programming. Fig. 2(h) shows an example of an optimal match computed via partial matching, based on the matrix shown in Fig. 2(g). Here, the misalignment of the solo segments $\mathrm{A}(75, 83)$ and $\mathrm{M}(99, 107)$ found in the Smith-Waterman case is not present. So partial matching, not enforcing any step-size or continuity conditions on the alignment, yields a more flexible alignment than the Smith-Waterman approach. However, in turn, this flexibility can also lead to spurious, inappropriate, and fragmented alignments, as can be seen in segments $\mathrm{A}(101, 110)$ and $\mathrm{M}(127, 147)$; see marked region in Fig. 2(h).

F. Concluding Remarks

In summary, one may think of two extremes: on the one hand, DTW relies on strong model assumptions, but works reliably in the case that these assumptions are fulfilled; on the other hand, partial matching offers a high degree of flexibility, but may lead to alignments being locally misguided or split into many fragments. The Smith-Waterman approach lies in between these two extremes.

Furthermore, alignment problems as discussed in this paper are closely related to tasks such as automated accompaniment [20], [21] and score following [22]. However, alignment strategies often employed in these fields such as hidden Markov models (HMMs) [23] and other graphical models [24], [25] are not further considered in the following. Such probabilistic methods usually require training data consisting of several different versions of the underlying audio material to identify statistical properties of the sound. Because only one audio version is available in our scenario, we have not incorporated such methods. Further discussion about the use of graphical models in alignment scenarios can be found in [23] and [24].

III. Consistency Alignment

As illustrated by the examples shown in Fig. 2, each synchronization strategy may contain satisfying as well as misguided parts. Therefore, with no definite a-priori knowledge about the input data, none of these alignment methods can in general guarantee a reliable and musically meaningful alignment. However, if several strategies with different design goals yield locally similar alignment results, then there is a high probability that these results are musically meaningful. Based on this simple idea, we present in Section III-A a novel late-fusion approach that combines several alignment procedures in order to identify passages in the MIDI and audio representations that can be reliably synchronized. Then, in Section III-B, we introduce a suitable quality measure which is employed in Section III-C to investigate the role of the parameters in our overall procedure.

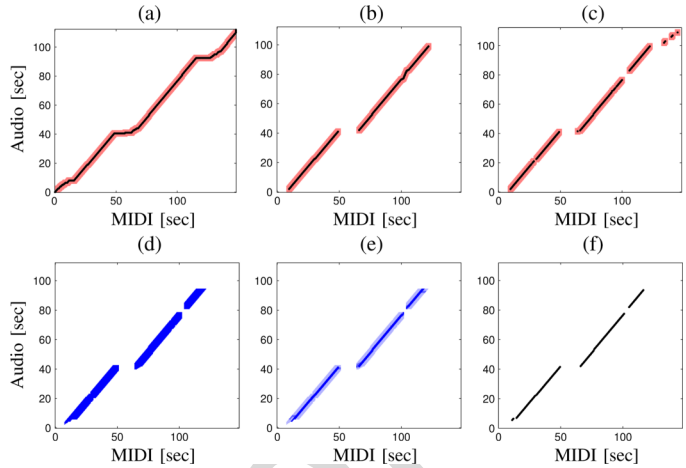

Fig. 4. Steps in our proposed method continuing the example shown in Fig. 2(a)–(c). Alignment (black) and corresponding augmented binary matrix (red and white) for the optimal global path (DTW), family of paths (Smith-Waterman), and the optimal match (partial matching). (d) Intersection matrix derived from (a)–(c). (e) Weighted intersection matrix. (f) Consistency alignment $\mathcal{C}$.

A. Proposed Method

Given a MIDI-audio pair for a song, we start by computing an optimal global path using DTW, a family of paths using recursive Smith-Waterman, and an optimal match using partial matching. Next, we convert each alignment into a binary matrix having the same size as the cost matrix C. Here, a cell in the matrix is set to one if it is contained in the corresponding alignment, and zero otherwise (in Fig. 2, the three alignments are already represented in this way). Next, we combine the three alignments using a late-fusion strategy to compute a kind of soft intersection. To this end, we augment the binary matrices by additionally setting every cell in the binary matrices to one if they are in a neighborhood of an alignment cell; see Figs. 4(a)–(c). Without such a tolerance, small differences between the individual alignments would lead to empty intersections. In the following, we use a neighborhood corresponding to one second. Here, our experiments have shown that changing the neighborhood size within reasonable limits does not have a significant impact on the final results. In a last step, we derive an intersection matrix by setting each matrix cell to one that is one in all three augmented binary matrices; see Fig. 4(d).

The intersection matrix can be thought of as a rough indicator for areas in the cost matrix where the three alignment strategies agree. However, this matrix does not encode an alignment that is constrained by any of the conditions described in Section II-B. Therefore, to derive a final alignment result from this matrix, we first weight the remaining cells in the intersection matrix according to how often they are contained in one of the original three alignments; see Fig. 4(e). Then, interpreting the weighted matrix as a score matrix, we use partial matching to compute an optimal match, $\mathcal{C}$, referred to as the *consistency alignment*; seeFig. 4(f).

In the following, we call a segment in the audio recording (in the MIDI version) *reliable* if it is aligned via $\mathcal{C}$ to a segment in the MIDI version (in the audio recording). Similarly, we call a segment *critical* if it is not aligned. Here, $\mathrm{A}(3, 39)$, $\mathrm{A}(39, 76)$

and $\mathrm{A}(83, 95)$ as well as $\mathrm{M}(8, 45)$, $\mathrm{M}(63, 99)$ and $\mathrm{M}(106, 117)$ are examples of reliable segments in the audio recording and in the MIDI version, respectively. However, the automatic detection of critical sections can also be very useful, as they often contain musically interesting deviations between two versions. For example, consider the critical segment $\mathrm{M}(45, 63)$. This segment contains the bridge found in the MIDI that was omitted in the audio recording as discussed in Section II-B. Here, our method automatically revealed the inconsistencies between the MIDI version and the audio recording. The differences between the audio and the MIDI version in the intro, outro, and solo segments have also been detected. Here, using multiple alignment strategies leads to a more robust detection of critical segments than using just a single approach. The reasons why a segment is classified as critical can be manifold and constitute an interesting subject for a subsequent musical analysis, beyond the scope of the current paper. In this context, however, our approach provides support for such an analysis.

B. Evaluation Setup

To systematically evaluate the performance of our procedure, we use 60 pieces from the classical and 60 pieces from the popular music collection of the RWC music database [26]. For each piece, RWC supplies high-quality MIDI-audio pairs that globally correspond to each other. To obtain a ground-truth alignment for each MIDI-audio pair, we employed a high-resolution global synchronization approach [27] and manually checked the results for errors.

To simulate typical musical and structural differences between the two versions, we severely distorted and modified the MIDI versions as follows. Firstly, we temporally distorted each MIDI file by locally speeding up or slowing down the MIDI up to a random amount between $\pm 50\%$. In particular, we changed the tempo continuously within segments of 20 seconds of length, and added abrupt changes at segment boundaries to simulate musical tempo changes (ritardandi, accelerandi, fermata). Secondly, we structurally modified each MIDI file by replacing several MIDI segments (each having a length of 30 to 40 s) by concatenations of short 2-s snippets taken from random positions within the same MIDI file. In doing so, the length of each segment remained the same. These modified segments do not correspond to any segment in the audio anymore. However, because they are taken from the same piece, the snippets are likely to be harmonically related to the replaced content. Here, the idea is to simulate a kind of improvisation that fits into the harmonic context of the piece, but that is understood as musically different between the audio and the MIDI version (similar to the differences in $\mathrm{A}(75, 83)$ and $\mathrm{M}(99, 107)$, discussed in Section II). Finally, we employ the ground-truth alignment between the original MIDI and the audio. Keeping track of the MIDI modifications, we derive a ground-truth alignment between the modified MIDI and the audio, in the following referred to as $\mathcal{A}^*$.

To present even more challenges to the alignment approaches, we created a second dataset with more *strongly modified* MIDI versions. Here, we not only distorted and replaced randomly chosen MIDI segments as described above, but inserted additional MIDI snippet segments. These additional structural modifications make the synchronization task even harder.

For a given modified MIDI-audio pair, let $\mathcal{A}$ denote an alignment obtained using one of the synchronization strategies described above. To compare $\mathcal{A}$ with the ground-truth alignment $\mathcal{A}^*$, we introduce a quality measure that is based on precision and recall values, while allowing some deviation controlled by a given tolerance parameter $\varepsilon > 0$. The precision of $\mathcal{A}$ with respect to $\mathcal{A}^*$ is defined by

$$\mathrm{P}(\mathcal{A}) = \frac{|\{\gamma \in \mathcal{A} \,|\, \exists \gamma^* \in \mathcal{A}^* : \|\gamma - \gamma^*\|_2 \leq \varepsilon\}|}{|\mathcal{A}|}$$

and the recall of $\mathcal{A}$ with respect to $\mathcal{A}^*$ is defined by

$$\mathrm{R}(\mathcal{A}) = \frac{|\{\gamma^* \in \mathcal{A}^* \,|\, \exists \gamma \in \mathcal{A} : \|\gamma - \gamma^*\|_2 \leq \varepsilon\}|}{|\mathcal{A}^*|}.$$

Here, $\|\gamma - \gamma^*\|_2$ denotes the Euclidean norm between the elements $\gamma, \gamma^* \in [1 : N] \times [1 : M]$; see Section II-B. In our experiments, we use a tolerance parameter ε corresponding to one second. This accuracy is meaningful in view of our chord labeling application. Finally, the F-measure is defined by

$$\mathrm{F}(\mathcal{A}) := \frac{2\mathrm{P}(\mathcal{A})\mathrm{R}(\mathcal{A})}{\mathrm{P}(\mathcal{A}) + \mathrm{R}(\mathcal{A})}.$$

C. Experiments

In a first experiment, we investigate the influence of the smoothing parameter λ on the performance of DTW, our recursive variant of the Smith-Waterman approach (rSW), partial matching (PM), and our proposed consistency alignment (CA). The parameter specifies the number of consecutive features taken into account for the smoothing. On the one hand, increasing λ emphasizes the structural properties of a cost matrix as discussed in Section II-A and is often a requirement to yield an overall robust synchronization result. On the other hand, smoothing can lead to a gradual loss of temporal accuracy in the alignment.

Fig. 5 shows the average precision (bold black), recall (dashed blue), and F-measure (red) for all four alignment procedures using increasing values for λ in combination with fixed values for the other parameters ($\tau = 0.2$, $\gamma = 1$). Here, we used the modified MIDI-audio pairs in Fig. 5(a) and the strongly modified pairs in Fig. 5(b). For computational reasons, we computed the average only over a subset of ten classical and ten pop pieces from the original dataset. Here, looking at the results for DTW, rSW, and PM reveals that increasing λ leads to a higher precision. This indicates an enhanced robustness for all three procedures. However, if the smoothing is applied strongly, the average recall slightly drops, indicating the gradual loss of temporal accuracy. Furthermore, the DTW procedure only yields a rather low average precision for the strongly modified MIDI-audio pairs. Here, the reason is the boundary condition forcing DTW to align both versions as a whole, even if there are locally no musically meaningful correspondences. Still, DTW offers a very high recall value, meaning that the correct alignment is often a true subset of the DTW alignment. This property is exploited by our consistency alignment which is

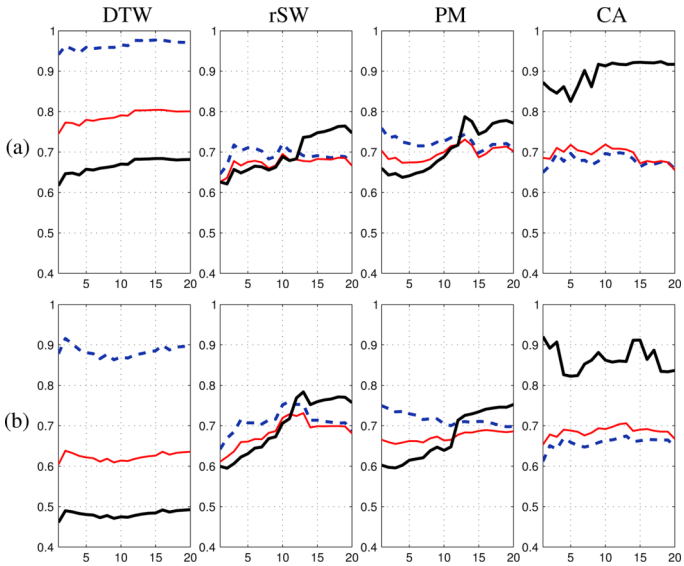

Fig. 5. Effect of the smoothing parameter λ on the alignment accuracy of the DTW, rSW, PM, and CA procedures leaving the remaining parameters fixed ($\tau = 0.2, \gamma = 1$). Horizontal axis: λ. Vertical axis: Precision (bold black), F-measure (red), Recall (dashed blue). (a) Results using modified MIDI-audio pairs. (b) Results using strongly modified MIDI-audio pairs.

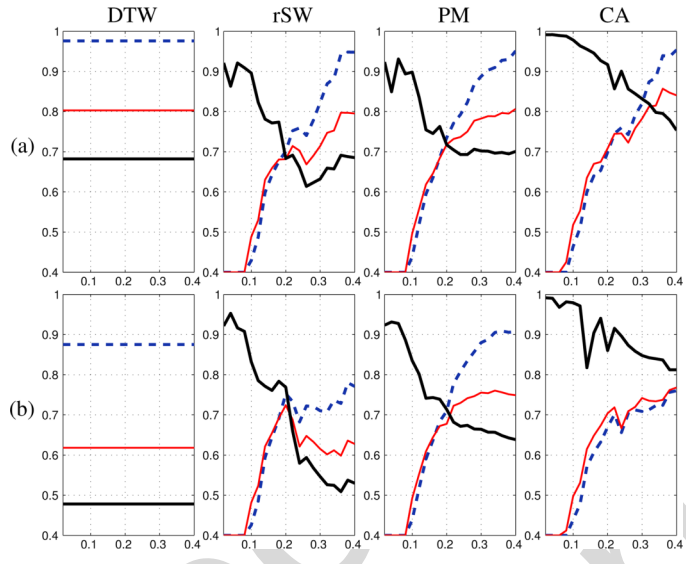

Fig. 6. Effect of the threshold parameter τ on the alignment accuracy of the DTW, rSW, PM, and CA procedures leaving the remaining parameters fixed ($\lambda = 12, \gamma = 1$). Horizontal axis: τ. Vertical axis: Precision (bold black), F-measure (red), Recall (dashed blue). (a) Results using modified MIDI-audio pairs. (b) Results using strongly modified MIDI-audio pairs.

often able to extract the correct parts of the DTW alignment, thus yielding a very high overall precision. Looking at the CA results reveals that our procedure yields a high precision with competitive F-measure and recall values for $\lambda \in [9\!:\!15]$. In the following, we set $\lambda = 12$ which corresponds to 6 s using a feature rate of 2 Hz.

In a second experiment, we analyze the role of the threshold parameter τ. This parameter controls which cells in the cost matrix C become positive score entries in the matrices S and $S_{\geq 0}$; see Section II-D. Fig. 6 shows the results for varying τ while fixing the other parameters ($\lambda = 12, \gamma = 1$). Apart from that, the same experimental setup is used as in the previous experiment. Note that the DTW procedure does not dependent on τ; thus, its results are constant in Fig. 6. As the experiment shows, using very small values for τ, only very similar feature sequences are aligned and both the rSW and PM procedures are able to produce alignments with a high precision. However, this is only possible at the cost of having a very low recall as many correct alignment paths are missed. The break-even point for both procedures is near 0.2. For this value, our proposed consistency alignment yields a recall similar to rSW and PM but the precision is significantly higher. Overall, since the increase in F-measure is noticeable for all procedures up until 0.2 and diminishes beyond, we use $\tau = 0.2$ in the following. This value was also found to deliver reasonable results in the context of audio matching [12].

In a third experiment, we inspected the influence of the gap-penalty parameter γ. This parameter controls the fragmentation level of the alignment resulting from rSW. Here, we found that the influence of this parameter is less significant compared to the other parameters. Still, the experiment indicated that using some penalty for the gaps is needed for rSW to yield a robust alignment in our scenario. Here, choosing γ between 0.5 and 2 yielded very similar results. In the following, we set $\gamma = 1$.

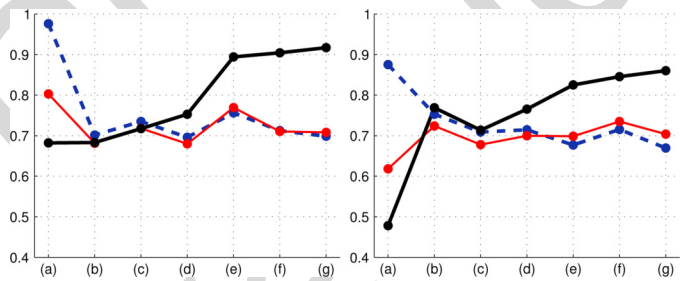

Fig. 7. Effect of using different combinations of alignment procedures to compute the consistency alignment on the alignment accuracy. The parameter settings are fixed ($\lambda = 12, \tau = 0.2, \gamma = 1$). Vertical axis: Precision (bold black), F-measure (red), Recall (dashed blue). Horizontal axis: (a) DTW. (b) rSW. (c) PM. (d) rSW/PM. (e) DTW/PM. (f) DTW/rSW. (g) DTW/rSW/PM. Left: Results using modified MIDI-audio pairs. Right: Results using strongly modified MIDI-audio pairs.

In general, our consistency alignment could be computed using an arbitrary combination of alignment procedures. In a fourth experiment, we investigate the alignment accuracy for all possible combinations of the DTW, rSW, and PM procedures (Fig. 7). All free parameters are fixed for the experiment ($\lambda = 12, \tau = 0.2, \gamma = 1$). A first interesting observation is that all three individual procedures—seeFig. 7(a)–(c)—only yield a rather low average precision; thus, none of them can guarantee a meaningful alignment on its own. Combining any two of the procedures results in a noticeable gain in precision; seeFig. 7(d)–(f). In particular, including DTW is important for a high precision; see Fig. 7(e)–(f). Finally, our proposed combination of all three methods yields the highest precision; see Fig. 7(g). As expected, the recall is slightly lower here, but is still on a competitive level.

In a final experiment, we determined the results for each alignment procedure separately for each available dataset. In Table I, we consider the full classical and popular music datasets (120 recordings in total) using modified and strongly

TABLE I
AVERAGE PRECISION (P), RECALL (R), AND F-MEASURE (F) FOR THE DTW, RSW, PM, AND CA PROCEDURES USING FOUR DIFFERENT DATASETS WITH FIXED PARAMETERS SETTINGS ($\lambda = 12$, $\tau = 0.2$, AND $\gamma = 1$)

(a)

	P	R	F
DTW	0.68	0.99	0.81
rSW	0.84	0.90	0.87
PM	0.86	0.93	0.89
CA	0.91	0.90	0.90

(b)

	P	R	F
DTW	0.52	0.96	0.68
rSW	0.81	0.89	0.85
PM	0.83	0.93	0.87
CA	0.90	0.87	0.88

(c)

	P	R	F
DTW	0.66	0.96	0.78
rSW	0.66	0.65	0.64
PM	0.70	0.71	0.70
CA	0.89	0.64	0.69

(d)

	P	R	F
DTW	0.47	0.87	0.61
rSW	0.62	0.63	0.61
PM	0.66	0.70	0.67
CA	0.89	0.58	0.65

modified MIDI-audio pairs. Here, comparing the results for the modified and the strongly modified MIDI-audio pairs reveals that all procedures are able to cope quite well with the additional structural differences used in the strongly modified pairs. For example, the precision/recall for rSW slightly decrease from 0.84/0.9—see Table I(a)—to 0.81/0.89—see Table I(b)—respectively. Only DTW, again being forced to align the versions as a whole, shows a significant drop in precision. Furthermore, comparing the results for the classical and the popular music dataset shows much lower values for the latter. Here, the underlying reason is that popular music tends to be highly repetitive. Combined with structural differences, this often leads to a higher confusion in the alignment. This is also reflected in Table I where most precision and recall values are significantly lower for the popular music dataset. For example, precision/recall for rSW decrease from 0.84/0.9—see Table I(a)—to 0.66/0.65—seeTable I(c)—respectively. On the contrary, this is not the case for the consistency alignment which achieves a high precision of 0.89 also for the popular music dataset. Again, the recall is still on a competitive level.

In summary, as our experiments illustrate, the consistency alignment is able to deliver alignments with a high precision in combination with a competitive recall. Furthermore, our proposed late-fusion procedure is less dependent on the employed parameter settings or on the given dataset compared to the other individual alignment procedures.

IV. AUTOMATIC CHORD LABELING

In the literature, most chord labeling procedures focus on chord labeling from audio data. Many of these procedures follow a two-step approach. In a first stage, chroma features (see Section II-A) are extracted from an audio file in a framewise fashion. Then, a statistical model is applied to the sequence of chroma vectors that optimizes the match between specific chord templates and local sections of the chromagram. Furthermore, the match of the overall sequence of chords to a global model such as a key or harmonic context is optimized. Typical statistical models applied as part of this second stage are hidden Markov models [28], [29] or more general graphical models [30]. Additional modeling constraints or auxiliary information can further improve chord labeling accuracy. These include the prior identification of the fundamental frequency or root note of each chord before the chromagram is estimated [31], information about the metrical structure [32], information about the musical structure [33], or the musical context [30]. Current state-of-the-art chord labeling programs from audio have reached an identification accuracy of up to 80% as measured by the time overlap between predicted and ground truth chord labels; see [34].

Only very few procedures have been proposed that make use of symbolic music data. Early models such as those by [35] and [36] were designed to perform music-theoretic harmonic analyses (roman numeral analyses) from symbolic music input. Identifying chords in harmonic context (key) was one component within these music-analytic procedures. Both Winograd's and Maxwell's procedures are rule-based and rely heavily on knowledge of Western music theory, designed for the use with Western art music. Reference [37] proposed key identification, chord labeling, and harmonic analysis procedures from a similar perspective. These procedures were implemented by Sleator and Temperley as part of the Melisma Music Analyzer [38], which is mainly based on preference rules and described in more detail below. In [39], the authors presented a hidden Markov model that uses symbolic MIDI data as input and produces a harmonic analysis of a musical piece including key and roman numerals labeling. Reference [40] describes a chord labeling system for MIDI guitar sequences that is based on the symbolic chord labeler proposed by [41]. However, to be applicable in a jazz or Latin music context, the chord labeling system in [40] is specifically designed for the recognition of more complex chords. Their procedure is based on a hybrid mixture of pattern-matching techniques, harmonic context rules, and rules based on stylistic knowledge, and the resulting system is thus somewhat specific to their chosen task.

Even in this very short literature summary, a trend becomes apparent, moving away from rule-based and style-specific chord labeling systems that use explicit, built-in expert knowledge, towards data-driven and statistical reasoning approaches that learn and adapt to musical data from arbitrary styles. In the following, we describe two current chord-labeling systems which are used in our evaluation later in Section V. They both follow a Bayesian statistical approach, which has proven to be very successful in many areas of computational music processing. The following more detailed overviews are given for the reader particularly interested in chord labeling, but are not needed to understand the subsequent evaluation (Section V).

A. Temperley's Melisma

The *Melisma* system [37] for chord labeling and harmonic analysis takes as input a list of MIDI pitches with on- and offset times as well as information about the metrical hierarchy.[1] From these input data, the module *harmony* derives information regarding the tonal pitch class labels of the active MIDI pitches (dissociating enharmonically identical pitches by their harmonic context) and subsequently yields an estimation of the root of the chord summarizing the harmonic content in a time window. This is achieved by a system of three preference rules

[1]Instead of deriving metrical information using the *meter* program from *Melisma*, we provided *harmony* with the correct information about quarter and sixteenth notes directly taken from each MIDI file.

for pitch spelling and the subsequent application of four harmonic preference rules for chord root identification described in [37] and inspired by [42]. From this output, the module *key* firstly infers the keys for all segments (bars) of the entire piece of music. Following [43], key estimation is achieved by a Bayesian algorithm that calculates the probability of a musical feature or structure (here, the key) given an empirical music surface (here, the frequencies of pitch classes in a musical segment). Thus, the probability computation for musical keys, given the pitch classes of a musical piece, is based on the relative frequency with which the 12 scale degrees appear in a key as well as on the probability of a subsequent segment of the piece being in the same key as the previous segment. The pitch class profiles for this Bayesian model are derived from relative frequencies of pitch classes in the Kostka-Payne corpus, a collection of 46 excerpts from a common-practice Western art music. As a last stage, *key* can produce chord labels and a roman numeral analysis, an analysis describing the relation between a chord and the key of the segment the chord is part of.[2] For the evaluation described below, we made use of the information about chord root, mode (major, minor, unspecified) and fifth (perfect, diminished, unspecified) as well as the onset and offset times. This leads to three possible chord classes, namely major, minor, and diminished.

B. Bayesian Model Selection Algorithm for Chord Labeling

Temperley's procedure is a combination of preference rule systems, Bayesian key induction, and a look-up procedure for identifying chord labels given a key and a root note. It depends on some parameters that are hard-coded into the system (e.g., the time window a chord root is inferred for is limited to a beat; then adjacent windows are joined together if they have the same chord root), other parameters need to be set by the user of the programs and still others (e.g., pitch class profiles) can be learned from data.

In contrast, [44] proposed a Bayesian approach for chord labeling, here abbreviated as *RLM*, that aims to incorporate all relevant parameters into the same modeling procedure, and then uses Bayesian model selection to choose the most likely chord model given the musical data. Because of their prevalence in popular music, the current model focuses on triad chords. However, the model can be extended in a straightforward manner to include more complex chords (e.g., 7th-chords or chords constructed from fourths and fifths instead of thirds). It assumes six possible chord classes: Major, minor, diminished, augmented, sus2, and sus4. The model-selection procedure models three independent aspects relevant for assigning a chord label:

- The proportion of triad notes t to non-triad notes $\bar{t}$.
- The proportion of root r, middle m, and upper u tone among the tones of a triad.
- The subdivision of a bar into time windows having the same chord. Here, all eight possible divisions of the bar are considered that do not subdivide the quarter beat.

The model (for a single time window) to infer the chord label, c, is built over the proportion of triad to non-triad tones and the proportions of the three triad tones within the overall proportion of triad tones. Each of the conditional distributions is modeled by a Dirichlet distribution for proportions [44]. Of all possible models for chord labeling a bar of music, the most likely one is chosen given the musical data using Bayes' rule. Here, not just the probability of the most likely chord label is taken into account for a given division model but the evidence from all possible chord labels and Dirichlet parameters is added together for each model of subdividing the bar. From the resulting probability distribution, the most likely model of bar subdivision is then selected for chord labeling. The necessary estimation of the parameters of the Dirichlet distributions for *RLM* was performed using a maximum-likelihood approach on a training corpus of 233 bars from 16 different pop songs using hand-annotated chord labeling data.

V. Cross-Version Chord Labeling Evaluation

Exploiting the availability of multiple versions of a given piece of music, we have suggested the general concept of a cross-version analysis for comparing and/or combining analysis results across the versions. We now exemplarily apply this concept in the context of harmony analysis. In particular, we automatically evaluate the two MIDI-based chord labelers *RLM* and *Melisma* from Section IV on the well-known Beatles dataset, where chord annotations are available for corresponding audio recordings (Section V-A). We then evaluate the two symbolic chord labelers described in Section IV, whose performance has not been clear so far, since no ground truth labels have been available on a larger scale (Section V-C). As an even more important contribution, we discuss a cross-version visualization (Section V-B) and demonstrate how such a functionality can greatly support a user in a qualitative analysis of the recognition errors (Section V-D).

A. Experimental Setup

In our evaluation, we exploit the audio data chord annotations provided by Christopher Harte, who manually annotated all 180 songs of the 12 Beatles studio albums [45]. Harte's annotations are generally accepted as the de-facto standard for evaluating audio-based chord labeling methods. Transferring these annotations from the acoustic to the symbolic domain allows for an efficient reuse of the existing ground truth for the evaluations of symbolic chord labelers. Furthermore, having a common set of ground truth across all available musical domains presents a starting point to identify exactly those positions in a piece where a method relying on one music representation has the advantage over another method, and to investigate the underlying musical reasons.

Our evaluation dataset consists of 112 songs out of the 180 songs. For these 112 songs, we not only have an audio recording with annotated chord labels, but also a corresponding MIDI version. Given a MIDI file and a corresponding audio recording, we start our evaluation by computing a MIDI-audio alignment. Be-

[2]Chord labels are only part of *key*'s internal data structure and its sole output is the roman numeral analysis. However, unsurprisingly, in tests with popular music *key*'s roman numeral analysis produced many uninterpretable results assigning the label *Chr* to many chords (*Chr* stands for chromatic and designates in Temperley's terminology a chord that cannot be derived from a major or minor scale by adding thirds to a scale note). We therefore by-passed the roman numeral analysis and accessed *key*'s internal data structure for chord labels.

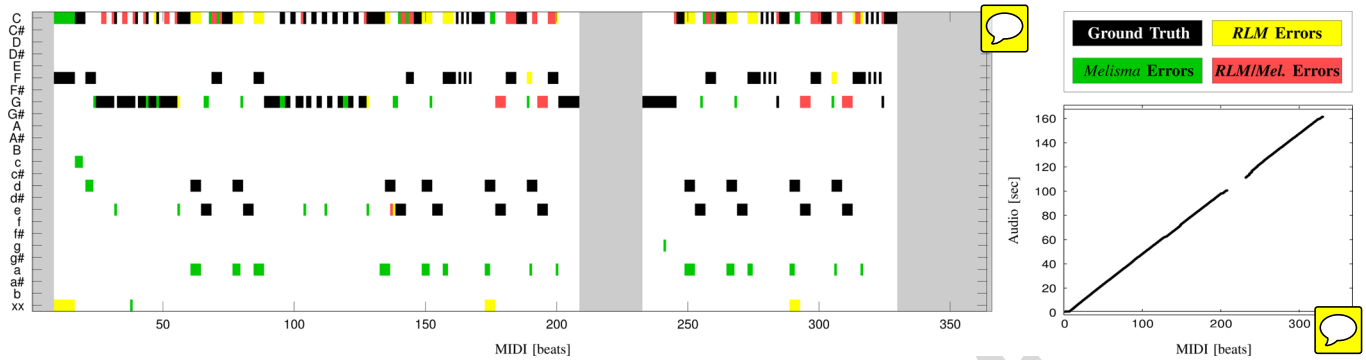

Fig. 8. Cross-version chord evaluation for the song *Getting Better*. (Left) Overlay of two MIDI-based chord labeling results (*Melisma* and *RLM*) and manually generated audio-based chord labels. (Right) Consistency alignment (horizontal axis specifies MIDI time in beats and vertical axis specifies audio time in seconds).

cause the MIDI versions often differ significantly, at local level, from the audio recordings, we cannot simply employ global synchronization techniques. Therefore, we employ our consistency alignment, as described in Section III, which identifies those sections that can be aligned reliably. Using the linking information provided by the alignment, we compute for each MIDI beat the corresponding position in the audio version. Using this linking information, we then transfer the audio-based chord labels to the MIDI version. If more than one audio chord label exists in the audio segment associated with a MIDI beat, we simply choose the predominant chord label as MIDI annotation. As the result, we obtain a beatwise chord label annotation for the MIDI version.

For our evaluation, we compare the transferred ground truth annotations to the automatically generated chord labels obtained from *Melisma* and *RLM* on the basis of the 12 major and the 12 minor chords. Therefore, using the interval comparison of the triad as used for MIREX 2010 [34], all ground truth chord labels are mapped to one of these 24 chords. Here, both a seventh chord and a major seventh chord are mapped to the corresponding major chord. However, augmented, diminished, or other more complex chords cannot be reduced to either major or minor and therefore are omitted from the evaluation.

B. Visualization

Using synchronization techniques allows for visualizing different chord recognition results simultaneously for multiple versions. Such cross-version visualizations turn out to be a powerful tool for not only analyzing the chord label results but also for better understanding the underlying music material [46]. We introduce our visualization concept by means of a concrete example shown in Fig. 8. Here, the chord labels generated by *Melisma* and *RLM* are visualized along with the transferred ground truth annotations using a common MIDI time axis given in beats (horizontal axis). The vertical axis represents the 24 major and minor chords, starting with the 12 major chords and continuing with the 12 minor chords. Associated with each beat is a black entry representing the ground truth chord label that we transferred to the MIDI files. For example, in Fig. 8, a G major chord label is assigned to beat 50. The colored entries in the figure are used to indicate where the two automatic chord labelers differ from the manual annotation. Here, yellow and green entries indicate that *RLM* and *Melisma* differ from the manual annotation, respectively. For example, in the beginning of the song the green entries show that *Melisma* detected a C major chord, while the ground truth specified an F major chord. If a chord labeler generated a chord label that cannot be reduced to either major or minor, then this is indicated by a colored entry in the "xx" row. For example, in the beginning of the song, *RLM* detected a complex chord corresponding to a yellow entry in the "xx" row. Sometimes, both automatic chord labelers differ from the ground truth, but agree on the same chord label. Such consistent deviations from the ground truth are marked in red. An example can be found around beat 200, where both automatic chord labelers specify a C major chord instead of an F major chord in the ground truth. Furthermore, areas in the figure with a gray background indicate beats for which no ground truth is available. For example, in Fig. 8, this can be observed between beat 210 and 230. Here, our consistency alignment, given on the right in the figure, shows that this section in the MIDI file could not be reliably aligned to a corresponding section in the audio. Furthermore, a ground truth annotation might also be unavailable for a beat if the chord label at that position is irreducible to major or minor—for example, if the chord label specifies an augmented chord.

Overall, our visualization allows for the identification of two different classes of inconsistencies. On the one hand, red entries in the visualization reveal positions, where the two chord labelers consistently differ from the ground truth. Here, the reason for the error may be of extrinsic or musical nature, independent of the specific chord labeler. On the other hand, yellow and green entries indicate intrinsic errors of the respective chord labeler. Thus, our visualization constitutes a useful tool to identify interesting or problematic passages in the audio recording.

C. Quantitative Evaluation

We now quantitatively evaluate the two MIDI-based chord labelers. Table II presents the results for nine exemplarily chosen songs as well as an average over all 112 pieces in our database. For each song, the precision values of *Melisma* and *RLM* are listed. Here, precision indicates the percentage of the manually annotated beats correctly classified by the respective chord labeler. Also, the alignment coverage (AC), which specifies the

TABLE II
RESULTS OF THE CROSS-VERSION CHORD EVALUATION FOR *RLM* AND *MELISMA*. THE FOUR COLUMNS INDICATE THE PIECE/DATASET, THE ALIGNMENT COVERAGE (AC), AS WELL AS THE PRECISION (PREC) FOR THE TWO METHODS

Piece	AC	Prec *RLM*	Prec *Melisma*
AnotherGirl	97	98	60
DoctorRobert	99	76	60
EightDaysAWeek	99	92	74
EverybodysTryingToBeMyBab	95	71	85
GettingBetter	83	60	52
GoodDaySunshine	82	85	55
InMyLife	97	90	75
IWannaBeYourMan	91	61	42
Money	56	38	11
Average	89	75	57
Average over all 112 songs	86	82	72

percentage of the MIDI version that has been aligned to the respective audio version, is listed.

As can be seen from Table II, the precision of *RLM*, averaged over all 112 songs, is 82%, whereas that of *Melisma* is only 72%. Using Bayesian model selection, *RLM* seems to be more data adaptive and performs better in our experiments than *Melisma*, depending on some hard-coded parameters. Furthermore, *Melisma* is tuned towards classical music, whereas *RLM* focuses on popular music, which might be advantageous with regard to the Beatles dataset.

Even though such a quantitative evaluation gives a general indication on the algorithms' performances, it is not very helpful for the understanding of the algorithmic or musical reasons of the recognition errors. We now show how our visualization framework can be used for a more in-depth analysis of the chord recognition results.

D. Qualitative Evaluation

Our cross-version visualization directly reveals two different types of errors: *extrinsic errors* that are independent of the employed chord labeling strategy (marked by red entries) as well as *intrinsic errors* of the two chord labelers (marked by yellow and green entries). In the following, we further detail on this observation by exemplarily performing a qualitative error analysis by means of some concrete song examples.

First, we discuss some typical intrinsic errors of the two chord labelers. For *Melisma*, it turned out that one main error source consists in confusing major and minor. Here, the song *Another Girl*—see Fig. 9(a)—serves as an example. As can be clearly seen from the visualization, *Melisma* recognizes most of the time A minor instead of A major. On the contrary, most of *RLM*'s errors are produced by specifying a complex chord label instead of a major or minor label in the ground truth. For example, looking at the song *Doctor Robert*—see Fig. 9(b)—one notices that an A major chord is annotated from beat 1 to beat 57 in the ground truth, whereas *RLM* often specifies a more complex chord corresponding to the "xx" row. Taking into account six different chord classes (major, minor, diminished, augmented, sus2, and sus4), *RLM* is susceptible to choose such a complex chord label instead of a simple major or minor chord label. Here, a manual inspection revealed that also simplifying assumptions in the manually generated audio

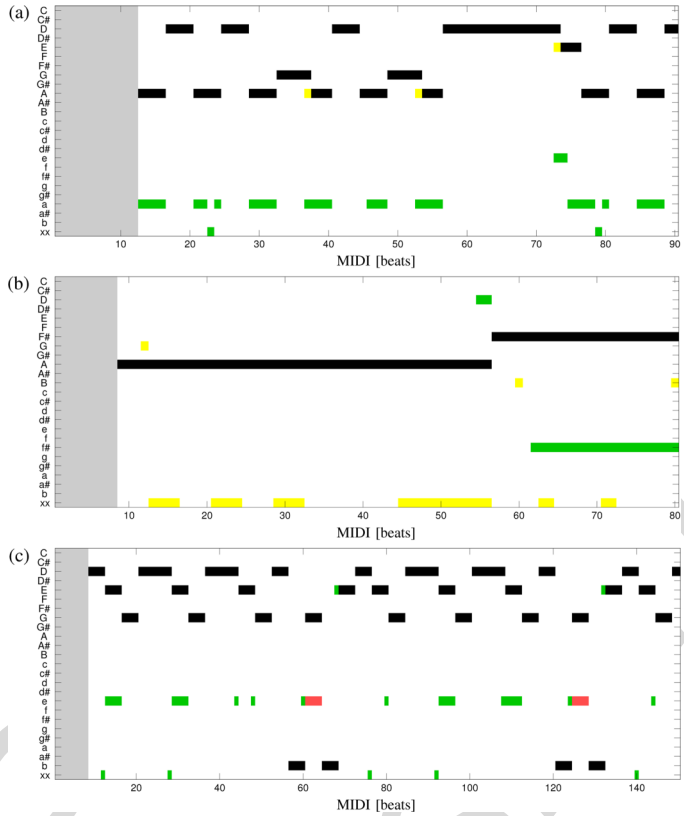

Fig. 9. Cross-version chord label visualization for the songs (a) *Another Girl* (beats 1–90), (b) *Doctor Robert* (beats 1–80), and (c) *Eight Days A Week* (beats 1–150).

annotations (taken as ground truth) and the reduction process are sources for confusion and ambiguity. Furthermore, also in the *Doctor Robert*—see Fig. 9(b)—example, *Melisma*'s confusion of major and minor appears again, where F♯ minor is recognized instead of F♯ major from beat 62 to beat 80.

The second type of error sources are extrinsic errors, which are errors that appear consistently for both chord labelers (marked by red entries). Such consistent misclassifications may appear for several reasons. Having performed an in-depth error analysis allows us to categorize these errors into the following four subclasses. Firstly, a consistent misclassification can appear due to errors in the synchronization. Secondly, inaccuracies in the manual ground truth annotations can be responsible for consistent misclassifications. Thirdly, a harmonic difference between the MIDI and the audio version may lead to a consistent deviation of the two chord labelers from the ground truth. Finally, as the fourth subclass, we detected errors that are caused by musical reasons. For example, the use of suspensions or the presence of passing notes and other nonharmonic tones often lead to local chord ambiguities. In particular, the leading voice often contains nonharmonic tones with regard to the underlying harmony. Precisely this phenomenon appears, e.g., in the song *Eight Days A Week*—see Fig. 9(c)—at beat 60, where the underlying chord is G major, which is also the labeled chord in the ground truth. However, both chord labelers specify an E minor chord here. This is due to the nonharmonic tone E in the leading voice, which, together with the tones G and B, forms an E minor chord.

VI. Conclusions and Future Work

We have introduced a cross-version analysis framework for comparing analysis results from different musical domains. As technical basis, we presented a novel synchronization approach that yields high-precision partial alignments by combining multiple alignment strategies. We demonstrated the utility of our framework in the context of harmonic analysis, where we evaluated MIDI-based chord labeling methods using audio-based ground truth annotations in a cross-domain fashion. The subsequent manual analysis and discussion of the critical passages exemplified how our framework facilitates interdisciplinary research by bridging the gap between music signal processing (SP) and music sciences. Visualizations and interfaces based on our framework allow even a technically unexperienced user to perform an error analysis of automatically generated annotations.

In the future, we plan to deploy our cross-version framework in other music analysis tasks, such as musical structure analysis and score-informed source and voice separation [47]–[49]. Here, the availability of closely related sources of information, such as alternate recordings, cover songs, multitrack recordings of original studio sessions, or score representations including MIDI versions, allows for innovative methods that may solve otherwise intractable problems.

References

[1] G. Wiggins, E. Miranda, A. Smaill, and M. Harris, "A framework for the evaluation of music representation systems," *Comput. Music J.*, vol. 17, no. 3, pp. 31–42, 1993.

[2] H. Fujihara, M. Goto, J. Ogata, and H. G. Okuno, "LyricSynchronizer: Automatic synchronization system between musical audio signals and lyrics," *IEEE J. Select. Topics Signal Process.*, vol. 5, no. 6, pp. 1252–1261, 2011.

[3] M.-Y. Kan, Y. Wang, D. Iskandar, T. L. Nwe, and A. Shenoy, "LyricAlly: Automatic synchronization of textual lyrics to acoustic music signals," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 338–349, 2008.

[4] R. Mayer and A. Rauber, "Musical genre classification by ensembles of audio and lyrics features," in *Proc. Int. Society for Music Information Retrieval Conf. (ISMIR)*, Miami, FL, 2011, pp. 675–680.

[5] C. McKay and I. Fujinaga, "Improving automatic music classification performance by extracting features from different types of data," in *Proc. ACM SIGMM Int. Conf. Multimedia Information Retrieval*, Philadelphia, PA, 2010, pp. 257–266.

[6] N. Hasler, B. Rosenhahn, T. Thormählen, M. Wand, J. Gall, and H.-P. Seidel, "Markerless motion capture with unsynchronized moving cameras," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR)*, Miami, FL, 2009, pp. 224–231.

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

[8] M. Müller and D. Appelt, "Path-constrained partial music synchronization," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Las Vegas, NV, 2008, pp. 65–68.

[9] S. Ewert, M. Müller, and R. B. Dannenberg, "Towards reliable partial music alignments using multiple synchronization strategies," in *Proc. Int. Workshop Adaptive Multimedia Retrieval (AMR), Lecture Notes in Computer Science (LNCS)*, Madrid, Spain, 2009, vol. 6535, pp. 35–48.

[10] M. Müller, *Information Retrieval for Music and Motion*. New York: Springer-Verlag, 2007.

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

[12] M. Müller, F. Kurth, and M. Clausen, "Audio matching via chroma-based statistical features," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, 2005, pp. 288–295.

[13] J. Serrà, E. Gómez, P. Herrera, and X. Serra, "Chroma binary similarity and local alignment applied to cover song identification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, pp. 1138–1151, Oct. 2008.

[14] J. Foote, "Visualizing music and audio using self-similarity," in *Proc. ACM Int. Conf. Multimedia*, Orlando, FL, 1999, pp. 77–80.

[15] M. Müller and F. Kurth, "Enhancing similarity matrices for music audio analysis," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Toulouse, France, 2006, pp. 437–440.

[16] L. Rabiner and B.-H. Juang, Fundamentals of Speech Recognition, ser. Prentice Hall Signal Processing Series, 1993.

[17] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *J. Molec. Biol.*, vol. 147, pp. 195–197, 1981.

[18] P. A. Pevzner, *Computational Molecular Biology: An Algorithmic Approach*. Cambridge, MA: MIT Press, 2000.

[19] V. Arifi, M. Clausen, F. Kurth, and M. Müller, "Synchronization of music data in score-, MIDI- and PCM-format," *Comput. Musicol.*, vol. 13, pp. 9–33, 2004.

[20] R. B. Dannenberg, "An on-line algorithm for real-time accompaniment," in *Proc. Int. Computer Music Conf. (ICMC)*, 1984, pp. 193–198.

[21] C. Raphael, "A probabilistic expert system for automatic musical accompaniment," *J. Computat. Graph. Statist.*, vol. 10, no. 3, pp. 487–512, 2001.

[22] R. B. Dannenberg and C. Raphael, "Music score alignment and computer accompaniment," *Commun. ACM, Special Issue: Music Information Retrieval*, vol. 49, no. 8, pp. 38–43, 2006.

[23] N. Orio, S. Lemouton, and D. Schwarz, "Score following: State of the art and new developments," in *Proc. Int. Conf. New Interfaces for Musical Expression (NIME)*, Montreal, QC, Canada, 2003, pp. 36–41.

[24] C. Raphael, "A hybrid graphical model for aligning polyphonic audio with musical scores," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, Barcelona, Spain, 2004, pp. 387–394.

[25] A. Cont, "A coupled duration-focused architecture for real-time music-to-score alignment," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 6, pp. 974–987, 2010.

[26] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical and jazz music databases," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, Paris, France, 2002.

[27] S. Ewert, M. Müller, and P. Grosche, "High resolution audio synchronization using chroma onset features," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, Taipei, Taiwan, 2009, pp. 1869–1872.

[28] A. Sheh and D. P. W. Ellis, "Chord segmentation and recognition using EM-trained hidden Markov models," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, Baltimore, MD, 2003.

[29] K. Lee and M. Slaney, "Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 291–301, 2008.

[30] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 18, no. 6, pp. 1280–1289, 2010.

[31] M. Ryynänen and A. Klapuri, "Automatic transcription of melody, bass line, and chords in polyphonic music," *Comput. Music J.*, vol. 32, no. 3, pp. 72–86, 2008.

[32] H. Papadopoulos and G. Peeters, "Simultaneous estimation of chord progression and downbeats from an audio file," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 121–124.

[33] N. C. Maddage, "Automatic structure detection for popular music," *IEEE Multimedia*, vol. 13, no. 1, pp. 65–77, 2006.

[34] MIREX 2010. Audio Chord Estimation Subtask, Retrieved 17.09.2010. [Online]. Available: http://www.music-ir.org/mirex/wiki/2010:Audio_Chord_Estimation.

[35] T. Winograd, "Linguistics and the computer analysis of tonal harmony," *J. Music Theory*, vol. 12, pp. 2–49, 1968.

[36] J. H. Maxwell, "Understanding Music with AI," in *An Expert System for Harmonic Analysis of Tonal Music*. Cambridge, MA: MIT Press, 1992, pp. 335–353.

[37] D. Temperley, *The Cognition of Basic Musical Structures*. Cambridge, MA: MIT Press, 2001.

[38] D. Sleator and D. Temperley, The Melisma Music Analyzer, 2003. [Online]. Available: http://www.link.cs.cmu.edu/music-analysis/.

[39] C. Raphael and J. Stoddard, "Functional harmonic analysis using probabilistic models," *Comput. Music J.*, vol. 28, no. 3, pp. 45–52, 2004.

[40] R. Scholz and G. Ramalho, "COCHONUT: Recognizing complex chords from MIDI guitar sequences," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, 2008, pp. 27–32.

[41] B. Pardo and W. Birmingham, The Chordal Analysis of Tonal Music University of Michigan, Dept. of Electrical Engineering and Computer Science, Tech. Rep. CSE-TR-439-01, 2001.
[42] F. Lerdahl and R. Jackendoff, *A Generative Theory of Tonal Music*. Cambridge, MA: MIT Press, 1983.
[43] D. Temperley, *Music and Probability*. Cambridge, MA: MIT Press, 2007.
[44] C. Rhodes, D. Lewis, and D. Müllensiefen, "Bayesian model selection for harmonic labelling," in *Proc. Int. Conf. Mathematics and Computation in Music (MCM), Revised Selected Papers (Communications in Computer and Information Science)*. Springer, 2009, pp. 107–116.
[45] C. Harte, M. Sandler, S. Abdallah, and E. Gómez, "Symbolic representation of musical chords: A proposed syntax for text annotations," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, London, U.K., 2005.
[46] V. Konz, M. Müller, and S. Ewert, "A multi-perspective evaluation framework for chord recognition," in *Proc. 11th Int. Conf. Music Information Retrieval (ISMIR)*, Utrecht, The Netherlands, 2010, pp. 9–14.
[47] J. Woodruff, B. Pardo, and R. B. Dannenberg, "Remixing stereo music with score-informed source separation," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, 2006, pp. 314–319.
[48] K. Itoyama, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno, "Instrument equalizer for query-by-example retrieval: Improving sound source separation based on integrated harmonic and inharmonic models," in *Proc. Int. Conf. Music Information Retrieval (ISMIR)*, Philadelphia, PA, 2008, pp. 133–138.
[49] Y. Han and C. Raphael, "Informed source separation of orchestra and soloist," in *Proc. Int. Society for Music Information Retrieval Conf. (ISMIR)*, Utrecht, The Netherlands, 2010, pp. 315–320.

Sebastian Ewert (S'??) **AUTHOR: WHAT YEAR?** received the M.Sc. degree (Diplom) in computer science from the University of Bonn, Bonn, Germany, in 2007. He is currently pursuing the Ph.D. degree in the Multimedia Signal Processing Group headed by Prof. M. Clausen, Bonn University, under the supervision of M. Müller.

He has been a researcher in the field of music information retrieval since 2008. His research interests include audio signal processing and machine learning with applications to automated music processing. His particular interests concern the design of musically relevant audio features as well as music synchronization and source separation techniques.

Meinard Müller (M'??) **AUTHOR: WHAT YEAR?** the M.Sc. degree (Diplom) in mathematics and the Ph.D. degree in computer science at the University of Bonn, Bonn, Germany.

In 2002–2003, he conducted postdoctoral research in combinatorics at the Mathematical Department of Keio University, Japan. In 2007, he finished his Habilitation at Bonn University in the field of multimedia retrieval writing a book titled *Information Retrieval for Music and Motion*, which appeared as Springer monograph. Currently, he is a member of the Saarland University and the Max-Planck Institut für Informatik, where he leads the research group Multimedia Information Retrieval and Music Processing within the Cluster of Excellence on Multimodal Computing and Interaction. His recent research interests include content-based multimedia retrieval, audio signal processing, music processing, music information retrieval, and motion processing.

Verena Konz received the Diploma degree in mathematics from the University of Cologne, Cologne, Germany, in 2008. She is currently pursuing the Ph.D. degree at Saarland University and the Max-Planck Institut für Informatik, Saarbrücken, Germany, where she is working within the Cluster of Excellence on Multimodal Computing and Interaction in the Multimedia Information Retrieval and Music Processing Group.

In addition, she studied music at the Hochschule für Musik Köln, Germany. Her research interests include music processing, music information retrieval, and computer-based harmonic analysis.

Daniel Müllensiefen studied systematic and historic musicology at the universities of Hamburg and Salamanca (Spain). He received the Ph.D. degree in memory for melodies in 2005.

He was a post-doctoral research fellow at the Department of Computing at Goldsmiths, University of London, London, U.K., from 2006–2009. Since 2009, he has been a lecturer and co-director of the M.Sc. program in Music, Mind and Brain in the Psychology Department at Goldsmiths. His research interests include all areas of music psychology, computational musicology, and psychological theories of similarity perception.

Geraint A. Wiggins studied mathematics and computer sciences at Corpus Christi College, Cambridge, and received Ph.D. degrees in artificial intelligence and in musical composition from the University of Edinburgh.

He is a Professor of Computational Creativity at Queen Mary, University of London, London, U.K. His research career has specialized in generality, covering computational linguistics, computational logic, computational modeling of music perception and cognition, and computational creativity. He was one of the founders of the computational creativity research area, and is the founding chair of the international Association for Computational Creativity. From 2000–2004, he chaired the Society for the Study of Artificial Intelligence and the Simulation of Behaviour, the U.K. learned society for AI and cognitive science.

Dr. Wiggins is an associate editor of *Musicae Scientiae*, the journal of the European Society for the Cognitive Sciences of Music, a consulting editor of *Music Perception*, and serves on the editorial board of the *Journal of New Music Research*.